Out-of-Distribution Generalization in Biological and Artificial Intelligence

a dissertation presented
by
Spandan Madan
to
The School of Engineering and Applied Sciences

in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy
in the subject of
Computer Science

Harvard University
Cambridge, Massachusetts
September 2024

This thesis is dedicated to my family back home in India—my father, Rakesh Madan, my mother, Poonam Madan, my brother and sister-in-law, Mayank and Kritika Madan, and finally, my baby niece, Parthavi Madan. Without their endless love and support, this past decade of my life in research would never have been possible.

Acknowledgments

I would like to extend my deepest gratitude to my advisors, Hanspeter Pfister and Gabriel Kreiman. As I often tell my friends, while Hanspeter taught me to zoom out, Gabriel taught me to zoom in.

Over nine years ago, Hanspeter took me under his wing as a young, eager master's student with a keen interest in research but no formal background in Computer Science. Through the highs and lows of both research and life, he has been a guiding force, teaching me much more than just science—he has taught me how to live. Above all, I have always felt his commitment to my growth and development. As a scientist, he has taught me above all to see the bigger picture, to approach problems with a broad perspective, and to value the importance of interdisciplinary thinking. One of the most valuable gifts Hanspeter gave me was the freedom to explore. He allowed me to learn anything, ask any question, and collaborate with anyone I needed to. This academic freedom was essential for the interdisciplinary work I did during my PhD, allowing me to grow as an independent thinker and a scientist.

On the other hand, over the past 5 years, Gabriel has taught me almost everything I know today about nuts and bolts of digging deeper into a question as a scientist. I was drawn to his unbridled curiosity to answer profound questions, and his mentorship has been transformative, shaping my approach to research and instilling in me a deep appreciation for the scientific method. His rigor has been a true inspiration, and his passion for discovery and his relentless pursuit of answers has been contagious. Working with Gabriel has taught me to embrace challenges with enthusiasm, to think critically and creatively, and to never shy away from asking difficult questions.

This journey would not have been possible without the invaluable mentorship and expertise of my senior mentors. I am especially grateful to Xavier Boix, with whom I collaborated on some of the most exhilarating and creative research projects of my life. I also want to thank Mengmi Zhang, Tzu-Mao Li, and Will Xiao whose expertise helped bring many of my ideas to life.

On a personal note, my nine years in Cambridge have been an unforgettable adventure. Full of ups and downs, but these have been some of the best years of my life thanks to my friends here. It would be impossible to name everyone, but I want to like to especially thank Abhimanyu Dubey, Chetan Krishna, Elisa Pavarino, Fenil Doshi, Geet Kalra, Harshika Bisht, Ishaan Grover, Jenny Zheng, Kamakshi Patapathi, Mayank Ojha, Pranav Misra, Ravi Srinivasan, and Serena Bono. They have been nothing less than a second family to me in Cambridge. I would also like to thank my labmates from both VCG and from KreimanLab who have been a constant source of knowledge, fun and support.

Finally, I owe an immense debt of gratitude to my family for their unwavering support throughout this journey. In my master's thesis 6 years ago, I remember writing that science is a never-ending pursuit. Only now do I fully understand how true those words were. I am deeply grateful to my family for taking the time to understand me, and why it was inevitable for me to take the path I have taken over the past decade.

Thesis advisors: Hanspeter Pfister, Gabriel Kreiman Spandan Madan

Out-of-Distribution Generalization in Biological and Artificial Intelligence

Abstract

This past decade has seen unprecedented success in Artificial Intelligence (AI), pushing the frontiers in ways most experts could have never predicted. However, most of this success has come in the form of performing well inside the data distribution the models have been trained with. Out-of-distribution (OOD) generalization still remains the achilles heel of modern AI. In contrast, biological systems exhibit a remarkable ability to adapt to novel situations. This thesis addresses this critical generalization gap, by studying Biological and Artificial Intelligence in tandem. The work presented includes new mathematical frameworks designed to better formalize generalization, behavioural benchmarks to identify the limits of both human and AI generalization capabilities, experiments to identify the underlying mechanisms driving generalization in both brains and neural networks, and engineering solutions to incorporate these findings to improve AI. To this end, this thesis presents scientific contributions made to the fields of Machine Learning, Computer Vision, Computer Graphics, Computational Neuroscience, and Psychophysics. Throughout the thesis, the goal of this work has been to advance our understanding and improve OOD generalization by working at the intersection of biological and artificial intelligence.

Thesis advisors: Hanspeter Pfister, Gabriel Kreiman Spandan Madan

Out-of-Distribution Generalization in Biological and Artificial Intelligence

Thesis Statement

Modern machine learning is driven largely by two factors—architectural advancement and dataset size. However, a large part of it is driven by accuracy on in-distribution performance i.e. training and testing on i.i.d. data drawn from the same distribution. Unfortunately, this approach has fallen short of achieving intelligence that can effortlessly adapt and generalize beyond the training data distribution like biological systems can. This thesis asserts that out-of-distribution (OOD) generalization is not just a matter of scale or architecture, but is significantly driven by a third factor—data diversity. Diverse training data leads to the development of invariant representations, which allow significantly improved generalization on samples drawn from distributions never seen during training. This thesis confirms these claims through computational, behavioural, and eletrophysiological experiments.

Contents

Part I

Preliminaries

Intelligence is the ability to adapt to change.

Stephen Hawking

1 Introduction

On January 15, 2009, US Airways Flight 1549 took off from New York's LaGuardia Airport. Shortly after, the aircraft struck a flock of geese and lost power in both engines. No engine power, no immediate airport for an emergency landing, and only a matter of seconds to decide. Despite never having trained for it, Captain Sully Sullenberger made the split-second decision of landing the plane in the Hudson river. He managed to perform a near-perfect water landing, saving all 155 people onboard. Captain Sullenberger has long admitted that the only training he ever received for landing on water was a classroom discussion over 40 years prior to the day he achieved the heroic feat[115].

This story serves as a poignant example of the delicate interplay exhibited in human judgment between calling on experience, and adapting to the unexpected. There is an elusive capability which enabled Captain Sully to adapt his expert training to land that plane in water. And albeit at a much smaller scale, it is intuitively the same capability we all tap into in situations like communicating with the locals on a vacation when we share no language in common. The human experience is defined by such ability to constantly adapt and respond to a situation that lie far outside the typical conditions for which one has been trained.

This past decade has seen Artificial Intelligence (AI) evolve in ways most experts could have never predicted. But the question at the heart of this thesis, and one I would like the reader to ask themselves is if they would rather prefer sitting in an aircraft piloted by Captain Sully or a highly skilled AGI assuming neither of the two had prior experience with landing on water. I know my answer, and if the reader feels they'd rather trust a human to adapt to unforeseen circumstances better than AI, it's likely because intuitively we all recognize a fundamental limitation of current AI systems. While these systems excel at tasks within their training domains, they often struggle to generalize to beyond scenarios they have seen in their training data.

This lack of faith in capability of AI to adapt to unforeseen circumstances is as well founded as can be. Take for example, Microsoft's Tay chatbot , which quickly became racist and offensive after interacting with users online, reflecting the biases present in the training data. Or consider the numerous instances where self-driving cars have encountered difficulties due to unexpected road conditions or objects that were not present in their training data. For example, self-driving cars have struggled to handle rare weather events such as heavy snow or dense fog, or in construction zones /cite where the presence of temporary road markings, equipment, and workers can confuse their sensors and algorithms. Furthermore, self-driving cars have struggled to handle unusual pedestrian behavior, such as jaywalking or running across the street, which can deviate from the patterns of pedestrian behavior that the cars were trained on .

Hopefully, the above has been sufficient to describe why to adapt beyond scenarios encountered before is a fundamental building block necessary for any agent claiming to be intelligent. These failures in the real-world are well understood by experts. In Artificial Intelligence terminology, this capability to generalizing beyond the training data is referred to as Out-of-Distribution (OOD) generalization, and the gap between the generalization capabilities of humans and machines is often referred to as the Generalization Gap. Out-of-Distribution generalization is well studied in Machine Learning and Artificial Intelligence literature and AI models are widely understood to struggle when tested on samples out of the training distribution.

In computer vision, models trained to recognize objects in images have shown significant performance drops when faced with objects under unseen viewpoints or color. Similarly, in natural language processing (NLP), AI systems that perform well on specific datasets can fail to understand or generate coherent responses when faced with different linguistic styles or domain-specific jargon. In the audio domain, speech recognition systems trained on clear, accent-neutral speech often struggle with regional accents or background noise, highlighting their limitations in handling audio data that is out of their training distribution. Even in domains like medical diagnosis, where AI has shown promise, models can underperform when applied to patient populations that differ from those seen during training, raising concerns about the applicability of these systems across diverse demographic groups.

In stark contrast, humans often adapt effortlessly. Or do they? Humans often serve as the unreachable, upper-bound for many AI models, representing the gold standard. But, the history of when and if humans can truly generalize beyond what they've experienced was deeply contested. Even now, several academics, including my advisor Gabriel Kreiman, question if humans truly can generalize, and work on identifying the limits of human generalization. In the next chapter, we turn to the history of human knowledge, and see how these ideas in AI stem from epistemology—the branch of philosophy that examines the nature, origin, and limits of knowledge.

The belief that the future will resemble the past
is a matter of instinct, not reason.

David Hume

2 The past and the present of Out-Of-Distribution generalization

Generalizing beyond experience is not merely a modern concern born of computational advancements, but rather the continuation of a philosophical inquiry that spans centuries. This chapter is undoubtedly the most ambitious in this thesis. The author hopes to walk the reader through the past 3 centuries of philosophical ideas on how knowledge emerges from experience. Then, we'll explore the modern era of defining these ideas mathematically through data distributions. Finally, this chapter sets the tone for the guiding principle of this thesis—controlling data distributions with real world data to concretely study out-of-distribution generalization in machine learning.

2.1 The past 350 years

The roots of generalization in machine learning can be traced back through centuries of philosophical inquiry, highlighting humanity's enduring quest to understand how knowledge is generated from experience and how we generalize from known to unknown contexts. This longstanding intellectual pursuit, deeply rooted in epistemology, has traversed the ideas of great thinkers such as John Locke, David Hume, Immanuel Kant, and Bertrand Russell. By examining their contributions, we can better understand how OOD generalization has evolved from abstract philosophical questions into a concrete problem in machine learning.

John Locke, in the 17th century, laid the groundwork for understanding human cognition by positing the central argument in the empiricist view of knowledge acquisition—knowledge is primarily derived from sensory experience. In his seminal book An essay concerning human understanding in 1685, Locke posited that the mind is a tabula rasa, or a blank slate at birth. He posited that all knowledge is derived from sensory experience i.e. the data provided during experience. For Locke, the generalization challenge was to explain how the mind can form general ideas and concepts from specific, fragmented inputs it receives through the senses.

Locke's theory of abstraction provides a solution, suggesting that the mind actively abstracts general ideas from particular instances. In essence, it also suggests a major theme that will become popular in this thesis—the data drives the knowledge an intelligent agent can extract as shown in Chapters /cite. Locke recognized that our perceptions are often shaped by the contexts in which they occur, was skeptical about the limits of empirical knowledge. This foreshadows the challenges faced by modern machine learning models, which often struggle to apply learned knowledge to novel distributions. A layman way to think of it is—how can a model understand a pattern if it is truly never shown in the data. On the flipside, if your data shows a repeating (but not general) pattern, how is it possible for the model to not learn it.

These questions become even more fascinating when thought in the context of our modern understanding of biological intelligence. Firstly, it is unclear when, how, and if biological intelligence can generalize knowledge to new and unseen situations. As humans we often assume that we can. But some results in Chapter 8 prove otherwise. There are many (and often surprising) limits to human generalization.

George Berkeley, another empiricist philosopher, offered a critique of Locke's theory of abstraction. In his works "A Treatise Concerning the Principles of Human Knowledge" (1710) and "Three Dialogues between Hylas and Philonous"

(1713), Berkeley rejected the notion of abstract ideas, arguing that it is impossible to conceive of a general idea that is devoid of all particular features. According to Berkeley, every idea in the mind is particular, and what we call general ideas are merely particular ideas used in a general way. Berkeley's idealism posited that only minds and their ideas exist, and that what we perceive as the physical world is merely a collection of ideas. Thus, generalization, for Berkeley, does not involve abstracting general ideas from particular experiences; instead, it involves recognizing similarities among particular ideas. For instance, the idea of a "tree" is not a general idea but a particular idea that stands for all trees because of the commonalities we recognize. Berkeley's critique highlighted the difficulties involved in explaining generalization, pushing the problem further into the realm of cognitive and perceptual processes. These ideas may be traced as precursors of machine learning research conducted in 1980s-1990s, when papers often proposed that complex tasks like visual recognition involved representing new images in the form of a small number of primal templates which best represent the object invariant to transformations like rotation and scale.

On the other hand, Gottfried Wilhelm Leibniz, in "Monadology" (1714) introduced the principle of sufficient reason, which states that nothing happens without a reason. Leibniz's principle implies that generalization is not only possible but necessary, as the universe operates according to rational principles that can be discovered through reason. His optimism about the power of reason to uncover the truths of the world provided a counterpoint to the skepticism of the empiricists. This work laid the groundwork for the rationalist tradition, which held that the mind, through reason, can grasp universal truths that go beyond mere sensory experience. And so, on one hand we have pioneers like Locke questioning whether we can ever get to general laws from experience, while we have stalwarts like Leibniz necessitating that we must understand general laws if we have any hope of understanding the systems that govern us.

Soon enough, David Hume, another towering figure in empiricism, advanced this discourse in the 18th century by directly addressing the problem of induction, which is at the very heart of the problem of OOD generalization. To put it lightly, Hume's problem of induction was the real killjoy. He argued that our knowledge is based on patterns we observe, but there is no logical certainty that future observations will follow the same patterns. He argued that there is no logical basis for assuming that the patterns observed in the past will necessarily hold in the future. This same skepticism about inductive reasoning underscores the difficulty of ensuring that models trained on one set of data will perform well on another, fundamentally different set. Any repeated experience in the training dataset may very well be just an artefact of that dataset, as opposed to a general truth about all data in the world, and may very well not hold true for new data drawn from out of the training data distribution.

A good example of this is the evolution of the phrase "black swan". Originally, the term "black swan" was used to denote something that was considered impossible or highly improbable, based on the observation of only white swans in Europe. The sighting of a black swan in Australia in the 17th century drastically altered this perception, proving that the assumption of all swans being white was incorrect. This illustrates the limits of empirical knowledge and how assumptions are often based on the available data, which can be incomplete or biased. Whatever system of knowledge biological systems employs, it seemed fit to conclude black swans are a logical impossibility based on the training data which contained only white ones.

While Hume may appear to be a real killjoy, the questions raised by skeptics are truly profound. While for Newton, the laws of motion distilled from observations of the heavenly bodies were laws set in stone, for a skeptic like Hume, these laws are something that ”can only belong to the mind that considers them”.

In response to such skepticism, philosophers like Immanuel Kant sought the middle ground. In his 1781 book, Critique of Pure Reason, Kant proposed that the mind actively shapes experience through a priori concepts and categories. He posited that innate knowledge (which in modern days we refer to as causality or space-time) is necessary to organize sensory input. For Kant, these categories are not derived from experience, but are preconditions for the human experience. Thus, generalization is grounded in the very structure of the human mind, which imposes order on the chaotic influx of sensory data.

Thus, Kant’s notion of a priori knowledge suggests that humans possess inherent structures that allow for the generalization of knowledge beyond direct experience. In machine learning, this concept parallels the development of models that incorporate prior knowledge or inductive biases to better handle OOD generalization. However, while Kant’s ideas provide a theoretical basis for generalization, it remains completely unclear what these a priori biases should be. A wide range of works in computer science have proposed such biases, but the problems still stand. An implementation of such structures in artificial systems remains an ongoing challenge, and the very idea that they are the way forward remains widely contested.

In an approach alternate to Kant's, John Stuart Mill in "A System of Logic" (1843) developed a theory of inductive reasoning that aimed to provide a foundation for scientific knowledge. Mill argued that while inductive reasoning could not provide absolute certainty, it could yield probable knowledge by identifying patterns and regularities in experience. Mill's methods of induction, such as the method of agreement and the method of difference, sought to establish causal relationships based on observed correlations. While acknowledging the limitations of induction highlighted by Hume, Mill believed that systematic observation and experimentation could provide a reliable basis for generalization. His methods aimed to establish causal relationships based on observed correlations, laying the groundwork for the scientific method. In many ways, this theory would explain the modern day usage of the phrase "Black swan", which evolved to mean a low probability event as opposed to an impossible event after the spotting of black swans by dutch philosophers in western australia.

Building on these works, Charles Sanders Peirce introduced a pragmatic approach in his essays on pragmatism, which is perhaps closest to the ideology at the heart of modern applied machine learning. Peirce argued that the meaning of a concept is grounded in its practical consequences. For Peirce, generalization beyond data is not about abstracting universal truths but about forming hypotheses that can guide action and inquiry, much like the modern day practitioner or machine learning. He introduced the idea of abduction, or inference to the best explanation, as a method for generating hypotheses based on observed data. Abduction, unlike deduction or induction, involves forming a plausible hypothesis that can explain the observed facts. Peirce's pragmatic approach highlights the dynamic and provisional nature of generalization, emphasizing that our generalizations are always subject to revision based on new evidence.

The 20th century saw further refinement of these ideas, with figures like Bertrand Russell and Gottlob Frege emphasizing the role of logic and mathematics in understanding the world. Russell's work highlighted the importance of formal systems in ensuring consistency and rigor in reasoning, which is directly relevant to the development of machine learning algorithms. In his "Begriffsschrift" (1879) and later works, Frege developed a formal system of logic that aimed to capture the principles of valid reasoning. Frege's work laid the groundwork for the logicist program, which sought to show that mathematics could be reduced to logic. In many ways, these works and those of their contemporaries have led us to where we stand today, where Artificial Intelligence—learning knowledge which would be akin to human is a discipline driven by formalisms and mathematics.

However, relating these philosophical insights to out-of-distribution (OOD) generalization provides a deeper understanding of the challenges and opportunities in developing robust AI systems. Each philosophical perspective offers a lens through which to view the complexities of training models that can generalize well to new, unseen data distributions. At the heart of it, these theories highlight that knowledge is highly dependent on the data used to learn it. Thus, understanding the data and clearly defining it is of paramount importance. In the next few sections, the author will provide a clear framework for defining and characterizing the data. The framework will also help us clearly define Out-of-Distribution generalization.

2.2 The present: Formalizing data and the knowledge learned

Out-of-distribution (OOD) generalization is a crucial aspect of machine learning that affects various domains, including supervised, self-supervised, and reinforcement learning. However, for the sake of clarity and to provide concrete examples, we will primarily ground our exposition in the context of supervised learning. This section introduces the mathematical and terminological framework used throughout the thesis. This section is heavily inspired by recent amazing work by Martin Arjovsky .

2.2.1 The Dataset and Data Distributions

A dataset in the supervised learning setup is represented by the dataset space. This is typically denoted as cartesian product of the input space and the output space. Thus, $\mathcal{D} : \mathcal{X} \times \mathcal{Y}$.

Here, $\mathcal{X}$ typically denotes the vector space representation of input data. For instance, pictures in the form of pixels, sentences in the form of tokenized vectors, and audio samples in the form of spectograms (among others). We use $\mathcal{Y}$ to denote the space of labels. The nature of $\mathcal{Y}$ depends on the specific model being trained, for classification this would be a discrete vector, and for regression it would be continuous.

A critical consideration in characterizing the dataset is to formulate it as a sample that has been sampled from the joint space $\mathcal{D}$. This formulation allows us to account for the fact that not all input-output samples are equally likely in our dataset source. For instance, consider a dataset consisting of emails. Here, $\mathcal{X}$ would denote all possible emails that could be constructed. Assuming the language is englishe, this space consists of all possible bodies of texts that could be constructed in the english language.

Each dataset is a specific sample from this space. Needless to say, the probability of sampling a dataset of completely non-grammatical emails is pretty low. However, this formulation can be much more helpful than just rooting out non-sensical datasets. Depending on the source of the dataset, the probability of sampling a particular sample x can differ significantly. For instance, a dataset sourced from personal emails is much more likely to contain emails with the subject "Save the date! You are invited to our wedding.", compared to a dataset constructed from workplace emails.

This idea can be mathematically defined in the data distribution—the joint distribution $P(x, y)$ of sampling a particular pair (x, y) from the data space $\mathcal{D}$. This framework us to concretely define:

- Joint Distribution: The likelihood of encountering a specific input-output pair (x, y) is modeled using a joint distribution P. This distribution captures which data points are likely to be encountered during training and testing.

- Distribution Shifts: A common issue in machine learning is that the training data distribution may not reflect the true underlying distribution of data in the real world. In principle, this could be modeled by assuming two different data distributions for the training data and the real world test data. Consider three such datasets sampled to follow distributions P_1, P_2, P_3. If the KL-divergence between P_1, P_2 is less than that between P_1, P_3 we can claim that the shift between Datasets 1 and 2 is lesser than that between Datasets 1 and 3. This is crucial for designing algorithms that are robust to distribution shifts.

It is important to note that in practice this approach quickly becomes intractable. It is exceptionally hard to clearly define the data distriubtion P in real world applications, which makes it incredibly hard to identify or clearly quantify the distribution shift. We will take a deeper look into this phenomenon later in this chapter. But for now, let's turn to how this formulation can be useful to help evaluate models.

2.2.2 Incorporating the data into the evaluation pipeline: Loss Functions and ERM

At the heart of machine learning we always assume there exists a true underlying relationship between the input and output we are trying to learn. This is often referred to as the ground-truth function and denoted by f. This function maps inputs to ground-truth labels, $f : \mathcal{X} \to \hat{\mathcal{Y}}$. The goal of supervised learning is to estimate the ground-truth f using the data. For this, we first start by defining the space of possible functions we will be looking in to find the best match for the ground-truth. Imagine, you have a sock in your hand, and you're trying to find it's best approximating match to make the pair. First, you must define the space of all possible socks you will search for, say your sock drawer. Mathematically we denote to this space as $\mathcal{H}$, and refer to it as the hypothesis space. Let's denote the function from $\mathcal{H}$ which best approximates f as $\hat{f}$.

Perfect, now given our dataset, of course we want to learn a model ($\hat{f}$) that would perform really well on all data possible. Unfortunately, we start with some bad news. Work has long shown that if we want to perform well on all possible data, i.e., we make no assumptions on what data we want to perform well on it spells disaster. This result was proved in the infamous No Free Lunch Theorems, which roughly translate to saying that one can find data points which will perform arbitrarily bad (i.e., as bas as you'd like) given a model.

So, we start with a data distribution P, and want the best possible approximation $\hat{f}$ for data is known to be sampled from this distribution. We see that we must first define what makes an approximation good before we can start looking for the best approximation. Turning to the literature, this has been a long standing problem in learning from data. Going by several names over the eras, but perhaps best known as the model-selection problem. Over the years, several works have posited a definition for the best model.

However, one that has become seminal was proposed by David Wolpert in his

1992 work cite, which we refer to as Empirical Risk Minimization. In practice, we start by defining a non-negative loss function: $L : \hat{\mathcal{Y}} \times \mathcal{Y} \rightarrow \mathbb{R}^{+}$ measures the difference between a predicted label $\hat{y} \in \hat{\mathcal{Y}}$ and a ground-truth label $y \in \mathcal{Y}$.

Given P, we assume the dataset sampled from this distribution contains n i.i.d. data points denoted by $(x_1, y_1), ..., (x_n, y_n)$. Then, we can first define the risk associated with the chosen approximation f^* as the expectation of the loss function w.r.t. the data distribution:

$$R(f^*) = \mathbb{E}[L(f^*(x), y)] = \int L(f^*(x), y)\, dP(x, y). \tag{2.1}$$

This risk is an estimate of the expected loss we will incur if we chose f^* as our model. Then, ERM defines the best model (approximation) simply as the function in $\mathcal{H}$ which would minimize this risk:

$$\hat{f} = \arg\min_{f^* \in \mathcal{H}} R(f^*). \tag{2.2}$$

Now, returning to the problem we earlier alluded to—it is virtually impossible to know the distribution P in real world settings. And thus, the expectation can never really be computed. Instead, we make two major assupmtions. Firstly, all points are i.i.d., and secondly, a finite summation over a finite dataset will be a decent approximation to this integrat. These assumptions reduce correspond to the "Empirical" in ERM, resulting in:

$$\hat{f} = \arg\min_{f^* \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} L(f^*(x_i), y_i) \tag{2.3}$$

In the context of out-of-distribution generalization, the idea is that we have two different data distributions—P_1 and P_2. Usually, the training data (T) and in-distribution test data (t_{in}) are sampled from one, while the out-of-distribution test data (t_{out}) is sampled from the other:

$$T \sim P_1, t_{in} \sim P_1, t_{out} \sim P_2 \tag{2.4}$$

2.2.3 Controlled distribution-shifts in real-world data.

Hopefully, the previous section made it clear to the reader that the real challenge in studying generalization in the real world is to clearly define these data distributions and the shifts between them mathematically. The approach to enable this been central to this entire thesis, and this section is dedicated to discussing how we achieved this in the real world.

As alluded earier, defining data distributions can be exceptionally challenging in the real world. To quote a question my advisor Gabriel Kreiman once asked me—"Even if we wanted to, how could we ever describe the distribution of all possible images of dogs in the real world¿'. Certainly, that is a daunting task. But the central idea in this thesis is to by-pass this problem by drawing inspiration from the same approach taken by Charles Darwin, when studying the underlying principles driving the evolution species.

In the wake of this problem, research in this field has split into two major themes. On one hand, we have theoretical analysis with simple, controlled distribution shifts . While these are truly insightful, the theoretical bounds derived with known distributions are hard to extend to the real world—what does the inability to generalize from data sampled from $\mathcal{N}(0,1)$ to $\mathcal{N}(0.5,1.5)$ tell us about the ability to generalize to unseen 3D viewpoints?

On the other hand, researchers have built large-scale OOD generalization datasets with little control and no formal understanding of the distribution shift. While these datasets are rich in complexity, it is hard to build an understanding of the underlying generalization capabilities due to the lack of control over distribution shifts.

To achieve this, we rely on a path that strays between these two directions—we first define data distributions in terms of scene attributes, and then use computer graphics to convert these scene attributes into rendered image datasets. Thus, we used computer graphics for controlled analysis of generalization with complex data which is similar to real world datasets used in practice. We have constructed several datasets using this approach, which will be presented in the chapters that follow.

Darwin famously said "there is no obvious reason why the principles which have acted so efficiently under domestication should not have acted under nature". This statement underscored his belief in the universality of evolutionary principles, and inspired him to study the evolution of isolated, individual traits through controlled breeding of species in domestication. Naturalists at the time strongly believed otherwise, but Darwin argued that the principles governing this artificial selection—where humans deliberately choose which traits to propagate—are not fundamentally different from the natural processes shaping wild species. This insight was pivotal, as it suggested that the mechanisms of variation and selection observed in domesticated organisms could also apply to natural populations.

This analogy provided a robust framework for understanding how species evolve and adapt in the wild, based on principles that were already well-documented in domesticated varieties. By bridging the gap between artificial and natural selection, Darwin laid the groundwork for the modern understanding of evolution, emphasizing that the same evolutionary forces shaping domesticated plants and animals also sculpt the diversity of life in nature.

In a similar vein, the bedrock of this thesis has been to design datasets where the data distributions can be clearly defined in order to study specific distribution shifts in isolation. This approach has allowed us to investigate the impact of out-of-distribution generalization across real-world factors including lighting, viewpoints, materials, shapes, gravity, size, and relative spatial organization, among others. With controlled and isolated distribution shifts in these attributes, we have studied when and how artificial and biological intelligence generalizes beyond the training data distribution.

Part II

Benchmarking generalization in AI

There is no obvious reason why the principles which have acted so efficiently under domestication should not have acted under nature.

The Origin of Species, Charles Darwin

3

When and How do CNNs generalize to Out-Of-Distribution data?

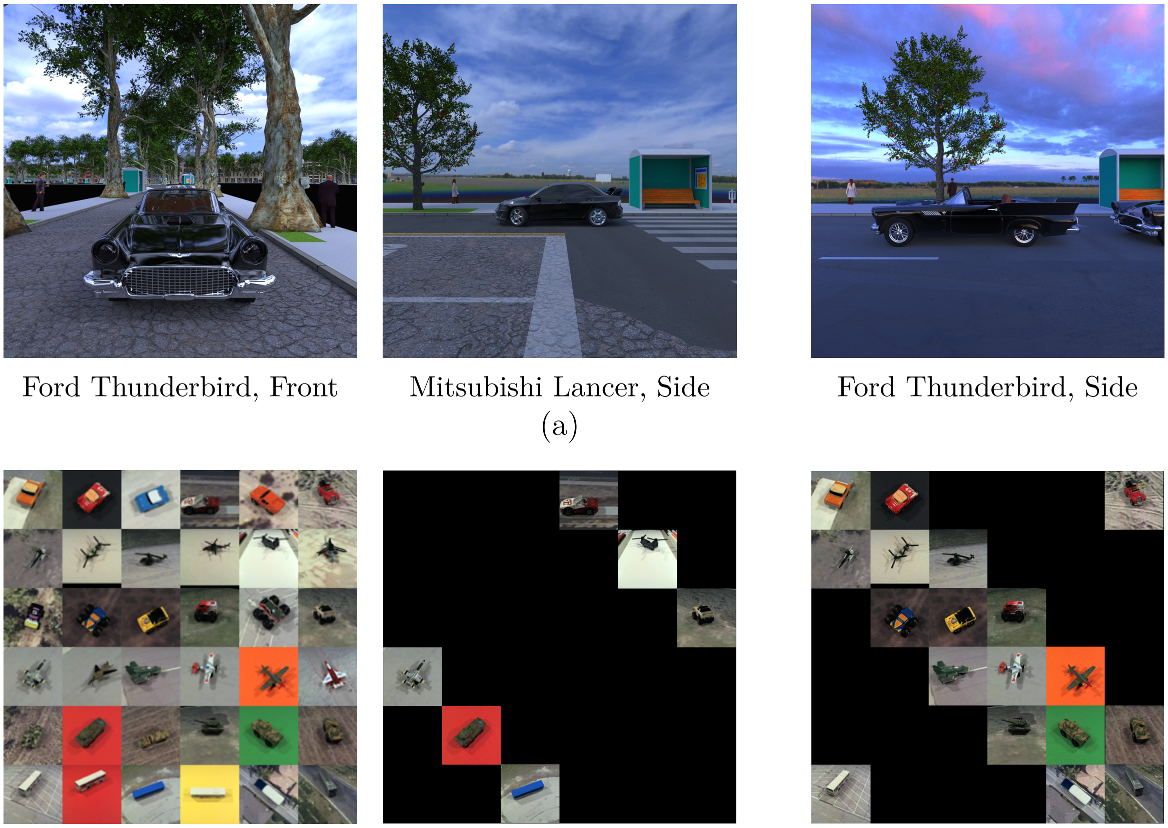
3.1 Introduction

The combination of object recognition and viewpoint estimation is essential for effective visual understanding. In recent years, convolutional neural networks (CNNs) have offered state-of-the-art solutions for both these fundamental tasks[162,361,171,356,252,104,247,9]. However, recent works also suggest that CNNs have a hard time generalizing to combinations of object categories and viewpoints not seen during training, i.e., out-of-distribution (OOD) generalization is a challenge. For object recognition, works have shown CNNs struggling to generalize across spatial transformations like 2D rotation and translation[105,20,352], and non-canonical 3D views[14,24]. For viewpoint estimation, previous works propose learning category specific models[252,382] or feed class predictions as input to the model[413,249], as generalizing to novel categories is a challenging task.

It remains unclear when and how CNNs may generalize to OOD category-viewpoint combinations. Fig. 3.1a presents a motivating example: would a network trained on examples of a Ford Thunderbird seen only from the front, and a Mitsubishi Lancer seen only from the side generalize to predict car model (category) and viewpoint for a Thunderbird shown from the side? If so, what underlying mechanisms enable such OOD generalization?

In this paper, we investigate the impact of two key factors (data diversity and architectural choices) on the capability of generalizing to OOD combinations, and the neural mechanisms that facilitate such generalization. Concretely, we introduce the following discoveries:

1. Data diversity significantly improves OOD performance, but degrades in-distribution performance: We investigate the role of data diversity by varying the number of in-distribution category-viewpoint combinations, keeping dataset size constant. We find that data diversity matters sig-



Ford Thunderbird, Front | Mitsubishi Lancer, Side | Ford Thunderbird, Side

(a)

iLab-2M dataset (b) | OOD Combinations (held-out) (c) | 50% in-distribution Combinations (d)

Figure 3.1: Category-Viewpoint datasets. (a) Our new Biased-Cars dataset: Can a network shown only the Ford Thunderbird from front and the Mitsubishi Lancer from side generalize to classify the category and viewpoint for a Thunderbird seen from the side? (b) iLab-2M dataset[39]: Each cell represents a unique category-viewpoint combination (categories vary between rows, viewpoints between columns) with multiple object instances per category and backgrounds. (c) Held-out test set of category-viewpoint combinations. Same held-out test set is used to evaluate networks trained with different number of in-distribution combinations. (d) Biased training set with 50% of category-viewpoint combinations. Number of categories and viewpoints selected is always equal.

nificantly. For a constant dataset size, increasing data diversity makes the task more challenging, as reflected in the deteriorating in-distribution performance. Yet, increasing data diversity substantially improves performance on OOD combinations.

2. Separate architectures significantly outperform shared ones on OOD combinations unlike in-distribution: We also analyze the performance of different architectures in the multi-task setting of simultaneous category and viewpoint classification, i.e., learning category and viewpoint in Shared or in Separate (no layers shared) architectures. Our results reveal that Separate architectures generalize substantially better to OOD combinations compared to Shared architectures. Also, this trend is in stark contrast with the trend for in-distribution combinations, where Shared architectures perform marginally better. Thus, the belief that Shared architectures outperform Separate ones when tasks are synergistic should be revisited[52], as their relative performance strongly depends on whether the test sample is in-distribution or OOD.

3. Neural specialization facilitates generalization to OOD combinations: Existing works suggest that OOD generalization is facilitated by selective and invariant representations[123,310,129,8]. However, this has not been previously demonstrated for deep learning, and does not extend to simultaneous category and viewpoint classification. To address this, we propose the neural mechanism of specialization—the emergence of two types of neurons, one driving OOD generalization for category, and the other for viewpoint. This corresponds to neurons selective to a category and invariant to viewpoint, and vice versa. We show that the CNN generalization behavior trends correlates with the degree of specialization of the neurons.

These results are consistent across multiple CNNs and datasets including the natural image dataset iLab-2M[39], variations of MNIST[214] extended with position and scale, and a challenging new dataset of car model recognition and viewpoint estimation—the Biased-Cars dataset, which we introduce in this paper. This dataset consists of 15K photo-realistic rendered images of several car models at different positions, scales and viewpoints, and under various illumination, background, clutter and occlusion conditions. With this, we hope to provide a first milestone in understanding the underlying mechanisms which enable OOD generalization in Multi-Task Learning for category and viewpoint classification.

3.2 Datasets for simultaneous category-viewpoint classification

Most existing datasets with category and viewpoint labels[412,47,24,257] present two major challenges - (i) lack of control over the distribution of categories and viewpoints, or (ii) small size. Thus, we present our results on the following datasets which do not suffer from these challenges:

iLab-2M dataset: iLab-2M[39] is a large scale (two million images), natural image dataset with 3D variations in viewpoint and multiple object instances for each category (Fig.3.1b). The dataset was created by placing toy objects on a turntable and photographing them from six different azimuth viewpoints, each at five different zenith angles (total 30). From the original dataset, we chose a subset of six object categories - Bus, Car, Helicopter, Monster Truck, Plane, and Tank. In Fig. 3.1b, each row represents images from one category, and each column images from one azimuth angle. All networks are trained to predict one of six category and viewpoint (azimuth) labels each.

MNIST-Position and MNIST-Scale: Inspired by the MNIST-Rotation dataset[212] which adds rotation to MNIST[214] images, we created two more variants by adding viewpoint in the form of position or scale. MNIST-Position was created by placing MNIST images into one of nine possible locations in an empty 3-by-3 grid. For MNIST-Scale we resized images to one of nine possible sizes followed by zero-padding. Images of the digit 9 were left out in both these datasets, ensuring nine categories and nine viewpoints classes (total of 81 category-viewpoint combinations). Sample images are available in the supplement S1.1.

Biased-Cars dataset: Building on other multi-view car datasets for viewpoint estimation[203,283], we introduce a challenging new dataset for simultaneous object category and viewpoint classification—the Biased-Cars dataset. Our dataset features photo-realistic outdoor scene data with fine control over scene clutter (trees, street furniture, and pedestrians), car colors, object occlusions, diverse backgrounds (building/road textures) and lighting conditions (sky maps). Biased-Cars consists of 15K images of five different car models seen from viewpoints varying between 0-90 degrees of azimuth, and 0-50 degrees of zenith across multiple scales. Our dataset offers two main advantages: (a) complete control over the joint distribution of categories, viewpoints, and other scene parameters, and (b) unlike most existing synthetic city datasets[299,47,100] we use physically based rendering for greater photo-realism, which has been shown to help networks transfer to natural image data significantly better[441,146]. Sample images are shown in Fig. 3.1a. As in[412,96], we choose to focus on azimuth prediction. The azimuth is divided into five bins of 18 degrees each, thus ensuring five category (car models) and five viewpoint classes (azimuth bins), for a total of 25 different category-viewpoint combinations.

Additional Datasets: In the supplement we provide results on two additional standard datasets—MNIST-Rotation[212] and the UIUC3D dataset[320]. Note that the UIUC dataset has a skewed joint distribution of category-viewpoint combinations. This makes it difficult to run controlled experiments. However, the experiments which were possible on this dataset confirm that our findings extend to it as well.

For all datasets, networks are trained to classify both category and viewpoint simultaneously without pretraining, and the number of classes for each task is kept equal to ensure equal treatment. More details can be found in supplement S1.2. As shown in the experiments, these datasets are challenging benchmarks for testing generalization, with a huge scope for improvement for state-of-the-art CNNs.

3.3 Factors affecting generalization behaviour

Below we present the two factors we study for their impact on generalization to OOD category-viewpoint combinations - (i) data diversity, and (ii) architectural choices.

Generating train/test splits with desired data diversity. All our ddatasets can be visualized as a square category-viewpoint combinations grid as shown for the iLab dataset in Fig. 3.1b. Each row represents images from one category, and each column a viewpoint, i.e., each cell represents all images from one category-viewpoint combination.

For each dataset, we start by constructing an OOD test split—a set of category-viewpoint combinations are selected and held out from the combinations grid as shown in Fig. 3.1c. We refer to these as the OOD combinations. Images from OOD combinations are never shown to any network during training. These images are only used to evaluate how networks generalize outside the training distribution. For a fair representation of each category and viewpoint, we ensure that every category and viewpoint class occurs exactly once in the OOD combinations, i.e., one cell each per row and column is selected.

Remaining cells in the combinations grid are used to construct multiple training splits with an increasing number of category-viewpoint combinations i.e., data diversity. For each training split, we first sample a set of combinations as shown in Fig. 3.1d, which we call the in-distribution combinations. Then, we build the training data-split by sampling images from these in-distribution combinations. We ensure that every category and viewpoint occurs equally in the in-distribution combinations, i.e., equal number of cells per each row and column. Fig. 3.1d shows the 50% in-distribution training split for the iLab dataset. To ensure that we evaluate the effect of data diversity and not that of data amount, the number of images is kept constant across train splits as the number of in-distribution combinations is increased. Thus, the number of images per combination decreases as the number of in-distribution combinations is increased. Also, note that every network is trained with only one of these training splits at a time, i.e., data diversity is kept constant during training.

Architectural choices. One central question addressed in this paper is the impact of architectural choices on the capability to generalize to OOD category-viewpoint combinations. While many separate models have been proposed for object recognition and viewpoint estimation[122,381], recent years have seen a growing a trend of multi-task learning inspired architectures which suggest that recognition models can benefit from an understanding of object viewpoint, and vice versa[286,443,252,356,217]. These architectures often learn a shared representation for both tasks, followed by task specific branches[356,443,133].
Here, we investigate the impact of learning shared representations on the network's capability to generalize to OOD category-viewpoint combinations i.e., to extrapolate in the multi-task setting of simultaneous category and viewpoint classification. For this, we defined two types of backbone agnostic architectures—the Shared and the Separate architectures. Fig. 3.2 depicts these architectures for a ResNet-18 backbone[162]. In the Shared case, all convolutional blocks are shared between tasks, followed by task-specific fully connected layers, while there are no layers shared between tasks in the Separate architecture. We also investigated 3 additional Split architectures which represent a gradual transition from Separate to Shared ResNet-18: the Split-1, Split-2, and Split-3 architectures. These were constructed by branching ResNet-18 after 1, 2, and 3 convo-

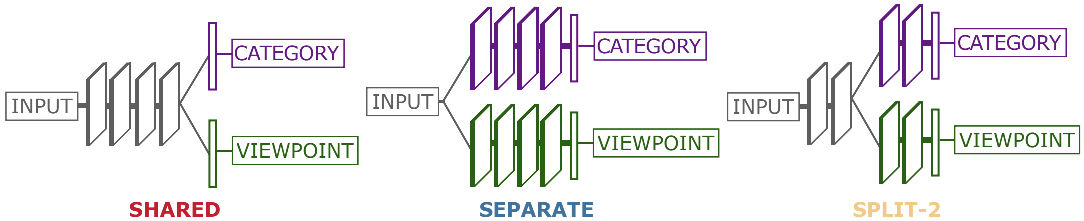


Figure 3.2: Architectures for Category Recognition and Viewpoint Estimation. Shared, Separate and Split-2 architectures for ResNet-18. In the Shared architecture, all layers until the last convolutional block are shared between tasks, followed by task specific fully connected branches. In the Separate architecture, each task is trained in a separate network with no layer sharing. Split-2 presents a middle ground. These architectures are designed similarly for backbones other than ResNet-18.

lutional blocks as shown in Fig. 3.2. Note that splitting at a layer leads to doubling of the number of neurons in that layer. In our experiments, we show that this increase in width does not provide an advantage.

3.4 Generalization through selectivity and invariance

Selectivity and invariance of neurons have long been hypothesized to facilitate generalization in both biological and artificial neural networks[42,310,129,8,291,276,300,315]. Neurons are commonly interpreted as image feature detectors, such that the neuron's activity is high only when certain features are present in the image[431,342,444,28,278]. We refer to this property as selectivity to an image feature. Selectivity alone, however, is not sufficient to generalize to OOD category-viewpoint combinations. For example, a neuron may be selective to features relevant to a category, but only so for a subset of all the viewpoints. Generalization is facilitated by selective neurons that are also invariant to nuisance features. For instance, in Fig. 3.1a, neurons that are selective to the Ford Thunderbird and invariant to viewpoint would have very similar activity for the Ford Thunderbird on in-distribution and OOD viewpoints, thus enabling generalization to category recognition. Similarly, generalization to viewpoint estimation can be enabled by neurons selective to viewpoint and invariant to category.

Here, we present our implementation for quantifying the amount of selectivity and invariance of an individual neuron. Let N be the number of categories or viewpoints in the dataset. We represent the activations for a neuron across all category-viewpoint combinations as an $N \times N$ activations grid, as shown in Fig. 3.5a. Each cell in this activations grid represents the average activation of a neuron for images from one category-viewpoint combination, with rows and columns representing average activations for all images from a single category (e.g., Ford Thunderbird) and a viewpoint (e.g., front), respectively. These activations are normalized to lie between 0 and 1 (see supplement S2.1). For neuron k, we define a_{ij}^k as the entry in the activations grid for row (category) i and column (viewpoint) j. Below we introduce the evaluation of a neuron's selectivity score with respect to category and invariance score with respect to viewpoint. Viewpoint selectivity score and category invariance score can be derived analogously.

Selectivity score. We first identify the category that the neuron is activated for the most on average, i.e., the category which has the maximum sum across the rows in Fig. 3.5a. We call this category the neuron's preferred category, and denote it as $i^{\star k}$, such that $i^{\star k} = \arg\max_i \sum_j a_{ij}^k$. The selectivity score compares the average activity for the preferred category (denoted as $\hat{a}^k$) with the average activity of the remaining categories ($\bar{a}^k$). Let S_c^k be the selectivity score with respect to category, which we define as is usual in the literature (e.g., [263,446]) with the following expression:

$$S_c^k = \frac{\hat{a}^k - \bar{a}^k}{\hat{a}^k + \bar{a}^k}, \quad \text{where } \hat{a}^k = \frac{1}{N}\sum_j a^k_{i^{\star k}j}, \;\; \bar{a}^k = \frac{\sum_{i \neq i^{\star k}} \sum_j a^k_{ij}}{N(N-1)}. \tag{3.1}$$

Observe that S_c^k is a value between 0 and 1, and higher values of S_c^k indicate that the neuron is more active for the preferred category as compared to the rest. Selectivity with respect to viewpoint, denoted as S_v^k, can be derived analogously by swapping indices (i, j).

Invariance score. A neuron's invariance to viewpoint captures the range of its average activity for the preferred category as the viewpoint (nuisance parameter) is changed. Let I_v^k be the invariance score with respect to viewpoint which we define as the difference between the highest and lowest activity across all viewpoints for the preferred category, i.e.,

$$I_v^k = 1 - \left(\max_j \; a^k_{i^{\star k}j} - \min_j \; a^k_{i^{\star k}j} \right), \tag{3.2}$$

where the range is subtracted from 1 to have the invariance score equal to 1 when there is maximal invariance. Invariance with respect to category, denoted I_c^k, can be derived analogously.

Specialization score. Generalization to category recognition may be facilitated by neurons selective to category and invariant to viewpoint. Similarly, viewpoint selective and category invariant neurons can help generalize well to viewpoint estimation. This reveals a tension when category and viewpoint are learned together, as a neuron which is selective to category, cannot be invariant to category. The same is true for viewpoint. One way this contradiction may be resolved is the emergence of two types of neurons—category selective and viewpoint invariant, and vice versa. We refer to this as specialization. This hypothesis is well-aligned with the findings in[425], which showed the emergence of groups of neurons contributing exclusively to single tasks. Thus, in the context of category recognition and viewpoint estimation, we hypothesize that neurons become selective to either category or viewpoint at later layers as the relevant image features for these tasks are disjoint (the category of an object cannot predict its viewpoint, and vice-versa).

To classify neuron k as a category or viewpoint neuron, we compare its selectivity for both category and viewpoint (S_c^k and S_v^k). If S_c^k is greater than S_v^k, then neuron k is a category neuron, otherwise, it is a viewpoint neuron. Since generalization capability relies on both invariance and selectivity, we introduce a new metric for a neuron, the specialization score denoted as Γ^k, which is the geometric mean of its selectivity and invariance scores, i.e.,

$$\Gamma^k = \begin{cases} \sqrt{S_c^k I_v^k} & \text{if } S_c^k > S_v^k \quad \text{(category neuron)} \\ \sqrt{S_v^k I_c^k} & \text{if } S_c^k \leq S_v^k \quad \text{(viewpoint neuron)} \end{cases} . \tag{3.3}$$

Below, we present results that show that the specialization score is highly indicative of a network's performance on OOD combinations.

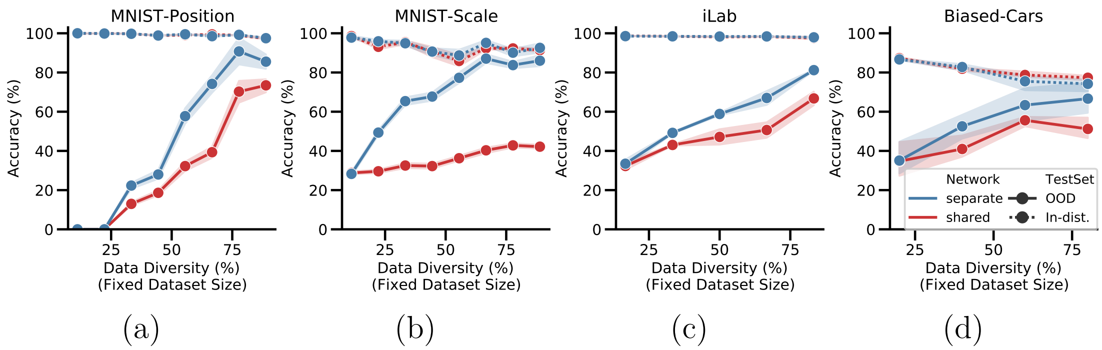


Figure 3.3: Generalization performance for Shared and Separate ResNet-18 as in-distribution combinations are increased for all datasets. The geometric mean of category recognition accuracy and viewpoint estimation accuracy is reported along with confidence intervals (95%) (a) MNIST-Position dataset. (b) MNIST-Scale dataset. (c) iLab dataset. (d) Biased-Cars dataset.

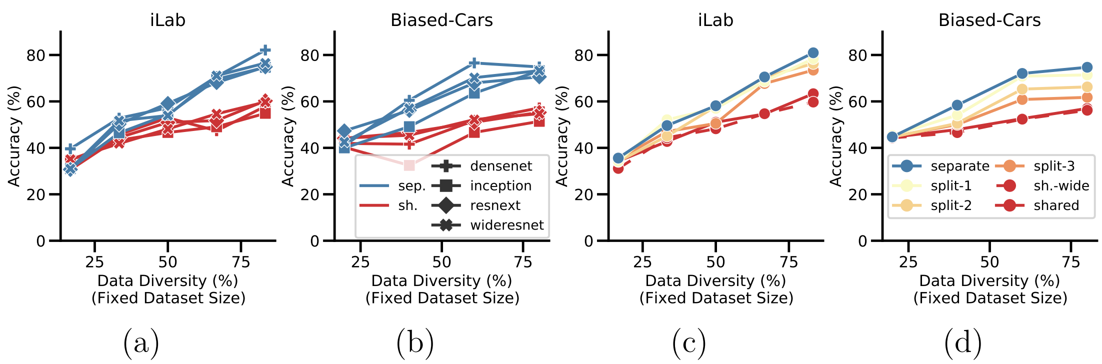


Figure 3.4: Generalization performance for different architectures and backbones as in-distribution combinations are increased for iLab and Biased-Cars datasets. The geometric mean between category recognition accuracy and viewpoint recognition accuracy is reported for OOD combinations as number of in-distribution combinations is increased. (a) and (b) Accuracy of Separate and Shared for backbones other than ResNet-18, for iLab and Biased-Cars datasets, respectively. (c) and (d) Accuracy of ResNet-18 Separate, Shared and different Split architectures made by splitting at different blocks of the network, for iLab and Biased-Cars datasets, respectively.

When do CNNs generalize to OOD combinations?

Below, we summarize our findings from evaluating Separate and Shared architectures when tested on unseen images from in-distribution and OOD category-viewpoint combinations. See supplement S3 for experimental details.

For fixed dataset size, data diversity enables better OOD generalization, but deteriorates in-distribution performance. Fig. 3.3 presents the geometric mean of category and viewpoint classification accuracy for Separate and Shared architectures with the ResNet-18 backbone, for all datasets. These experiments were repeated three times, and here we present the mean performance with confidence intervals. For fixed dataset size, increasing in-distribution combinations makes the task more challenging as images with each category and viewpoint become more diverse, leading to some drop in accuracy on in-distribution combinations. In contrast, both architectures show a significant improvement of their performance on images from OOD combinations, as data diversity increases. We ensured that this result can not be attributed to having closer viewpoint angles between in-distribution and OOD combinations as data diversity is increased (supplement S4.1). CNNs do not theoretically guarantee viewpoint invariance[291], but our result provides reassurance that CNNs can become robust to OOD category-viewpoint combinations as long as they are shown enough diversity during training. Taken together, these results suggest an inherent trade-off between getting better on in-distribution combinations and extrapolating to OOD combinations, which is impacted by training data diversity. Also, these results add to a growing body of works investigating the trade-offs inherent to multi-task learning[353,336].

Even though the geometric mean of category and viewpoint classification increases consistently with increased in-distribution combinations, individual accuracy for these tasks does not always increase consistently (see supplement S4.2). We attribute this to the randomness in the selection of in-distribution and OOD combinations. Furthermore, the relative accuracy of the two tasks varies depending on the dataset, and no task is consistently harder than the other across all datasets.

Separate architectures generalize significantly better than Shared ones in OOD combinations, unlike in-distribution. A striking finding that emerged from our analysis is the contrast in the trends of the in-distribution and OOD performance. While both architectures perform well on new images from in-distribution combinations, Separate architectures outperform Shared ones by a very large margin on OOD combinations. For the ResNet-18 backbone, this result can be seen consistently across all 4 datasets as shown in Fig. 3.3. Supplement S4.2 shows that Separate also outperforms Shared for category and viewpoint classification individually. Note that previous works have shown that Shared architectures are superior for synergistic tasks, as networks can share features among tasks. These works test on the same combinations as seen during training (in-distribution), and when we do so, we also observe that Shared architectures perform same or slightly better than Separate ones (Fig. 3.3 dashed lines). Thus, our results reveal that the relative performance between Shared and Separate depends not only on the synergy between tasks, but also whether the evaluation is in-distribution or OOD.

We extended our analysis to Separate and Shared architectures with different backbones, namely ResNeXt[419], WideResNet[429], Inception v3[361] and the

DenseNet[171], as shown in Fig. 3.4a and b. As can be seen, Separate architectures outperform Shared ones by a large margin for all backbones, which confirms that this result is not backbone specific. Investigating further, we experiment with Split architectures, and as can be seen in Fig. 3.4c and d, there is a consistent, gradual dip in the performance as we move from the Separate to the Shared architectures. Thus, generalization to OOD category-viewpoint combinations is best achieved by learning both tasks separately, with a consistent decrease in generalization as more parameter sharing is enforced.

To make sure that Separate architectures do not perform better due to the added number of neurons, we made the Shared-Wide architecture by doubling the neurons in each layer of the Shared ResNet-18 network. As Fig. 3.4c and d show, this architecture performs very similarly to the Shared one (see additional results in S4.3). This is in accordance with previous results that show that modern CNNs may improve in performance as the width is increased but to a limited extent[270,53].

In the supplement, we provide a number of additional controls that support the generality of our results. Concretely, we show results for different number of training images (supplement S4.4), viewpoint estimation for 4 new car models and category prediction for new viewpoints (supplement S4.5), and the order in which category and viewpoint are learned (supplement S4.6). We also present results on additional datasets (supplement S4.7) and architectures (supplement S4.8).

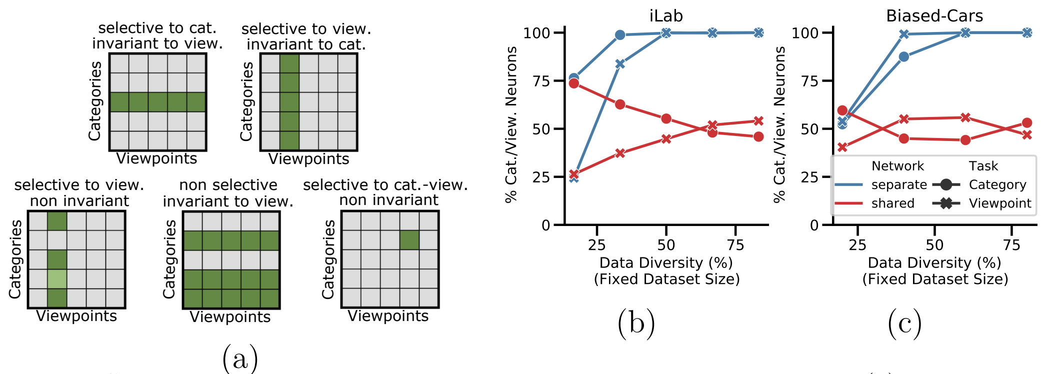


Figure 3.5: Specialization to category recognition and viewpoint estimation. (a) Prototypical activation grids for different types of selective and invariant neurons. (b) and (c) Percentage of neurons after ResNet-18 block-4 that are specialized to category and viewpoint, for iLab and Biased-Cars datasets, respectively. ResNet-18 Separate and Shared networks are evaluated; for Separate, only the task-relevant neurons for each branch are displayed.

3.5 How do CNNs generalize to OOD combinations?

We now analyze the role of specialized (i.e., selective and invariant) neurons in driving generalization to OOD category-viewpoint combinations.

Specialization score correlates with generalization to OOD category-viewpoint. We first investigate the emergence of category and viewpoint neurons in the final convolutional layer of the networks. Fig. 3.5b and c show the percentage of neurons of each type in Shared and Separate architectures as in-distribution combinations are increased. As can be seen, all neurons in the category and viewpoint branches of the Separate architecture become specialized to category and viewpoint respectively. But in the Shared case, as the network is expected to simultaneously learn both tasks, both kinds of neurons emerge at a ratio of about 50%. We found that this ratio depends on the relative weight of loss terms for the two tasks. When using a different weight from the optimal in terms of maximum geometric mean accuracy, the 50% ratio of specialized neuron becomes unbalanced. For a small number of in-distribution combinations, the ratio of specialized neurons may also be impacted by the relative difficulty of two tasks, with more neurons becoming specialized for the easier task (see supplement S5.1).

In Fig. 3.6 we present the median of specialization scores across neurons, i.e., the median of Γ^k, in the final convolutional layer for Shared, Split, and Separate architectures across multiple backbones in Biased-Cars dataset (see supplement S5.2 for results in other datasets). These results are presented separately for the category and viewpoint neurons. We show that as in-distribution combinations increase, there is a steady increase in the specialization score for both category and viewpoint neurons, suggesting specialization. These trends mirror the generalization trends, which suggests that specialization facilitates OOD generalization. Invariance and selectivity scores are reported separately in supplement S5.3. We also show that specialization builds up across layers (supplement S5.4) as expected [129,291].

Separate networks facilitate the emergence of specialized neurons. Fig. 3.6 shows that Separate architectures facilitate specialization, while the Shared architecture makes it harder for the neurons to specialize (lower specialization scores). This might be because unlike the Shared architecture, the branches of the Separate architecture are not forced to preserve features relevant to both tasks. Each branch can develop features which are selective to only one task, and invariant to the other. This may facilitate an increase in specialization and thus enable better performance on OOD combinations. Even though the Shared architec-

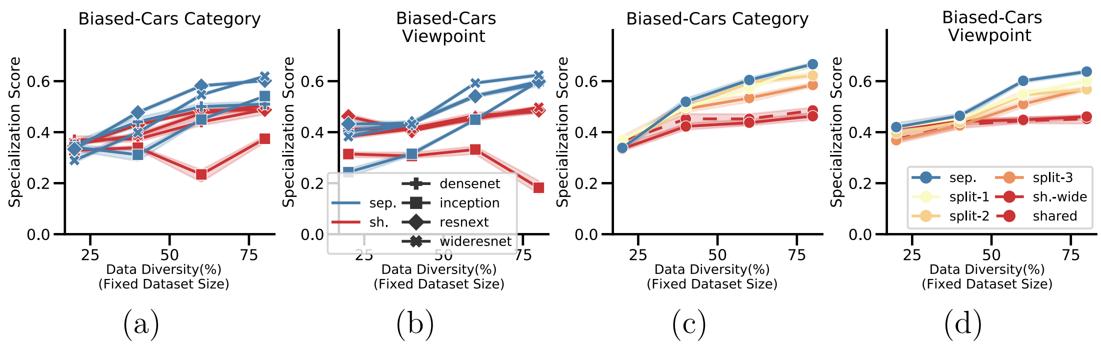


Figure 3.6: Neuron specialization (selectivity to category and invariance to viewpoint, and vice versa) in the Biased-Cars dataset. (a) and (b) Median specialization score of neurons (Γ^k) in Separate and Shared architectures for category and viewpoint classification tasks respectively, for backbones other than ResNet-18. Confidence intervals (95%) displayed in low opacity. (c) and (d) Median specialization score of neurons in ResNet-18 Separate and Shared architectures with splits made at different blocks of the network, for category and viewpoint classification tasks respectively.

ture tries to split into two specialized parts, this specialization is much stronger in the Separate architecture due to already having separate branches.

3.6 Conclusions

We have demonstrated that CNNs generalize better to OOD category-viewpoint combinations as the training data diversity grows, for constant dataset size. We have also shown that networks trained separately for category and viewpoint classification surpass by a large margin a shared network trained on both tasks when tested on OOD combinations. We attribute this to the branches in the Separate architecture not being forced to preserve information about both tasks, which facilitates an increase in specialization, i.e., selectivity to category and invariance to viewpoint, and vice versa. These results are consistent across five CNN backbones and six datasets, one of them introduced in this paper as a controlled yet photo-realistic benchmark for CNN generalization.

We also found that the aforementioned impact of data diversity and Separate architecture are the opposite for in-distribution and OOD combinations—increased data diversity degrades in-distribution performance, and Separate networks perform worse than Shared ones in in-distribution combinations. This highlights that findings from in-distribution analysis do not apply to OOD.

As a first step towards understanding generalization to OOD combinations, our work makes certain assumptions (summarized in the supplement S6) which present interesting directions for future work. These include understanding how generalization is impacted by a larger number of tasks, multiple objects in the image, object symmetries, non-rigid objects, and non-uniform ways of holding-out the test set, among others. Finally, we are intrigued to explore what other factors can help learn selective and invariant neural representations which can generalize better and lead the way towards robust, trustable CNNs.

Data and Code Availability Statement

Source code, demos and data are available on Github at https://github.com/Spandan-Madan/generalization_biased_category_pose (https://zenodo.org/record/5636158#.YZRBKS1h1js). The paper corresponding to this work can be accessed at https://www.nature.com/articles/s42256-021-00437-5.

The past beats inside me like a second heart.

John Banville

4

Emergent representations

4.1 Introduction

Recognizing objects in novel orientations lies at the heart of biological and artificial intelligence, as it is a fundamental capacity to understand the visual world[344,385]. However, it remains as an outstanding question.

In the realm of artificial systems, Deep Neural Networks (DNNs) have re-

a

b

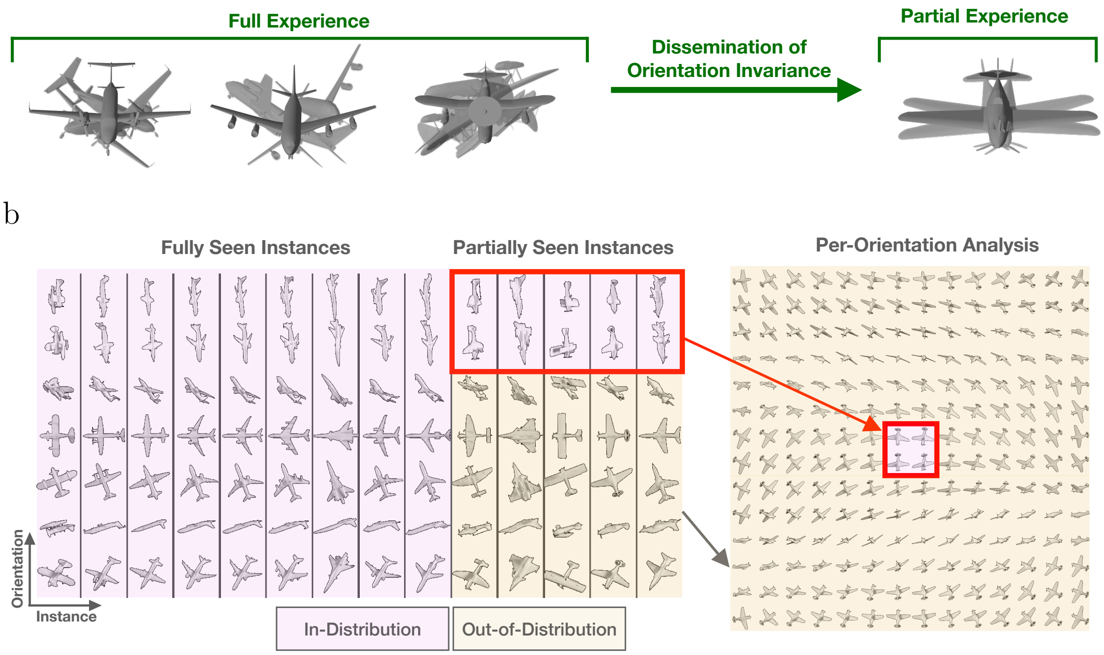


Figure 4.1: Learning paradigm and network's per-orientation accuracy. (a) An intuitive presentation of our experimental paradigm. The network is trained with images of certain airplanes at all orientations, constituting a 'full experience.' The network is also trained with images of a small subset of orientations for other airplanes, constituting a 'partial experience.' The out-of-distribution, or OoD, generalization capacities of the network are evaluated by measuring the classification accuracy for partially-seen airplanes at unseen orientations. Our results suggest that OoD generalization is facilitated by the dissemination of orientation invariance developed for all orientations for the fully-seen airplanes to the OoD orientations of the partially-seen airplanes. (b) Left: The learning paradigm employed in this work. Each column is a sample object instance (here from the airplane dataset) and each row is a sample orientation. The training set includes all orientations for fully-seen instances, and a partial set of orientations (outlined in red) for partially-seen instances (in this example, with the airplanes' nose pointing down). The orientations included in the training set are referred to as in-distribution orientations (pink shading). Orientations of the partially-seen instances that are not included in the training set are referred to as OoD (yellow shading). Right: A visualization of per-orientation-analysis. The set of all orientations can be portrayed as a square, where proximity relationships between orientations are captured in their arrangement in the square. (Further details are provided in Fig.4.2a.) The in-distribution and OoD sets for partially-seen airplanes are represented in their respective shadings.

cently made large strides in learning to recognize objects. However, DNNs' generalization is often limited to the distribution of images used for training, known as in-distribution data, and DNNs' performance tends to deteriorate when confronted with out-of-distribution (OoD) data. Previous studies have shown that DNNs perform poorly when objects are presented in novel orientations, even when learning from large datasets with millions of examples[24,14,241].

An approach to understand the capabilities of DNNs is to leverage the knowledge gained from studying biological intelligence[158,386]. In natural settings, biological intelligent agents observe instances of object categories from diverse orientations. When encountering a new object instance, these agents often demonstrate the capacity to accurately identify the object in different orientations by drawing upon past experiences with similar instances. There has been a extensive investigation dedicated to the exploration of human and mammalian perceptual capabilities in the domain of object recognition in unfamiliar orientations and the neural mechanisms underlying these cognitive abilities. Studies have revealed that recognition accuracy varies across novel orientations, with some orientations exhibiting superior generalization compared to others[235]. Additionally, compelling evidence suggests that neurons responds to their own specific set of object features when these are present in the visual field[92,200]. This neural tuning has been reported to be invariant to a certain degree from the object's orientation[236]. Theoretical frameworks have proposed that such neural invariance to object orientation forms the basis for the ability to recognize objects in novel orientations within biological systems[291].

We study DNNs under conditions akin to the operating regime of biological brains, in which some instances of an object category (e.g., a 'Boeing 777 airliner' is an instance of the 'airplane' category) are seen from all orientations during training (fully-seen instances), while other instances are only seen in a subset of all orientations (partially-seen instances). During test time, we evaluate the generalization performance of the networks by measuring instance classification performance on OoD orientations (i.e., those orientations not included in the training set) of partially-seen instances. This is a simple paradigm, inspired by the paradigm by[430], that facilitates analyzing the impact of several key factors that may influence OoD generalization, such as the number of fully-seen instances and the in-distribution orientations of the partially-seen instances. This paradigm allows us to more precisely characterize performance challenges of DNNs for OoD orientations. Figure 4.1 summarizes the paradigm that we follow in this work.

In this paper, we employ the same analytical tools utilized in the study of biological brains, with the aim of understanding the DNNs' generalization abilities in OoD orientations. Specifically, we investigate questions that have already been investigated within the realm of biological systems, such as whether DNNs exhibit different recognition failures across OoD orientations, and explore whether individual neurons within DNNs display feature tuning for object recognition while also exhibiting invariance to different orientations. We aim at explaining the generalization abilities observed in DNNs.

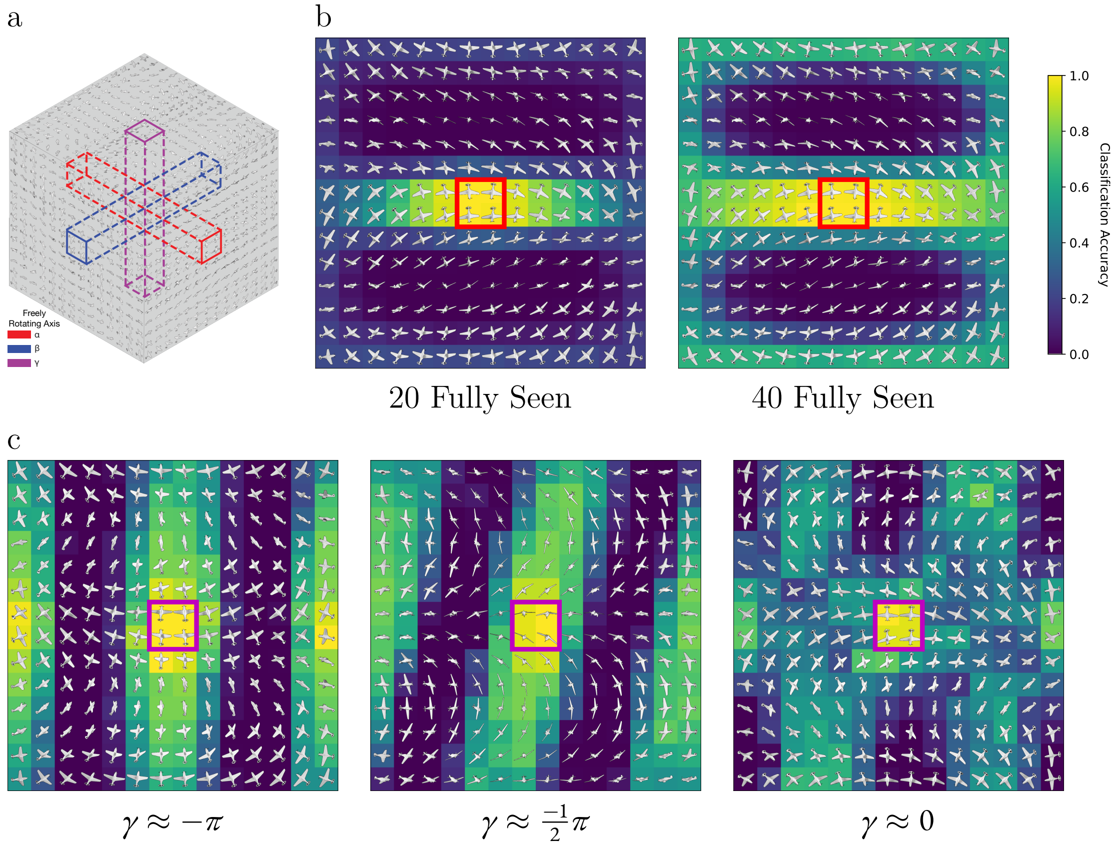


Figure 4.2: Observed generalization patterns in per-orientation accuracy heatmaps. When trained with a combination of fully-seen instances and partially-seen instances, DNNs demonstrate the ability to generalize outside of their training distribution. Generalization behaviors are demonstrated measuring per-orientation accuracy. (a) All orientations can be described by three Euler axes ($\alpha, \beta, \gamma,$) and rotations are periodic around these axes. These properties allow for the visualization of all possible orientations with an orientation cube, shown here. The orientations contained within the colored rectangular prism are those orientations of the partially-seen instances included in training (i.e., are in-distribution). The in-distribution orientations differ depending on the experiment. All other orientations are OoD. (b) Increased network generalization for OoD orientations, with increased instance diversity (i.e., number of fully-seen.) Each cell in the heatmap is the average classification accuracy of the network for a given value of β and γ, across all values of α. Chance level is 0.02 (2%). (c) Different in-distribution parameters affect the generalization behaviors. The generalization patterns for a different span of in-distribution orientations ($-0.25 \leq \alpha \leq 0.1, -0.1 \leq \beta \leq 0.25, -\pi \leq \gamma < \pi$) as outlined by the purple box. In this case, each cell is of a given value for α, β, γ.

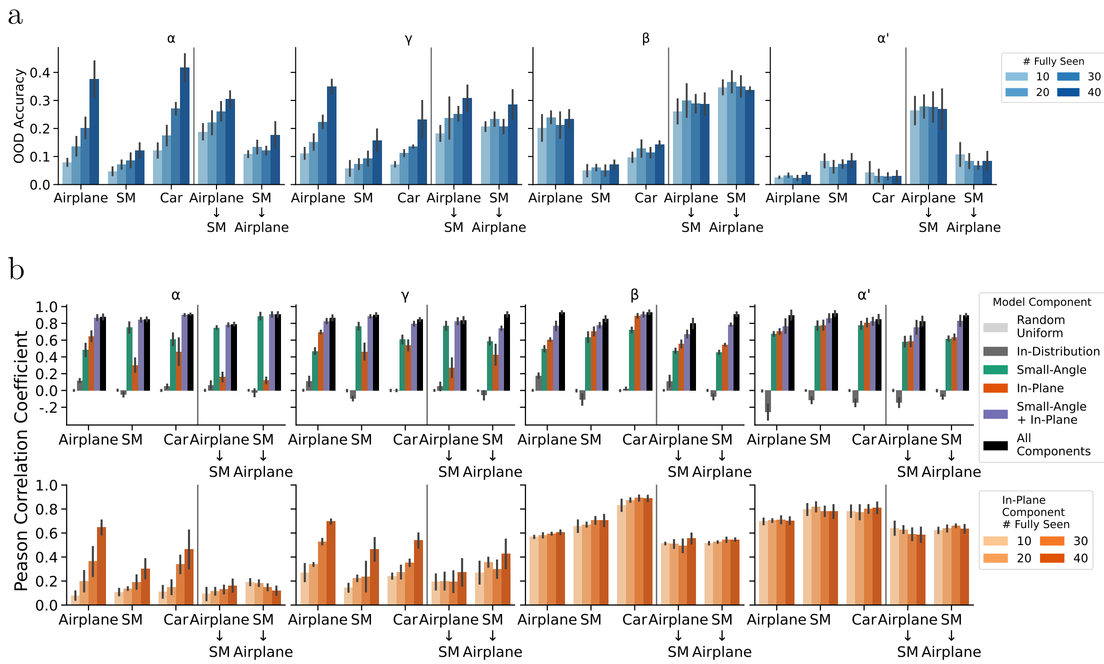


Figure 4.3: Modeling generalization patterns for OoD orientations. The bar plots show several trends related to DNN OoD classification patterns. The trends are measured under the various controls, including in-distribution orientations conditions ($\alpha, \gamma, \beta, \alpha'$) and object category, which is either a single object as in Airplane, SM, Car, or transfer across two categories, when the fully-seen instances are of a different category than the partially-seen instances as in Airplane → SM and vice versa. These transfer cases are visually separated from the other cases. (a) Network generalization for OoD orientations increases with increasing number of fully seen (blue shading.) This trend holds across object category and in-distribution orientations conditions. (b) Top: We introduce a predictive model for OoD orientation generalization (black — "All Components") which is highly predictive of experimental results, with greater than 0.8 Pearson Correlation Coefficient for all experimental controls. (Results are shown for experiments with 40 fully-seen instances.) Null hypothesis predictive models, including "Random Uniform" and "In-Distribution," have very low correlation coefficients. We also ablate our predictive model, including only some sub-components, like only-"Small Angle", only-"In-Plane" or only-"Small Angle + In-Plane." These ablated models have lower correlation coefficients than "All Components," and vary in relation to one another depending on the experimental condition. Bottom: We isolate the predictive power of the only-"In-Plane" component for all experiments with a range of number of fully-seen. The increasing predictive power of the "In-Plane" component correlates with increasing OoD accuracy as the number of fully-seen instances increases. This suggests that generalization to "In-Plane" orientations drives OoD accuracy.

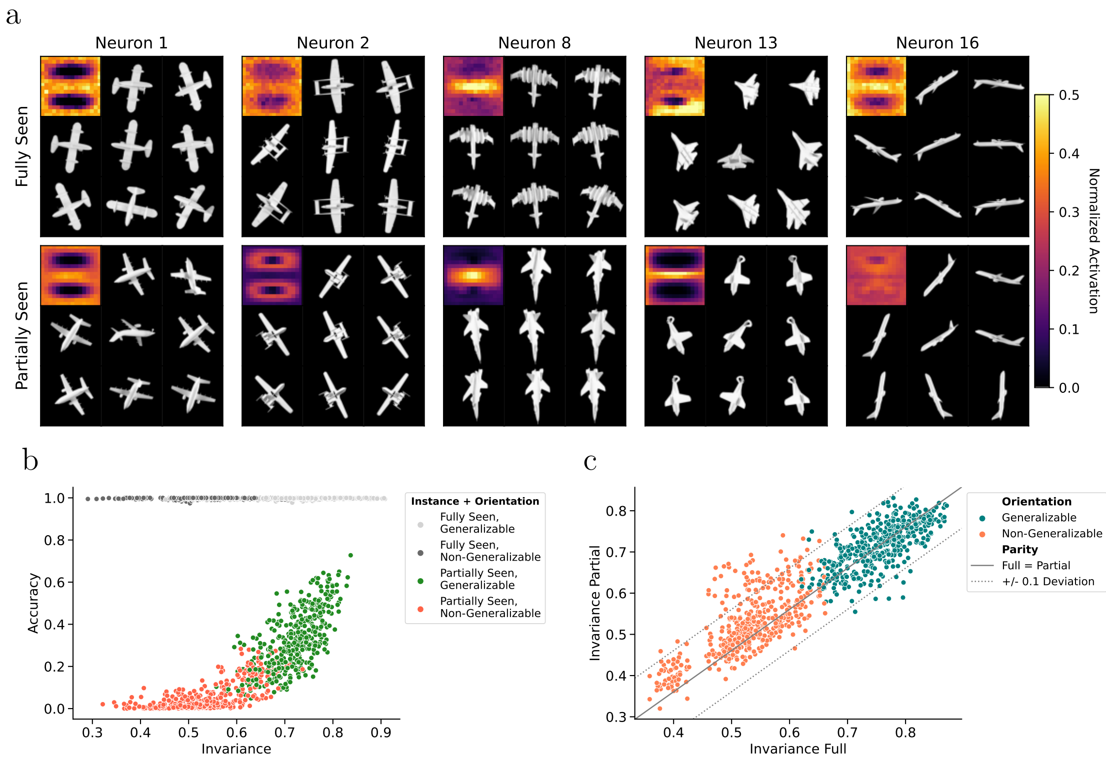


Figure 4.4: Neuronal analysis, Invariance and Dissemination. (a) An intuitive visualization of neural activity. Each square is the response of a single neuron to the airplane instance that most highly activates it portrayed in two ways: 1) the top-8 images that most highly activate the neuron (in no particular order), 2) the heatmap of the per-orientation normalized neural activity for the airplane instance. Neurons tend to exhibit patterns of activation related to the patterns of generalization behavior (Figs. 4.2b for example,) and are invariant to a range of orientations that respect the partitioning of OoD orientations. The patterns of activation are similar between the fully-seen instance that most highly activates the neuron and the partially-seen instance that most highly activates it due to shared visual, part and semantic features between these instances. Several randomly sampled penultimate layer neurons, arranged into columns, demonstrate that these findings apply to many neurons. (b) Averaging the activations in the partitioned regions ('seed', generalizable, non-generalizable) and computing the invariances (defined here: Eq.4.6) between 'seed' and OoD regions captures overall generalization in the network. Plotting the generalization metrics against accuracy for those regions demonstrates a clear correlation between increasing invariance and increasing OoD classification accuracy (i.e., partially-seen instances in OoD orientations.) (c) Plotting fully-seen invariance against partially-seen invariance also yields a tight correlation, suggesting that dissemination of invariance from fully-seen to partially-seen instances enables increasing generalization in OoD orientations of partially-seen instances.

4.2 Results

Overview. In exploration of the generalization capabilities of DNNs to novel orientations, our computational experiments show that the OoD orientation space is divided between generalizable and non-generalizable orientations, in terms of the networks behavior. We find this division to be governed by a set of rules which determine a partitioning of the orientation space, given a 'seed' of orientations seen by the DNNs at train time. Among several partitioning rules, we identify a rather intuitive one: small 3D perturbations around seen orientations will be included in the highly-generalizable partition. We find other rules to be more surprising, including that the highly-generalizable partition also consists of shape and silhouette preserving rotations, such as in-plane (i.e., 2D) rotations and flips along axes of symmetry of the seen orientations. Our analysis shows that the predicted network behavior induced by these partitioning rules is highly correlated with the measured network behavior under a variety of training regimes. We further explore the DNNs internal representations and identify neuronal mechanisms in the network, that allow the dissemination of orientation-invariance from familiar objects to novel objects and orientations.

Per-orientation accuracy heatmaps. A detailed inspection of the network's generalization capability for OoD orientations is brought forward by introducing per-orientation accuracy heatmaps. In brief, the continuous space of object orientations, represented by Euler angles, is discretized into cubelets (Fig. 4.2a). For each cubelet the network's performance is evaluated in terms of the classification accuracy $\Psi(\theta)$, where θ is an orientation of interest. Accuracy heatmaps are 2D projections across a specified dimension of the full accuracy orientation cube (Fig. 4.2b,c; Methods). These heatmaps reveal a reproducible pattern of generalization in the form of increased classification accuracy for OoD (i.e., novel) orientations. For example, Figure 4.2b shows that for 'seed' orientations at the center of the heatmap (red box), the network (in this experiment - ResNet18[162]; see Methods) yields the highest accuracy (brightest cubelets) for adjacent orientations around the 'seed', depicting small 3D perturbations of the 'seed' orientations. Further inspection of the heatmap reveals other orientations, in this example, brighter cubelets forming the figure '8' (stretching 'seed' sideways and along the heatmap's boundaries, enclosing two darker 'holes'), for which the network performs better than for the rest of the OoD orientations. These orientation mainly depict in-plane rotations of the 'seed' orientations.

When considering the average classification accuracy across all OoD orientations, our experiments reproduce previous results. In particular, we can reliably quantify the effect of data diversity on the OoD generalization, as the amount of training examples is kept constant with our learning paradigm. Figure 4.3a clearly shows an increase in OoD accuracy as data diversity (i.e., the number of fully-seen instances) increases, under various conditions, including different 'seed' orientations, different image datasets and across datasets. The accuracy heatmaps provide a complementary means of assessment to the overall average accuracy measure, depicting the generalization patterns and indicating which orientations account for the network increased performance (e.g., Fig. 4.2b).

The patterns of increased accuracy depict a partitioning of the orientation space, which reappears for various 'seed' orientations (Fig. 4.2c), various sizes of the training set and different object categories (e.g., Airplane, Car, Shepard & Metzler (SM) objects[334]), as shown in several experiments (Supplementary Figs. Supp.16, Supp.17; Methods).

Modeling generalization patterns. We hypothesize a set of rules which govern the partitioning of the orientation space into generalizable and non-generalizable orientations. To evaluate this hypothesis we formulate a model of the partitioning rules, which can be used to predict the OoD generalization patterns of the network, given a 'seed' of in-distribution orientations. Briefly, the model, denoted by $f_{\mathbf{w}}(\theta)$ has three components: $A(\theta)$, which captures small angle rotations around θ; $E(\theta)$, which captures in-plane (2D) rotations; $S(\theta)$, which captures object silhouette projections at the orientation θ (see details in Methods). We evaluate the model's performance by measuring the Pearson correlation coefficient ρ between the accuracy of the networks as measured in our experiments and as predicted by the model, i.e., $\rho(\Psi(\theta), f_{\mathbf{w}}(\theta))$. Figure 4.3b shows the predictive power of the model and its components in experiment with different 'seed' orientations and several object categories. The model's component $A(\theta)$ ('small angle' rotations), is the best predictor for the network's OoD behaviour, for highly articulated objects such as the SM objects. On the other hand, the model's component $E(\theta)$ ('in-plane' rotations), is a better predictor for non-articulated objects with inherent symmetries. Further analysis of this component illustrates how generalization to 'in-plane' rotations emerges with the increase in data diversity.

We conducted a large series of experiments under various settings, including different ‘seed’ orientation distributions, various amounts of training examples, different object scales, object categories with different levels of symmetry, image datasets and DNN architectures (see Methods, Supplementary Figs. Supp.16, Supp.17, Supp.18, Supp.19, Supp.20). In all experiments our model highly predicts the network’s behaviour, indicating that indeed the networks generalization patterns for OoD orientations follow the model’s partitioning rules. (See Supp.21 for reports of average accuracy in generalizable and non-generalizable orientations.) This is true even across categories, when the ‘seed’ is taken from one category (e.g., SM) and the ‘fully-seen’ instances are taken from another (e.g., Airplane).

Individual unit neuronal analysis. In search of how generalization and dissemination emerge in DNN’s we turn to analyze the neurons’ activation in the trained networks. We focus on neurons in the penultimate layer of the network, which are attuned to the highest level features in the input stimuli, but reflect a consolidated representation of the entire network for inferring the downstream task (classification in our simulations).

Figure 4.4a illustrates activation of individual neurons for stimuli of fully-seen and partially-seen instances. Each group of images depict input stimuli of a particular instance for which a particular neuron has the highest activation, along with the neuron’s per-orientation activation heatmap for the instance. The patterns seen in the neurons’ activation heatmaps resemble the partitioning patterns of the accuracy heatmaps shown in Figure 4.2b. Some neurons exhibit similar activation patterns for both fully-seen and partially-seen instances, while others do not.

A quantifiable measure of these neuronal responses can help understanding how generalization occurs in the network, in particular for the partially-seen instances observed only at the 'seed' orientations during training. Hence, any generalization to OoD orientations for the partially-seen instances must stem from the 'seed' orientations. We define an activation invariance score in the range $[0, 1]$ (Eq. 4.6) between sets of orientations, in particular between the 'seed' orientations and OoD generalizable orientations or non-generalizable orientations. The invariance score yields higher values when a neuron fires for both sets of orientations, and lower values when it fires only for one set (see details in Methods). We expect that generalization, reflected by the accuracy level, will correlate with the invariance score.

Figure 4.4b depicts a scatter plot of the invariance score against the classification accuracy. Each dot represents an instance at an orientation set, and the coloring indicates the respective instance set (fully-seen or partially-seen) and orientation set (generalizable or non-generalizable). As expected, there is a clear correlation between increasing levels of classification accuracy and increasing invariance score for the partially-seen instances. Furthermore, the plot shows a clear partition between generalizable and non-generalizable orientations with respect to the invariance score, where significantly higher invariance scores are measured for the generalizable orientations.

For fully-seen instances (Fig. 4.4b gray dots), all orientations are in-distribution, including the ‘seed’, generalizable and non-generalizable. Therefore, the network easily achieves accuracy at ceiling levels regardless of the neuronal invariance score. Nevertheless, the fully-seen instances exhibit the same invariance partitioning between generalizable and non-generalizable orientations as the partially-seen instances.

Figure 4.4c depicts a direct comparison between the invariance score of the fully-seen and partially-seen sets for both the generalizable and non-generalizable orientations. The partition between generalizable and non-generalizable orientations is exhibited again — the non-generalizable invariances are in the bottom left corner, while the generalizable invariances are in the top right corner. Each point in this plot represents the joint invariance of fully-seen and partially-seen instances at a given orientation. The plot clearly shows a tight correlation between the invariance scores of the fully-seen and partially-seen instances, as most of the points lie within a band roughly 0.1 units away from the line of parity, $x = y$. This correlation suggests that an increase in the invariance score of the network at a set of orientations for the fully-seen instances will be disseminated to the partially-seen instances (see also Supp.22).

4.3 Discussion

Our results support the hypothesis that the network disseminates orientation-invariance of fully-seen instances to partially-seen instances using brain-like mechanism like those reported by[236,291]. Neurons are feature detectors and some of the features that neurons are tuned are features shared between fully-seen and partially-seen instances (Fig.4.4a), and therefore the same neuron helps detecting features for several objects. Features are seen at multiple orientations in the fully-seen object instances, which enables neurons to develop orientation invariance. Since features detected by the neurons are shared among fully-seen and partially-seen instances, invariance that develop for fully-seen instances are gained “for free” for partially-seen instances in the same orientations. Note that orientation-invariance is learned through the fully-seen instances, as increasing the number of fully-seen instances (while keeping the number of training examples constant) results in an increase of the DNN capability of disseminating orientation-invariance obtained from familiar objects. Our results elucidate the intricate neural processes involved in object recognition and also underscores the critical role of individual neuron, feature-based representations for OoD object recognition.

Dissemination of orientation-invariance has been observed for orientations that appear like 2D rotations (in-plane) of in-distribution orientations. In some cases, when the network relies on the object instance's silhouette for recognition, the in-distribution orientations also include orientations that have the same silhouette as the seen orientations. For non-generalizable orientations, the network has not developed orientation-invariance with respect to the seed orientations (demonstrated by the lower invariance score in our results). It is worth noting that despite the absence of orientation-invariance, the network is still able to recognize fully-seen instances in such non-generalizable orientations. This is due to the fact that these orientations fall within the training distribution and the network has learned to associate them with their corresponding object instances. However, in the case of non-generalizable orientations, the dissemination of orientation-invariance is not feasible. This is even the case when neurons are tuned to features shared with partially-seen instances, as they do not exhibit orientation-invariance for these non-generalizable orientations and the training process does not provide any information to establish associations with the corresponding object instances. These findings reveal discernible patterns in the successes and failures of DNNs across diverse orientations. Such patterns can be effectively characterized and explained through the analysis of neural activity. This underscores the potential for more comprehensive analyses of DNNs that transcend the conventional approach of solely focusing on average accuracy.

A key question arising from our results is to explain why DNNs disseminate orientation-invariance only to in-plane orientations. All object instances are distinguishable at all orientations, as evidenced by the high in-distribution accuracy achieved by the DNNs. Therefore the lack of orientation-invariance for such non-generalizable orientations is an outcome of the DNN's learning process. We speculate that this may be because orientations that are not in-plane are affected by self-occlusion, which poses a particular challenge for DNNs. Various efforts have been made to enhance DNNs' generalization capabilities to OoD orientations including leveraging preconceived components for DNNs, such as 3D models of objects[16] or sophisticated sensing approaches like omnidirectional imaging[72]. However, these approaches rely on ad-hoc approaches tailored to specific objects and do not address the fundamental limitations of the DNN learning process in recognizing objects in OoD orientations. Instead, novel network architectures that extend the emergent orientation-invariance inherent within networks might allow for further gains of OoD generalization. Biological agents may overcome the difficulties associated with recognizing OoD orientations by leveraging the temporal dimension to associate orientations and learn invariant representations[312,183]. The mechanisms that utilize temporal association may hold fundamental significance, given that they have access to a plentiful source of training data that does not rely on external guidance and task specific labels. This data is readily available prior to any visual task and has the potential to contribute to the emergence of orientation-invariant representations beyond in-plane orientations.

Previous studies have extensively compared the behavioural and electrophysiological aspects of brains and DNNs[422]. However, a direct comparison between these systems alone has limitations in providing insights into the underlying mechanisms of object recognition in DNNs. This is due to the possibility that while certain fundamental mechanisms may be shared across these systems, the manifestation of these fundamental mechanisms can differ at the behavioral and electrophysiological levels. Our study has provided compelling evidence of brain-like neural mechanisms in DNNs that facilitate object recognition in novel orientations, even though these mechanisms are manifested differently than in biological systems. For instance, while humans and primates can recognize objects in orientations that are not simply 2D rotations, this capability is not fully replicated in DNNs. Thus, we can conclude that the neural mechanisms that have been observed to govern recognition in biological systems largely apply to DNNs, albeit with distinct manifestations across these systems. It will be interesting to follow this line of investigation across biological and artificial systems to envision a general theory to explain emergent mechanisms in both brains and machines.

4.4 Methods

4.4.1 Per-Orientation Accuracy Visualization

Previous works have typically reported average performance over all orientations. In contrast, we evaluate the network's performance for each orientation across the entire range of orientations. To express an orientation of an object instance we use $\theta := (\alpha, \beta, \gamma)$, the Euler angles with respect to the orthogonal axes of a reference coordinate system $\mathbb{R}^3$ [126], with the convention that α and γ are bounded within 2π radians, and β is bounded within π radians. We define $\Psi(\theta) \in [0, 1]$ to be the network's average classification accuracy at an orientation $\theta = (\alpha, \beta, \gamma)$ over either the fully-seen or partially-seen instances.

To facilitate intuition of Ψ we introduce a visual representation of this function. Since orientations are continuous values and are related spatially we map the range of bounded values of orientations (α, β, γ) onto a Cartesian coordinate system, resulting in a cube—the basis of our visualization. We discretize the continuous space of orientations into cubelets, which are sub-cubes with a width of $\frac{1}{\#\text{Cubelets}}$ of the full range of each respective angle. This approach preserves local behavior in aggregate analysis. In addition, we outline the range of orientations which are in-distribution for the partially-seen instances — the rest are OoD orientations. To illustrate the object orientation at a given cubelet, we sample one representative image and overlay it onto the heatmap at the location of the cubelet.

See Fig. 4.2a, which shows this visual representation scheme, and Figs. 4.2b and 4.2c for examples.

4.4.2 Model of DNN Per-Orientation Generalization

In Results we briefly introduce the hypothesis that DNNs are capable of generalizing to orientations which are small angle rotations of the in-distribution orientations images and to orientations that are in-plane relative to the in-distribution images. In this section we formalize this model.

Recall that we defined $f_{\mathbf{w}}(\theta)$ as the predictive model for generalization per each orientation. To measure the goodness of our prediction, we employ the Pearson correlation coefficient to measure how closely our model correlates with DNN recognition accuracy, $\Psi(\theta)$. We choose this metric because it normalizes data with respect to amplitude and variance, and therefore measures patterns of behavior across θ and relative to other θ, rather than the exact performance for every θ.

Our model $f_{\mathbf{w}}(\theta)$ is composed by three components ($A(\theta)$, $E(\theta)$ and $S(\theta)$), which we introduce next. These three components easily lend themselves to formalization with Euler's rotation theorem[126]. The theorem states that any rotation can be uniquely described by a single axis, represented by a unit vector $\hat{\mathbf{e}} \in \mathbb{R}^3$, and an angle of rotation, denoted as $\varphi \in [0, \pi]$ around the axis $\hat{\mathbf{e}}$. We employ this representation to describe the rotation between an arbitrary orientation of interest, θ, and an orientation in the set of in-distribution, denoted $\theta_s \in \mathbf{\Omega}_s$. We use $\hat{\mathbf{e}}_{\theta,\theta_s}$ and $\varphi_{\theta,\theta_s}$ to denote the unit vector (axis) and the angle of this rotation, respectively.

Component 1: Small Angle Rotation, $A(\theta)$. The first component of the model captures orientations that are small angle rotations from the orientations in the training distribution. Visually similar orientations are those that are arrived at by small rotations from in-distribution orientations, or small $\varphi_{\theta,\theta_s}$. We therefore define the first component $A(\theta)$ as

$$A(\theta) := \max_{\theta_s \in \Omega_s} \left| 1 - \frac{\varphi_{\theta,\theta_s}}{\pi} \right| \in [0, 1]. \tag{4.1}$$

The $\max_{\theta_s \in \Omega_s}$ operator chooses the in-distribution orientation that is closest to θ of interest.

Component 2: In-plane Rotation, $E(\theta)$. The second component of the model captures orientations which appear as in-plane rotations of in-distribution images. Let $\mathbf{c} \in \mathbb{R}^3$ be the unit vector representing the camera axis. In-plane rotations are those for which the axis of rotation is parallel to the camera axis. Thus, an orientation appear as an in-plane rotations of an in-distribution images when $\mathbf{c} \in \mathbb{R}^3$ and $\hat{\mathbf{e}}_{\theta,\theta_s} \in \mathbb{R}^3$ (i.e., the vector of object instance rotation) are parallel. Taking their standard inner product yields the proximity to being parallel, which is therefore the degree to which the rotation is in-plane.

Thus, we define the second component $E(\theta)$ as follows:

$$E(\theta) := \max_{\theta_s \in \Omega_s} \left| \mathbf{c}^\top \hat{\mathbf{e}}_{\theta,\theta_s} \right| \in [0,1], \tag{4.2}$$

where $\mathbf{c}^\top$ denotes the transpose of $\mathbf{c}$.

Component 3: Silhouette, $S_A(\theta)$, $S_E(\theta)$. The third component of the model captures orientations which project object silhouettes onto the camera that are similar to the silhouettes of the object when in-distribution — for example, the airplane when viewed from above, and the silhouette being the airplane viewed from below. These orientations are defined as a π radians rotation around the γ axis, which results in a silhouette orientation. We transform all the in-distribution orientations, $\mathbf{\Omega}_s$, in this way, and we call these silhouette in-distribution orientations $\mathbf{\Omega}_{\hat{s}}$. We then compute $S_A(\theta)$ and $S_E(\theta)$, substituting $\mathbf{\Omega}_{\hat{s}}$ for $\mathbf{\Omega}_s$ in $A(\theta)$ and $E(\theta)$ respectively.

Nonlinearities. The components described above capture a general trend, but do not match the range of values given by a 0-100% accuracy metric. We therefore fit the components with a logistic function. The ‘S’-like shape of the logistic function allows for the highest and lowest values of $E(\theta)$, $A(\theta)$, $S_A(\theta)$ and $S_E(\theta)$ to be close to the highest and lowest values of $\Psi(\theta)$. In addition, it allows for a smooth transition between these highest and lowest values. Most importantly, the simplicity of the logistic function allows for fitting while preserving the interpretability of the model components, ensuring that the models remains related to small angle, in-plane and silhouette rotations. We employ the following logistic function:

$$\sigma(x;(a,b,c)) = \frac{1}{1+e^{b(-x^c+a)}}, \tag{4.3}$$

where $x \in \{E(\theta), A(\theta), S_A(\theta), S_E(\theta)\}$. a and b translate and scale the values of the predictive components and c spreads out saturated values of the component.

Fitting the Model with Gradient Descent. The model combines four components $A(\theta)$, $E(\theta)$, $S_A(\theta)$ and $S_E(\theta)$ by taking the sum of their respective values after applying the logistic function σ:

$$\begin{aligned} & f_{\mathbf{w}}(\theta) \\ := & \sigma(A(\theta);\mathbf{w}_A) + \sigma(E(\theta);\mathbf{w}_E) + \\ & \sigma(S_A(\theta);\mathbf{w}_{SA}) + \sigma(S_E(\theta);\mathbf{w}_{SE}), \end{aligned} \tag{4.4}$$

where $\mathbf{w}$ represents the parameters of the logistic functions i.e., $\mathbf{w} = (\mathbf{w}_A, \mathbf{w}_E, \mathbf{w}_{SA}, \mathbf{w}_{SE})$. The logistic fitting function is differentiable, and $f_{\mathbf{w}}(\theta)$, the linear combination of these logistic functions, is also differentiable. Further, the Pearson correlation coefficient is also differentiable. Therefore we employ gradient descent to fit $\mathbf{w}$ with the Pearson correlation coefficient as the cost function.

4.4.3 Neural Analysis

In Results we discussed our findings that OoD generalization in the network is allowed for by dissemination of orientation invariance from fully-seen instances to partially-seen instances. In this section, we outline the process by which we quantify several different network invariance metrics. We first formalize the notation for neural activations for single orientations and for sets of orientations. We then define the invariance score (Eq. 4.6). Finally, we average together many invariance calculations to arrive at the network invariance metric.

We begin by formalizing our approach to neural activations. In Sec.4.4.1 we introduced $\Psi(\theta)$, the network's average accuracy at a specific orientation. We can similarly define the neural activation at a specific orientation, though we do so with more granularity. Namely, we introduce $\Phi_i^n(\theta)$, which is the average activation of a neuron n from the set of all penultimate-layer neurons N (i.e., $n \in N$) across all images of an object instance i from the set of all object instances I (i.e., $i \in I$) for a given orientation θ. We normalize the activity of each neuron by dividing the activity level of each image by the maximum activity generated by any image. We exclude any neurons with a maximum activation of 0 from further analysis.

Having defined Φ we note that it is useful to perform analysis not on single orientations only, but sets of orientations. We demonstrated that under our experimental conditions, orientations can be partitioned into coherent subsets — in-distribution and OoD orientations. Further, the OoD orientations can be partitioned into generalizable orientations, i.e., those OoD orientations that the network can generalize to, and non-generalizable orientations. We refer to the in-distribution, generalizable and non-generalizable orientation sets as *InD*, G and $\neg G$ respectively. The determination of membership of the generalizable and non-generalizable orientation sets is as follows: We compute 10% of the maximum value of $f_{\mathbf{w}}(\theta)$, the predictive model, in the experiment with 40 fully-seen instances. All orientations for which f is greater than the 10% threshold are considered generalizable, otherwise they are considered non-generalizable. We can now compute the average activation of a set or orientations. For example, the average activation for a given neuron n and object instance i of the generalizable orientations is defined in the following way:

$$\bar{\Phi}_i^n(G) = \frac{1}{|G|} \sum_{\theta \in G} \Phi_i^n(\theta). \tag{4.5}$$

The same may be computed for in-distribution and non-generalizable orientations.

To determine how dissemination occurs in the network, we calculate the degree of similarity in a neuron's response to a given instance across different orientations. Specifically, given a neuron n and instance i, we calculate the similarity between the neuron's response at an orientation pair $\Phi_i^n(\theta_1)$, and $\Phi_i^n(\theta_2)$, or pair of sets of orientations $\bar{\Phi}_i^n(\mathrm{InD})$, $\bar{\Phi}_i^n(G)$ for example. We use ∂, invariance score, as the similarity metric, which is defined (based on previous work[241]) in the following way:

$$\partial(\bar{\Phi}_i^n(InD), \bar{\Phi}_i^n(G)) = 1 - \left| \frac{\bar{\Phi}_i^n(G) - \bar{\Phi}_i^n(InD)}{\bar{\Phi}_i^n(G) + \bar{\Phi}_i^n(InD)} \right|. \tag{4.6}$$

We note that under some conditions, ∂ reports a high, yet trivial, invariance. Namely, if the response of a neuron is low or zero for both elements of the pair, the denominator approaches zero and the invariance becomes large. However in this case the neuron is not responding to anything — any activity is most likely noise. We therefore calculate a threshold of activity for neural response invariances to be considered to contribute to the generalization capability of the network. Otherwise, these invariances are not integrated into the overall network invariance metric. The threshold, τ, is the 95th percentile of activity for all neurons across all images. We employ τ with an indicator function as follows:

$$\begin{aligned} &1(\bar{\Phi}_i^n(InD), \bar{\Phi}_i^n(G)) \\ &:= \begin{cases} 1 & \text{if } \bar{\Phi}_i^n(InD) \geq \tau \wedge \bar{\Phi}_i^n(G) \geq \tau \\ 0 & \text{otherwise} \end{cases}. \end{aligned}$$

Finally, we can compute the overall network generalizable and non-generalizable invariance scores. To do so, we compute a triple average: an average activation over the set of orientations (Eq. 4.5) and averaged over the invariance of all neurons and object instances. We say that the generalizable invariance score is the invariance between the in-distribution orientations and the generalizable orientations determined as follows:

$$\frac{1}{L}\sum_{n\in N}\sum_{i\in I} 1(\bar{\Phi}_i^n(InD), \bar{\Phi}_i^n(G)) \cdot \partial(\bar{\Phi}_i^n(InD), \bar{\Phi}_i^n(G)), \tag{4.7}$$

where L is the quantity of activity pairs above the threshold τ, i.e.,

$$L = \sum_{n\in N}\sum_{i\in I} 1(\bar{\Phi}_i^n(InD), \bar{\Phi}_i^n(G)). \tag{4.8}$$

The definition of the network's non-generalizable invariance score is the same, though $\neg G$ replaces G.

4.4.4 Experimental Controls

Proportion of fully-seen instances. We vary diversity in terms of the number of fully-seen instances N between 10 (20% of the total number of instances) and 40 (80%). The remaining instances are partially-seen. For a fair evaluation of the effect of data diversity, the amount of training examples is kept constant as we vary the data diversity.

Object Categories. We used three categories of objects: Airplanes, Cars and Shepard&Metzler objects. For the airplanes and cars we curated 50 high quality object instances of each category from the ShapeNet[57] database. Both airplanes and cars have clear axes of symmetry, which allow for intuition of how networks generalize to OoD orientations. We therefore also experimented with highly asymmetric objects similar to those tested for 3D mental rotations in[334]

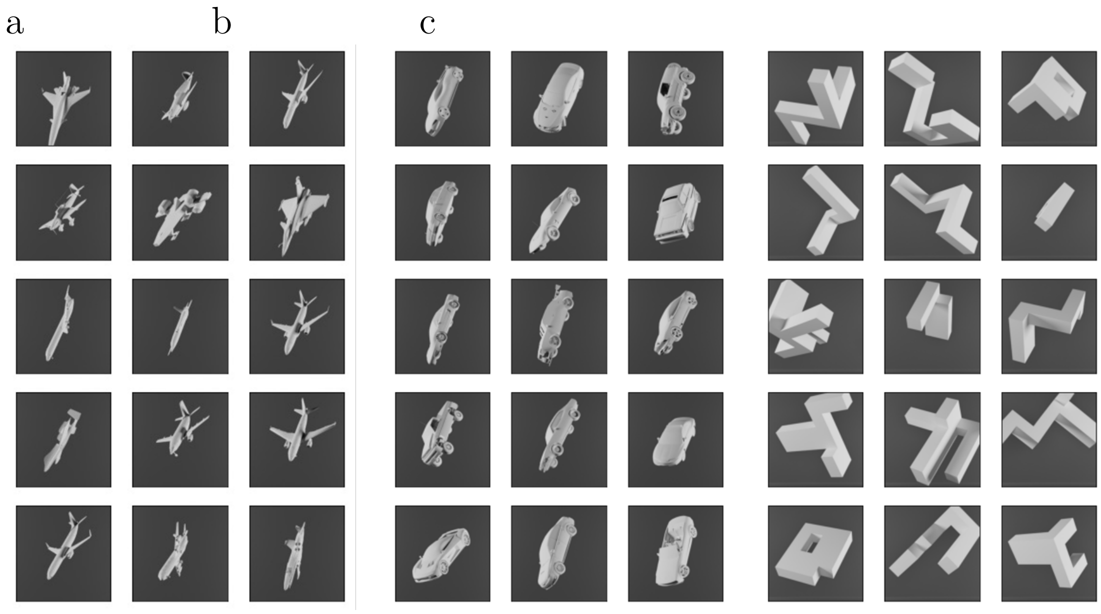


Figure 4.5: Object Datasets. In our experiments we used three object categories: (a) Airplanes, (b) Cars, and (c) Shepard&Metzler objects. The first two were curated from ShapeNet[57] and we procedurally generated the last one. There are 50 instances per object category (e.g., 'Concorde' or 'Spitfire' for the Airplanes). Images were rendered from the 3D models under fixed lighting conditions, and the models were centered and fully contained within the image frame. For fully-seen instances (see Fig. 4.1), orientations were uniformly sampled at random using Euler angles in the range of $-\pi \leq \alpha < \pi, -\frac{\pi}{2} \leq \beta < \frac{\pi}{2}, -\pi \leq \gamma < \pi$. For partially-seen instances, orientations were uniformly sampled from a subset of these ranges.

(which we denote as Shepard&Metzler objects; Fig. 4.5).

DNN Architectures. We used ResNet18[162], DenseNet[171] and CORnet[208] in our experiments. The first two were chosen as they are representative feed-forward DNNs. The architecture of CORnet is brain-inspired and includes recurrence at higher layers in addition to convolutions in lower layers.

Repetition. We re-run each experiment five times, each time randomly sampling the specific instances which comprise the fully-seen and partially seen sets.

Hyperparameters for training. We trained the three deep convolutional neural networks using the Adam Optimizer[198] with following learning rates and batch seizes, respectively:

- ResNet18
  - Learning rate: 0.001
  - Batch size: 230
- DenseNet
  - Learning rate: 0.001
  - Batch size: 64
- CORnet
  - Learning rate: 0.0001
  - Batch size: 128

Batch sizes were chosen to be as large as possible while still fitting the model, the batch of images and forward-pass computations in memory. Learning rates were chosen from $10^x, x \in \{-1, -2, -3, -4, -5\}$ to be as large as possible while ensuring that OoD generalization remained stable. Each network was trained for 10 epochs. After this point in-distribution performance was stabilized at 100% and OoD performance reached an asymptote.

Dataset Size. Each dataset is 200k images, 4k image for each of the 50 object instances. A training epoch iterates through every image in the dataset once.

Hardware details. Experiments were run with one CPU, 25GB of memory and on several generations of Nvidia GPUs with a minimum of 11GB of memory.

Science can amuse and fascinate us all, but it is engineering that changes the world."

Isaac Asimov

5

Enforcing invariant representations to improve generalization.

5.1 Introduction

It has been observed that data bias caused by data collection process, i.e., systematic difference between all possible examples and training data can harm the generalization ability of machine learning models for image classification[373].

We hypothesize that one major contributor towards this performance drop is the difference in the nuisance attributes between the train and test images (e.g., position, orientation and illumination conditions). For instance, Alcorn et al. [14] and Barbu et al. [24] reported that even state-of-the-art Deep Neural Networks (DNNs) see a sharp drop in accuracy on test data with diverse orientations. It is naturally desirable that machine learning models can generalize to such cases, and we refer to this as generalization beyond data bias.

Recent work by Madan et al. [241] presents a quantitative investigation on the capability of CNNs to generalize beyond data bias, and the underlying mechanisms driving this generalization behaviour. They begin by representing the entire dataset with an images grid, where each cell represents all images from a specific combination of object categories and nuisance attributes like viewpoint (Fig. 5.1.). By building training data from only a subset of cells (seen combinations), they quantitatively control data bias and evaluate how well CNNs generalize to images from remaining cells (unseen combinations) in the images grid. They also investigate the role of invariant and selective representations in driving such generalization, and present correlational evidence suggesting CNNs generalize better to unseen combinations as invariance and selectivity emerges.

Building on this observation, here we propose a regularization technique to facilitate better generalization beyond data bias which we call Invariance Enforcement Regularization (IER). We begin by deriving a set of sufficient conditions for neural activity that would enable generalization beyond data bias, which we refer to as identical and complementary activity. We then derive our regularization technique (IER) based on this analysis. Finally, we show the capability of this regularization technique at enabling better generalization across four different datasets.

We organized four datasets to evaluate the effectiveness of IER on generalization beyond data bias in a controlled manner following the images grid approach. Two of them are introduced in this paper. First is a new car CG dataset created analogously to[241], while the second is a new dataset of natural images created physically from scratch; this dataset consists of images of ten objects photographed under five illumination conditions, two object aspects, twenty object orientations, and five camera angles (Fig. 5.1) using a robotic arm setup. We plan to make both these datasets publicly available upon publication.

Our contributions are summarized as follows:

- We derive the sufficient conditions for neural activity in a layer to enforce invariance, which we refer to as identical and complementary activity, building on the framework of invariance and selectivity scores of individual neurons.
- Based on the relationship between identical activity and invariance, we propose a simple yet effective regularization technique (IER) that enforces invariance to facilitate generalization beyond data bias.
- We show the utility of IER on generalizing beyond data bias on four different datasets, including two new datasets introduced in this paper.

Related work If we consider each subset of training data with different nuisance attribute category (e.g., images taken with red light) as a domain, our problem setting of generalization and IER seem to have similar flavor to domain adaptation, domain generalization and methods used there. Domain adaptation is a type of transfer learning aiming at obtaining a good model for target domain with various types of constraints by utilizing a model trained with labeled data in the source domain[40,116,237,238,316,317,346]. In any tasks, data of target domain are required. Therefore, it is different from our problem setting. Domain generalization is another type of transfer learning aiming at obtaining a model with common invariant features useful for target domains using data in source domains[51,59,101,121,141,176,182,223,224,225,392]. The extreme case of domain generalization is called single domain generalization, which is the task of obtaining invariant models for multiple target domains from a single source domain[298]. There is a similarity with our study in the sense that invariant properties are learned from source domain(s), which may be considered as seen data in our study. One point of differences in problem settings is that our approach uses one mixed dataset, on the other hand, most of them uses several datasets belonging to different domains. Secondly, researches that treat 3D orientations and illumination conditions are very few. Furthermore, most of them tackle the problems by proposing methods using data augmentation[298], new network architecture[59,176], or new learning framework[51,101,121,141,182,223,224,225,392]. Finally, IER is a simple regularization technique, which can be applied to almost all deep learning models and is free from a priori assumptions about the bias of the data and data augmentation techniques (see Sec. 5.2.3).

5.2 Method

Invariance and selectivity of neurons have long been hypothesized to facilitate generalization in both biological[42,92,276,300,310,315] and artificial[129,291,347] neural networks. In this section we begin by presenting our formulation for quantifying these through invariance and selectivity scores of individual neurons analogously to[241]. We then analyze these scores and derive the ideal neural activity which would enable generalization beyond bias, and elaborate on the role of sparseness. Based on this analysis, we present our regularization technique to enforce invariant neural activity in order to facilitate generalization beyond data bias, and also introduce new scores for sparseness and complementarity in terms of neural activity of a layer. Finally, we explain the data preparation protocol used for conducting bias-controlled experiments.

5.2.1 Invariance and selectivity scores

We consider deep neural networks with activation functions that output non-negative values. Let $\mathbf{x}$ be an image. For the m-th neuron in the n-th layer, we use $a^{nm}(\mathbf{x}) \geq 0$ to denote its activity for the image. We consider the following variation-base decomposition of neural activity:

$$a^{nm}(\mathbf{x}) = \rho^{nm}\tilde{a}^{nm}(\mathbf{x}) + b^{nm}, \tag{5.1}$$

where $\rho^{nm} > 0$ is a scale factor, $b^{nm} \geq 0$ is a base activation and $\tilde{a}^{nm}$ is a variation of activity (see Appendix for details).

The selectivity score and invariance score are defined with a pair of attributes. In this study, we consider selectivity score with respect to object and invariance score with respect to a nuisance attribute (e.g., position, orientation or illumination condition). We use $\mathcal{I}$ to denote the set of all object categories and $\mathcal{J}$ the set of all nuisance attribute categories. We consider the averaged activity $\alpha_{ij}^{nm} = \text{average}_{i,j} \tilde{a}^{nm}(\mathbf{x})$, which is the average of $\tilde{a}^{nm}(\mathbf{x})$ over population of $\mathbf{x}$ for i-th object category and for j-th nuisance attribute category. In practice, we consider all images to be the population. Analogous to [241], to calculate these scores, we first identify the object category that a neuron is most active on average, i.e., $i^{*nm} = \text{argmax}_i \sum_j \alpha_{ij}^{nm}$. This is called preferred category. The invariance score I^{nm} is defined as

$$I^{nm} = 1 - (\max_j \alpha_{i^{*nm}j}^{nm} - \min_j \alpha_{i^{*nm}j}^{nm}). \tag{5.2}$$

It ranges from 0 to 1 and takes the maximum in the case that the neuron outputs the same value for the preferred category on average regardless of the nuisance attribute category. The selectivity score S^{nm} is defined as

$$S^{nm} = \frac{\hat{\alpha}^{nm} - \bar{\alpha}^{nm}}{\hat{\alpha}^{nm} + \bar{\alpha}^{nm}}, \tag{5.3}$$

where $\hat{\alpha}^{nm} = \frac{1}{\#(\mathcal{J})} \sum_j \alpha_{i^{*nm}j}^{nm}$ and $\bar{\alpha}^{nm} = \frac{\sum_{i \neq i^{*nm}} \sum_j \alpha_{ij}^{nm}}{\#(\mathcal{J})(\#(\mathcal{I})-1)}$ denote the average activity for the preferred category and for the remaining categories, respectively. This score also ranges from 0 to 1 and takes its maximum value in the case that the neuron only outputs positive values for the preferred category. These scores have been shown to be effective at analysing generalization beyond data bias[241].

5.2.2 Theoretical analysis on invariance and selectivity

In this section, we analyze how selectivity and invariance scores are impacted by neural activity.

Identical and complementary activity: We denote the activity of neurons in layer n as $\boldsymbol{a}^n(\mathbf{x}) = [a^{n1}(\mathbf{x}), a^{n2}(\mathbf{x}), \ldots, a^{nM_n}(\mathbf{x})]^\top$, where M_n denotes the number of neurons in layer n. As shown in Eq. 5.1, this vector can be described as $\boldsymbol{a}^n(\mathbf{x}) = \rho^n \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}) + \boldsymbol{b}^n$, where $\odot$ denotes the element-wise product. Let $\mathcal{X}_i$ be the set of all images belonging to object category i. To achieve the maximum value for the invariance scores for all neurons in layer n, it is sufficient that the neural activity achieves the following condition for all images and all object categories:

$$\widetilde{\boldsymbol{a}}^n(\mathbf{x}) = \widetilde{\boldsymbol{a}}^n(\mathbf{x}') \text{ for all } \mathbf{x} \in \mathcal{X}_i \text{ and } \mathbf{x}' \in \mathcal{X}_i. \tag{5.4}$$

On the other hand, to achieve the maximum value for the selectivity scores for all neurons in layer n, it is sufficient that the neural activity achieves the following condition for all images and all object categories:

$$\widetilde{\boldsymbol{a}}^n(\mathbf{x}) \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}') = \mathbf{0} \text{ for all } \mathbf{x} \in \mathcal{X}_i \text{ and } \mathbf{x}' \notin \mathcal{X}_i, \tag{5.5}$$

where $\odot$ denotes the element-wise product of vectors. We refer to Eq. Supp.1 as identical activity and Eq. Supp.2 as complementary activity.

It is to be noted that minimizing a norm of $\widetilde{\boldsymbol{a}}^n(\mathbf{x}) - \widetilde{\boldsymbol{a}}^n(\mathbf{x}')$ for images $\mathbf{x}$ and $\mathbf{x}'$ belonging to the same object category forces Eq. Supp.1 to hold. Although $\widetilde{\boldsymbol{a}}^n(\mathbf{x})$ and $\widetilde{\boldsymbol{a}}^n(\mathbf{x}')$ cannot be obtained during training phase, the following relationship holds: $\widetilde{\boldsymbol{a}}^n(\mathbf{x}) - \widetilde{\boldsymbol{a}}^n(\mathbf{x}') = (\boldsymbol{a}^n(\mathbf{x}) - \boldsymbol{a}^n(\mathbf{x}')) \oslash \rho^n$, where $\oslash$ denotes the element-wise division. Therefore minimizing a norm of $\boldsymbol{a}^n(\mathbf{x}) - \boldsymbol{a}^n(\mathbf{x}')$ is sufficient to minimize the corresponding norm of $\widetilde{\boldsymbol{a}}^n(\mathbf{x}) - \widetilde{\boldsymbol{a}}^n(\mathbf{x}')$. This observation is the basis of our regularization technique.
While the above can be achieved via a loss term, it is difficult to construct a regularization term which forces Eq. Supp.2 to hold as $\boldsymbol{b}^n$ is generally unknown during training phase. More details on this point are presented in Appendix.

Sparse activity: Given that complementarity is difficult to use for constructing a regularization term, we consider another condition on neural activity. We call the neural activity in layer n sparse when the following condition holds:

$$\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0} \ll \#(\widetilde{\boldsymbol{a}}^n(\mathbf{x})). \tag{5.6}$$

Here $\#(\cdot)$ denotes the number of elements and $\#(\cdot)_{\neq 0}$ denotes the number of non-zero elements. If neural activity is sparse, it is highly probable that complementarity holds (see Appendix).

5.2.3 Invariance Enforcement Regularization (IER)

In this section, we propose a regularization technique based on the analysis in Sec. 5.2.2, aiming to enforce invariant neural activity and facilitate generalization beyond data bias. We assume that a specific layer $\boldsymbol{n}$ is chosen for enforcing invariance.

Let $\boldsymbol{f}^{\text{in}}(\cdot)$ be the network on the input side, including the target layer $\boldsymbol{n}$, and $\boldsymbol{f}^{\text{out}}(\cdot)$ be the network after that. The activity of the target layer $\boldsymbol{n}$ for an image $\mathbf{x}$ is $\boldsymbol{a}^n(\mathbf{x}) = \boldsymbol{f}^{\text{in}}(\mathbf{x};\theta^{\text{in}})$ and the output of the whole network is $\boldsymbol{f}^{\text{out}}(\boldsymbol{f}^{\text{in}}(\mathbf{x};\theta^{\text{in}});\theta^{\text{out}}) = \boldsymbol{f}^{\text{out}}(\boldsymbol{a}^n(\mathbf{x});\theta^{\text{out}})$. Let $(\mathbf{x},\hat{\mathbf{x}})$ be a pair of different images belonging to the same category. We consider the following regularization term:

$$R = \|\boldsymbol{f}^{\text{in}}(\mathbf{x};\theta^{\text{in}}) - \boldsymbol{f}^{\text{in}}(\hat{\mathbf{x}};\theta^{\text{in}})\|_p, \tag{5.7}$$

where $\|\cdot\|_p$ denotes the Lp norm. As explained in Sec. 5.2.2, the minimization of R forces Eq. Supp.1 to hold, which is a sufficient condition for the maximum invariant scores. We add this term to the loss L of the original problem to obtain the overall loss function $L+\gamma R$, where γ is a regularization weight. We call this regularization technique Invariance Enforcement Regularization (IER).

Algorithm 3 represents a learning algorithm with IER, where $\mathrm{y}^{(k)}$ be the object category label corresponding to an image $\mathbf{x}^{(k)}$. Note that our algorithm requires labels only for the object category, and not for the nuisance attribute category. The object category labels are enough to create a new mini-batch $\{(\hat{\mathbf{x}}^{(k)},\hat{\mathrm{y}}^{(k)})\}_{k=1}^{B}$ for the regularization term apart from the original training mini-batch $\{(\mathbf{x}^{(k)}, \mathrm{y}^{(k)})\}_{k=1}^{B}$. Fig. 5.2 illustrates Algorithm 3.

Algorithm 1 Learning algorithm with IER

Require:Training data $\mathcal{D}^{(\mathrm{train})}$, Learning rate η, Regularization weight γ, Epochsize, Batchsize B, Norm $\|\cdot\|_p$

Ensure:Network parameters $\theta = (\theta^{\mathrm{in}}, \theta^{\mathrm{out}})$

while Epochsize do

  $\{(\mathbf{x}^{(k)}, \mathrm{y}^{(k)})\}_{k=1}^{B}, \{(\hat{\mathbf{x}}^{(k)}, \hat{\mathrm{y}}^{(k)})\}_{k=1}^{B} \leftarrow \mathrm{T}(\mathcal{D}^{(\mathrm{train})})$

  $R \leftarrow \frac{1}{B}\sum_k \|\boldsymbol{f}^{\mathrm{in}}(\mathbf{x}^{(k)}; \theta^{\mathrm{in}}) - \boldsymbol{f}^{\mathrm{in}}(\hat{\mathbf{x}}^{(k)}; \theta^{\mathrm{in}})\|_p$

  $L \leftarrow \frac{1}{B}\sum_k \mathrm{Loss}(\mathrm{y}^{(k)}, \boldsymbol{f}^{\mathrm{out}}(\boldsymbol{f}^{\mathrm{in}}(\mathbf{x}^{(k)}; \theta^{\mathrm{in}}); \theta^{\mathrm{out}})))$

  $\theta \leftarrow \theta - \eta \nabla_\theta (L + \gamma R)$

end while

function $\mathrm{T}(\mathcal{D})$

  for $k = 1, \ldots, B$ do

    $(\mathbf{x}^{(k)}, \mathrm{y}^{(k)}) \leftarrow$ sample from $\mathcal{D}$

    $(\hat{\mathbf{x}}^{(k)}, \hat{\mathrm{y}}^{(k)}) \leftarrow$ sample from $\mathcal{D}$ with $\hat{\mathrm{y}}^{(k)} = \mathrm{y}^{(k)}$

  end for

return $\{(\mathbf{x}^{(k)}, \mathrm{y}^{(k)})\}_{k=1}^{B}, \{(\hat{\mathbf{x}}^{(k)}, \hat{\mathrm{y}}^{(k)})\}_{k=1}^{B}$

end function

While any norm can work to enforce invariant neural activity in principle, different norms may work differently on sparseness, and thus differently on complementarity and selectivity. The effects of different norms on sparseness, complementarity and selectivity will be investigated experimentally using the metrics we introduce below.

IER is a simple, general purpose regularizer that can be applied to all deep learning models as long as the activation function is non-negative. It is also worth highlighting that no a priori assumptions about the bias of the data are required, and no data augmentation are also required to enforce invariance; this is a prominent advantage of IER.

5.2.4 Scores of sparseness and complementarity

To evaluate the sparseness and complementarity in addition to invariance and selectivity, we introduce two metrics: the mean rate of activation (mrA) and mean overlap of activation (moA). We define neurons with $\widetilde{a}^{nm}(\mathbf{x}) \neq 0$ to be activated. Note that the following scores range from 0 to 1, and the closer to 1, the better.

Mean rate of Activation (mrA): This score measures the ratio of activated neurons to all neurons as an indicator of the sparseness. Formally, it is defined as

$$1 - \frac{1}{\#(\mathcal{I})} \frac{1}{\#(\mathcal{X}_i)} \sum_{i \in \mathcal{I}} \sum_{\mathbf{x} \in \mathcal{X}_i} \frac{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0}}{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))}. \tag{5.8}$$

Mean overlap of Activation (moA): This score takes a sufficiently large number of combinations of samples with alternating object categories i and uses the overlap of the activation as the indicator of the complementarity. Let $\mathcal{X}_\mathrm{r}$ be a set of randomly sampled combinations $(\mathbf{x}, \mathbf{x}')$ with different labels of object category. The moA is defined as

$$1 - \frac{1}{\#(\mathcal{X}_\mathrm{r})} \sum_{(\mathbf{x}, \mathbf{x}') \in \mathcal{X}_\mathrm{r}} \frac{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}') \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0}}{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))}. \tag{5.9}$$

In this study, we set $\#(\mathcal{X}_\mathrm{r}) = 500$, which we experimentally confirmed to be large enough for our purpose.

5.2.5 Bias-controlled experiment

Following[241], we conduct bias-controlled experiments to evaluate the performance of IER. For this, we begin by preparing datasets where each image has a label for a nuisance attribute as well as a label for the object category. Then, we select images with certain combinations of object categories and nuisance attribute categories (called seen data; see Fig. 5.1) which we use as training and validation data (for hyper-parameter tuning). Finally, the trained models are tested on the images belonging to remaining combinations never shown during training (called unseen data; see Fig. 5.1). Through this protocol, we can quantitatively control the bias between the training data and the test data, and can evaluate generalization beyond data bias explicitly.

Creating seen and unseen data: Let $\mathbf{x}^{(k)}$ be an image and $\mathbf{y}^{(k)} = (i^{(k)}, j^{(k)})$ be the corresponding label. We divide our dataset as follows. First, we select certain combinations of object and nuisance attribute categories $\mathcal{S} \subset \mathcal{I} \times \mathcal{J}$. Seen data is defined as

$$\mathcal{D}^{(\text{seen})} = \{(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}) | \mathbf{y}^{(k)} \in \mathcal{S}\}. \tag{5.10}$$

This is further divided into train data and validation data. We impose conditions $\{i \mid (i,j) \in \mathcal{S}\} = \mathcal{I}$ and $\{j \mid (i,j) \in \mathcal{S}\} = \mathcal{J}$ to ensure that $\mathcal{D}^{(\text{seen})}$ contains all object categories and all nuisance attribute categories at least once (but not all combinations of them). Unseen data is defined as

$$\mathcal{D}^{(\text{unseen})} = \{(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}) | \mathbf{y}^{(k)} \notin \mathcal{S}\}. \tag{5.11}$$

The term unseen test accuracy refers to the accuracy on $\mathcal{D}^{(\text{unseen})}$. We also define seen rate as $s = \#(\mathcal{S})/\#(\mathcal{I} \times \mathcal{J})$. To examine the dependency of the generalization on seen rate, we vary seen rate and attribute combinations as $\mathcal{S} \subset \mathcal{S}'$ if $s < s'$, while keeping $\#(\mathcal{D}^{(\text{seen})}(\mathcal{S})) = \#(\mathcal{D}^{(\text{seen})}(\mathcal{S}'))$.

5.3 Datasets

We use the following four datasets, which have labels for both object category and several other nuisance attributes to evaluate the effectiveness of IER. For details, see Appendix.

MNIST-Positions Starting with the MNIST dataset[215], we created a dataset of 42x42 pixels with nine numbers (0 to 8; 567378 images) by resizing images to 14x14 and placing them in one of 9 possible positions in a 3x3 empty grid. We call it MNIST-Positions. In our experiments, the numbers are considered to be the object category set $\mathcal{I}$ and the positions where the numbers are placed is considered as the nuisance attribute set $\mathcal{J}$. Samples are shown in Fig. ??. We used 100K training, 10K validation and 10K test images for our experiments.

iLab-Orientations iLab-2M is a dataset created from iLab-20M dataset[?]. The dataset consists of images of 15 categories of physical toy vehicles photographed in various orientations, elevations, lighting conditions, camera focus settings and backgrounds. It has 1.2M training images, 270K validation images, 270K test images, and each image is 256x256 pixels. We chose from the original iLab-2M dataset six object categories — bus, car, helicopter, monster truck, plane, and tank as $\mathcal{I}$ and six orientations as $\mathcal{J}$. We call it iLab-Orientations. Samples are shown in Fig. ??. The sample size is 471791, and we resized each image to 64x64 pixels. We used 70K train, 8K validation and 8K test images for the experiment.

CarCGs-Orientations CarCGs-Orientations is a new dataset that consists of images of ten types of cars in various conditions rendered by Unreal Engine[107]. The conditions consists of ten orientations, three elevations, ten body colors, five locations and three time slots. We synthesize 45K images with 1920x1080 pixels and resize them as 224x224 pixels for our experiment. We chose ten types of cars as $\mathcal{I}$ and ten orientations as $\mathcal{J}$. Samples are shown in Fig. ??. Further samples are provided in supplementary materials. In the experiment, we used 3400 train, 445 validation and 800 test images.

MiscGoods-Illumination MiscGoods-Illumination is a novel dataset constructed for this study. The dataset consists of ten physical miscellaneous goods photographed with five illumination conditions, two object aspects, twenty object orientations, and five camera angles. This amounts to a total of 10K images. Each image is 640x480 pixels in size. We chose five object categories — stuffed dolphin, stuffed whale, metal basket, imitation plant and cup as $\mathcal{I}$ and five illumination conditions as $\mathcal{J}$ as shown in Fig. 5.1. Further samples are provided in supplementary materials. We resize the images to 224x224 pixels for our experiments. we used 800 train, 200 validation and 200 test images.

5.4 Experiments

In this study, we conduct two kinds of experiments. (1) We test whether IER enforces invariance score and improves generalization performance beyond data bias using MNIST-Positions, iLab-Orientations, CarCGs-Orientations, and MiscGoods-Illumination with representative values of the seen rate s and the regularization weight γ. We also provide analysis on other scores. (2) We examine the dependency of the effect of IER on the seen rates and regularization weight using MNIST-Positions and iLab-Orientations. Both (1) and (2) are bias-controlled experiments.

The results of the experiments have revealed the following. (1) Generalization performance beyond data bias, test accuracy on unseen data $\mathcal{D}^{(\text{unseen})}$, is improved by IER for all patterns of norms for all bias-controlled experiments; invariance scores are enforced in almost all cases as expected. (2) As the seen rate is increased, the performance improves. However, IER worsens the performance if the seen rate is extremely low. About weight dependency, appropriate weights improve the performance for all norms. However, if the weight is greater than the appropriate value, the performance drops catastrophically.

Details of experiments ResNet-18[163] is adopted as the network for all experiments. All neurons employ the rectified linear function $g(z) = \max\{0, z\}$ and satisfy $a^{nm}(\mathbf{x}) \geq 0$. Glorot uniform initializer[125] is adopted for the network weights initialization for all experiments. Further network configurations are shown in

Appendix. IER is applied to the last fully-connected layer "activation_17" with 512 neurons shown in Appendix. Adam[197] is employed as the optimization algorithm. The cross-entropy loss is employed as the loss L.

The pixels of images are normalized within 0 to 1 as a preprocessing for all datasets. The epoch size, learning rate, and batch size are confirmed to produce reasonable accuracy in the vanilla case (i.e., without IER) for each dataset and we employ the same values for all experiments with the same dataset. For example, we use 15, 0.001, and 128 as epoch size, learning rate and batch size, respectively, for MNIST-Positions. For the invariance and selectivity scores, we use the median of the scores of all neurons to represent the layer activity. All other parameters are shown in Appendix. The combinations of the object category i and the nuisance attribute category j in the seen data were chosen randomly as long as satisfying the conditions explained in Sec. 5.2.5, and the unseen data was created accordingly. The source codes are implemented based on Python v3.6.9, using TensorFlow v2.1.0 and NumPy v1.18.1.

5.4.1 Effects of IER

We examine the invariance scores and unseen test accuracy (test accuracy on unseen data $\mathcal{D}^{(\text{unseen})}$) with and without IER for all four bias-controlled datasets. We also examine the differences of the effects on sparseness, complementarity and selectivity scores caused by the norm selection. Finally, we examine the impact of each score on generalization performance. The seen rate s employs 4/9, 3/6, 5/10, and 3/5 for MNIST-Position, iLab-Orientations, CarCGs-

Orientations, and MiscGoods-Illumination, respectively. The regularization weight γ for L1, L2, and Linf norms are set at 0.001, 0.01, and 0.01, respectively. We conduct five trials and provide arithmetic means and 95% confidence intervals of scores and unseen test accuracy. Regarding invariance and selectivity scores, the median in all neurons is regarded as layer scores. For five trials, we have sampled five pairs of seen data $\mathcal{D}^{(\mathrm{seen})}$ and unseen data $\mathcal{D}^{(\mathrm{unseen})}$ keeping the same seen rate s. Train accuracy and test accuracy on seen data are reported in Appendix. Source codes are also provided in supplementary materials.

Enforcement of invariance IER is designed primarily to target identical activity (Eq. Supp.1) and is therefore expected to improve invariance scores. Fig. 5.4 shows that invariance scores for MNIST-Positions and iLab-Orientations are improved with all norms, and among all, L2 is the best. In CarCGs-Orientation, improvement of invariance scores is relatively small; there is a decrease from vanilla case by L1 norm in Fig. 5.4c. It is likely that the small differences between car categories made it difficult to enforce invariance. In MiscGoods-Illumination, the invariance scores are also improved in most cases, but they are smaller than the other datasets; The invariance score of Linf case is decreased from the vanilla case as shown in Fig. 5.4d. The classification of object categories in MiscGoods-Illumination requires color information, which may cause the difficulty of obtaining invariance to illumination.

Enforcement of sparseness, complementarity and selectivity The improvement in sparseness, complementarity, and selectivity scores is not explicitly imposed by the regularization term of IER (Fig. 5.4). However, it is expected that appropriate norm can improve them. As shown in Fig. 5.4, for all datasets, mrA, moA and selectivity scores are increased by L1 norm. This is explained by the Lasso-like effect[370] of L1 norm, which promotes sparse activation and complementarity, and improves selectivity consequently. Scores tend to be decreased for the other norms in the three datasets. The decrease is particularly sharp for the MNIST-Positions. This is because the neuronal activity is not sparsified by L2 and Linf; complementarity and selectivity are considered to have been reduced by enforcing invariance.

Improvement of unseen test accuracy There is a substantial increase in performance from vanilla case with all norms in all datasets as shown in Fig. 5.4; especially for MNIST-Positions, there is a great improvement of 0.678 (vanilla) to 0.973 (Linf) as shown in Fig. 5.4a, and even for a relatively difficult task MiscGoods-Illumination, there is an improvement of 0.340 (vanilla) to 0.475 (L2) as shown in Fig. 5.4d. The smallest improvement is in iLab-Orientations, but there is still an increase of 0.800 (vanilla) to 0.838 (L2), which seems a sufficient improvement (Fig. 5.4b). In almost all cases, the unseen test accuracy is improved with an improvement of invariance scores. Although there is a large decrease in selectivity in L2, and Linf, its impact on the unseen test accuracy is limited (but not entirely negligible). In fact, the L2 norm for MNIST-Positions causes a drop in the selectivity score, and the unseen test accuracy is not the best, despite the greatest improvement in invariance scores (see Fig. 5.4a). As seen in the experiment of the MiscGoods-Illumination (Fig. 5.4a), the unseen test accuracy is also improved if the invariance scores are relatively enforced even if the absolute values of the invariance score are small. Finally, it is certain that the increase in the invariance score improves the unseen test accuracy in almost all cases. However, we would add that the relationship is highly nonlinear, and in some trials, the relationship between score and unseen test accuracy is not straightforward.

5.4.2 Seen rate and weight dependency

We investigate the dependency of unseen test accuracy on the seen rate s and regularization weight γ. We performed experiments on all norms and only one trial on each parameter in MNIST-Positions and iLab-Orientations. Results are shown in Fig. 5.5.

Seen rate dependency In this experiment, we fix the weight γ and vary the seen rate s. The weights for L1, L2, and Linf norms are fixed at 0.001, 0.01, and 0.01, respectively. Figs. 5.5a and 5.5b show the relationship between the seen rate s and unseen test accuracy. In the high seen rate settings, the unseen test accuracy with norms is higher than that without them (vanilla) in almost all trials. However, we note that the unseen test accuracy is lower with norms than vanilla under extremely low seen rate settings. This is due to the fact that although IER tends to induce invariance with respect to the nuisance attribute category utilizing the seen data, it is not possible to take pairs of samples with large variety of changes in attribute category within the same object category when the seen rate is extremely low. Also, extremely biased data may make optimization difficult.

Weight dependency We investigate the unseen test accuracy for all norms by decreasing the regularization weight γ from 1 to 0.0001. The seen rates are fixed at $4/9$ and $3/6$ for MNIST-Positions and iLab-Orientations, respectively. Figs. 5.5c and 5.5d show the dependence of unseen test accuracy on weight. Choosing the appropriate weight improves performance on both datasets for all norms; the actual values were 0.001, 0.01, and 0.01 for L1, L2, and Linf, respectively. The appropriate weights in both datasets are in the same range, and the dependence on the dataset is relatively small; we recommend to use these values at first. If the weight is smaller than the appropriate value, the performance gradually approaches vanilla case. If the weight is too large, the performance drops drastically. This is because over-enforcement of invariance makes the model always output the same value.

5.5 Limitations

There are three limitations in this study. First, invariance and selectivity (and sparseness and orthogonality related to selectivity) were considered as the main factors of generalization beyond data bias; however, we cannot exclude the possibility that other implicit factors may affect generalization. Second, only ResNet-18 was employed for the experiments. Although it has been shown in[241] that other neural networks show similar trends regarding selectivity score, invariance score, and generalization performance to ResNet-18, it has not been tested in this study. Finally, we treated only three nuisance attributes explicitly, namely, position, orientation and illumination condition though they are major causes of data bias. The results of this study are supposed to be valid for other nuisance attributes as well, but they have not been tested.

5.6 Conclusion

This study proposed a novel regularization method, named Invariance Enforcement Regularization (IER), to facilitate generalization beyond data bias by enforcing the invariant nature of the activity of intermediate layer neurons and demonstrates its effectiveness in bias-controlled experiments with four datasets. In most cases, improvements in invariance bring improved generalization performance. Regarding selectivity of neural activity, L1 norm seems to be the best choice from our experiments. It is also important to note that the method is very simple and applicable to almost all models and learning frameworks. Therefore, we believe that IER has a significant impact on the issue of data bias. As for future work, we are planning to test the effects of IER with datasets whose bias is introduced by actual data collection process and not controlled. For instance, CIFAR-10.1[303] and ImageNetV2[304].

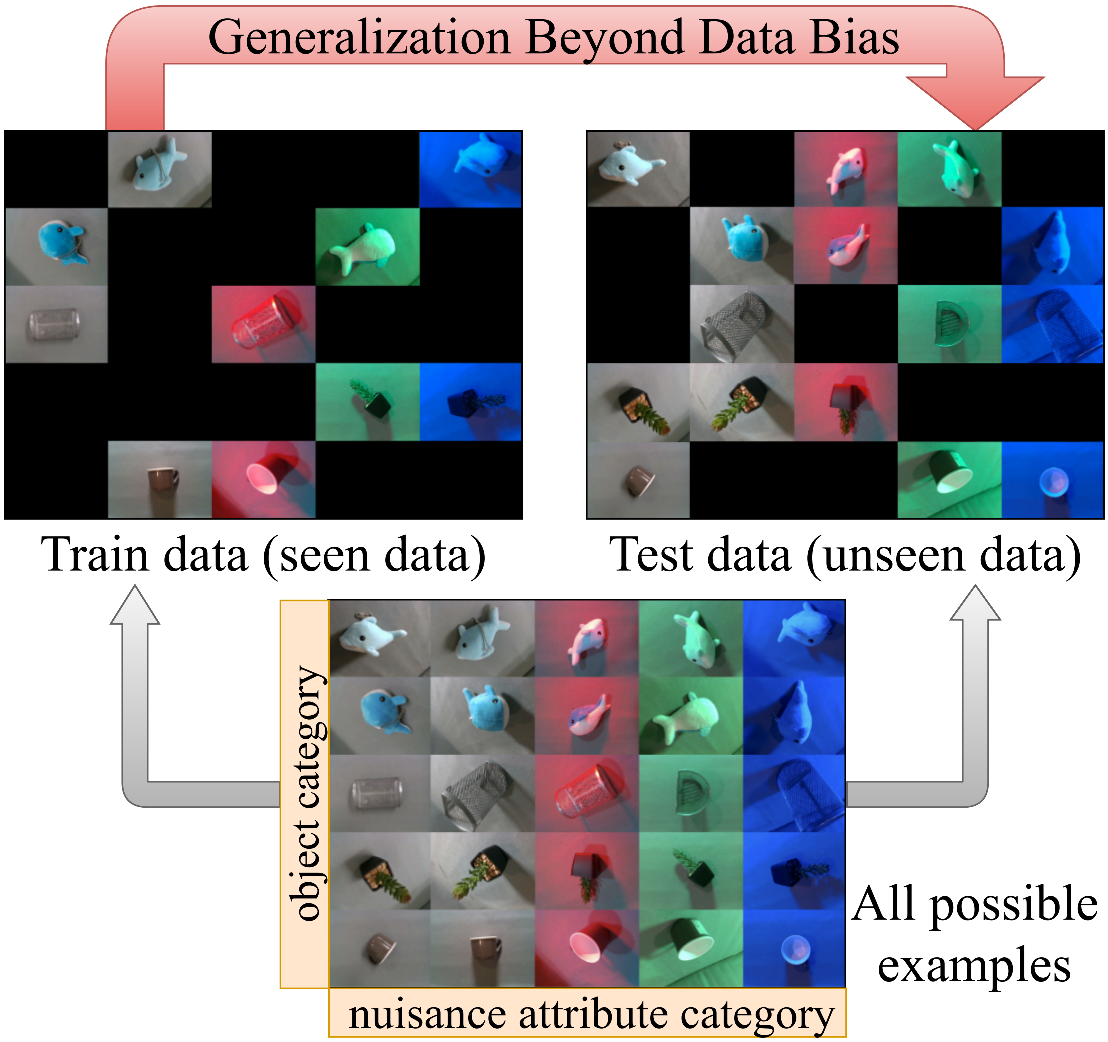


Figure 5.1: How to evaluate the effectiveness of our proposed method (Invariance Enforcement Regularization; IER) on generalization beyond data bias. We train a deep neural network (DNN) with IER using a set of images; each image is characterized by a certain combination of object category and nuisance attribute category. Then we test the DNN on the complementary set.

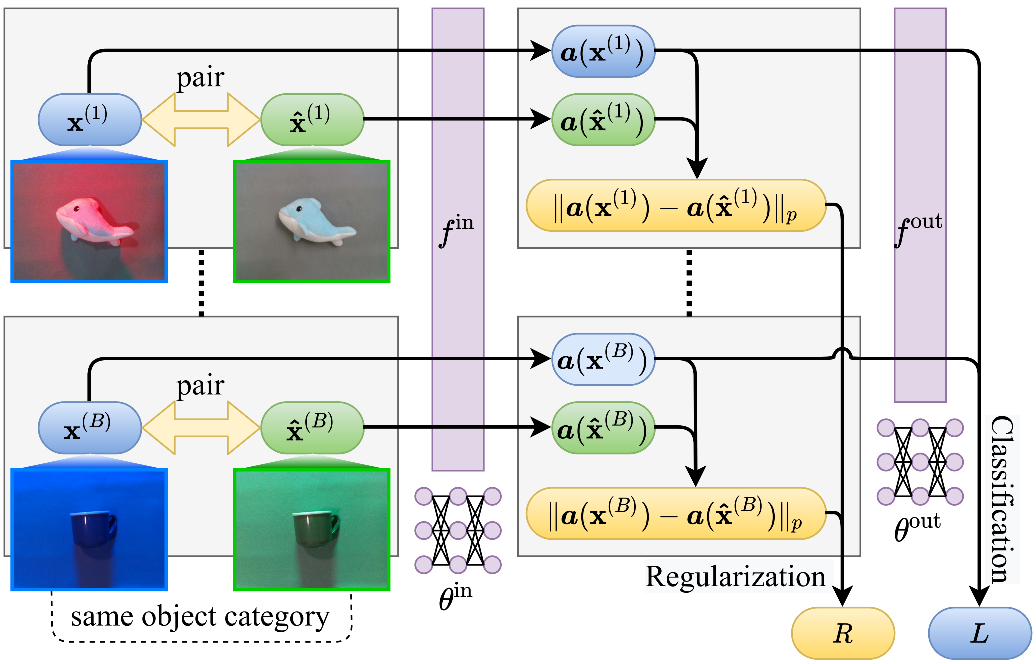
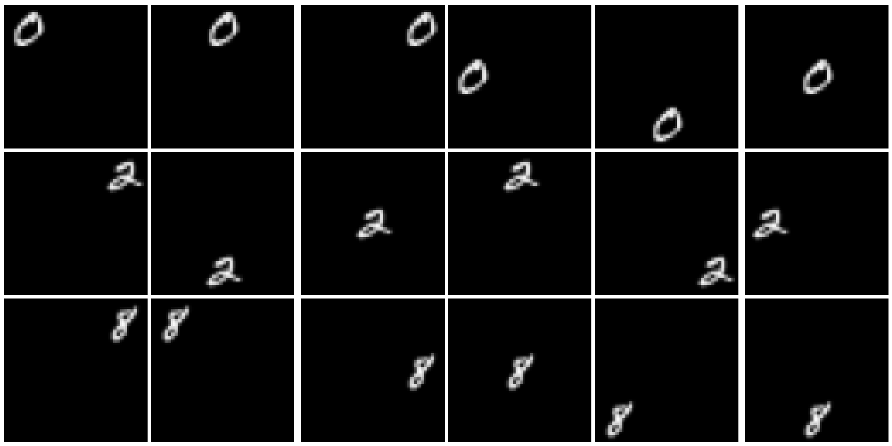
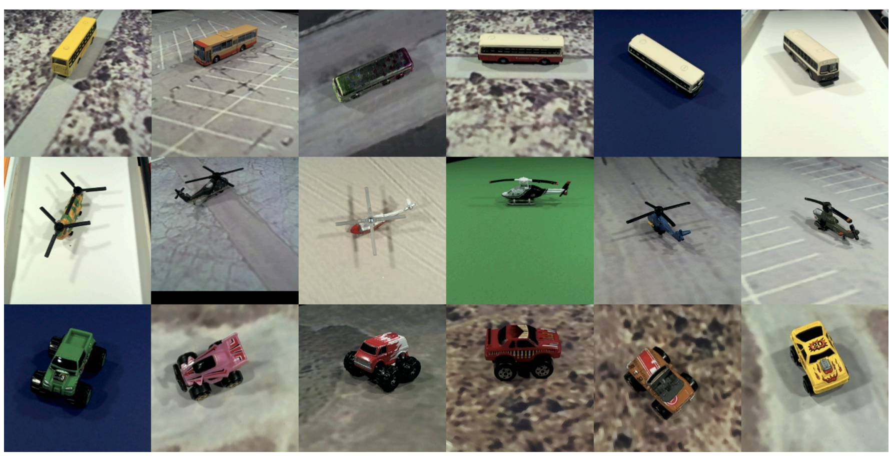


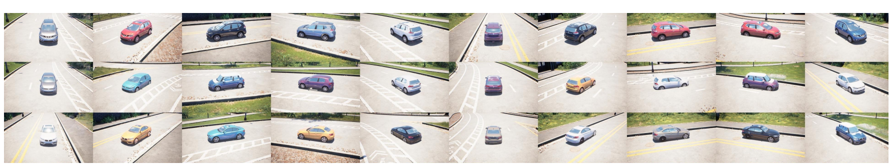
Figure 5.2: Depiction of a learning algorithm with Invariance Enforcement Regularization (IER): Pairs of images that belong to the same object category are fed in the network. The norms between the pairs of activity $(\boldsymbol{a}^n(\mathbf{x}), \boldsymbol{a}^n(\hat{\mathbf{x}})) = (\boldsymbol{f}^{\text{in}}(\mathbf{x}; \theta^{\text{in}}), \boldsymbol{f}^{\text{in}}(\hat{\mathbf{x}}; \theta^{\text{in}}))$ in a middle layer are used as the regularization term. Classification loss L is calculated with the network output. The total loss is produced as $L + \gamma R$, where γ denotes the regularization weight.



(a) MNIST-Positions

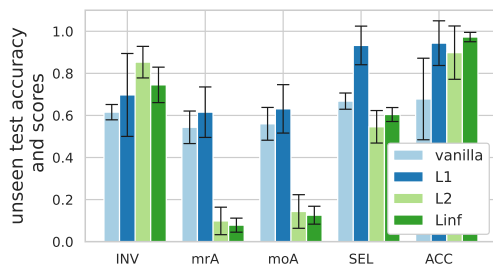


(b) iLab-Orientations

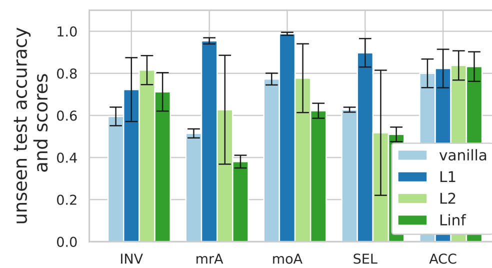


(c) CarCGs-Orientations

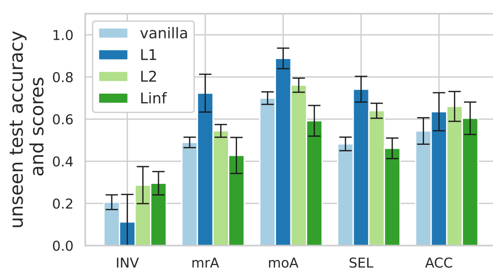
Figure 5.3: Sample images of (a) MNIST-Positions, (b) iLab-Orientations, and (c) CarCGs-Orientations. Samples from each dataset are arranged in a grid pattern. Each row indicates object categories, and each column indicates nuisance attribute categories.



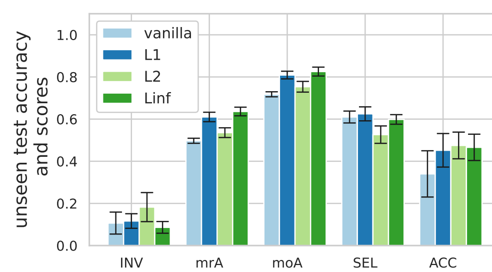
(a) Scores and accuracy in MNIST-Positions



(b) Scores and accuracy in iLab-Orientations

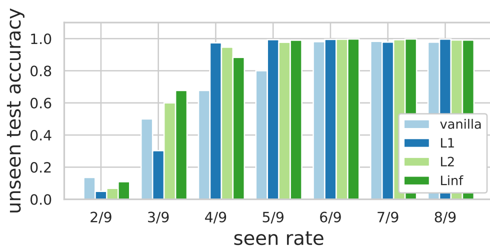


(c) Scores and accuracy in CarCGs-Orientations

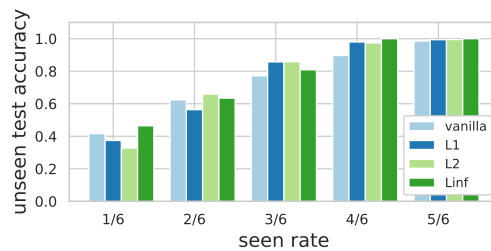


(d) Scores and accuracy in MiscGoods-Illumination

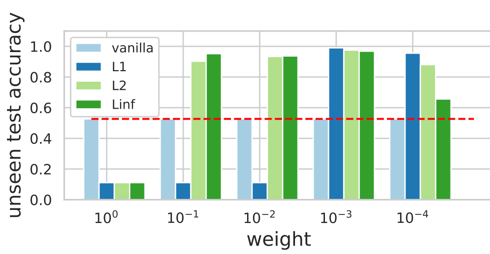
Figure 5.4: Enforcement of scores and improvement of unseen test accuracy: each subfigure shows the mean and 95% confidence interval for the scores and unseen test accuracy for each norm in each dataset. Inv, mrA, moA, and SEL denote invariance score, mean rate of activity, mean overlap of activity, and selectivity score, respectively. Median in all neurons of invariance and selectivity scores are regarded as layer scores. ACC denotes the unseen test accuracy (test accuracy on unseen data $\mathcal{D}^{(\text{unseen})}$).



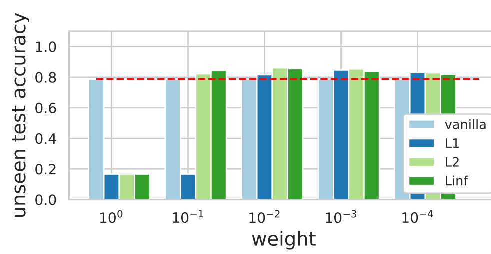
(a) Seen rate dependency in MNIST-Positions



(b) Seen rate dependency in iLab-Orientations



(c) Weight dependency in MNIST-Positions



(d) Weight dependency in iLab-Orientations

Figure 5.5: Seen rate dependency and weight dependency: (a) and (b) show the unseen test accuracy on several seen rates in MNIST-Positions and iLab-Orientations, respectively; the vertical axis indicates the unseen test accuracy (the test accuracy on unseen data $\mathcal{D}^{(\mathrm{unseen})}$) and the horizontal axis indicates the value of the seen rate s. The regularization weight γ for L1, L2, and Linf norms are 0.001, 0.01, and 0.01, respectively. (c) and (d) show the unseen test accuracy dependency on the regularization weight γ; the vertical axis indicates the unseen test accuracy and the horizontal axis indicates the value of the weight γ. The seen rates are 4/9 and 3/6 for MNIST-Positions and iLab-Orientations, respectively. All experiments were performed for vanilla and all norms of L1, L2, and Linf; one trial for each seen rate s or weight γ of each data.

Video games are the Petri dish of human behavior.

Nir Eyal.

6 Improving generalization in Reinforcement Learning.

6.1 Introduction

Consider the process of learning how to play tennis. You might think that the best way to prepare for an outdoor match is to train under the same outdoor

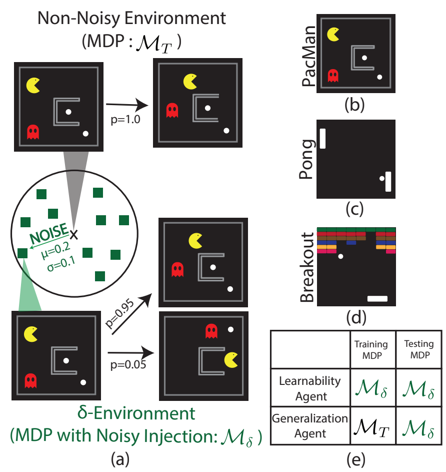


Figure 6.1: ATARI games modified with Noise Injection. (a) In the original Target Environment($\mathcal{M}_T$), when the agent (PacMan) moves right, PacMan moves right with probability 1.0. Noise Injection allows us to create multiple worlds in the vicinity of this environment by adding controlled Gaussian noise (δ) to the original Transition Function (T). When the agent takes the action right in these $\delta-$environments, with a low probability the game may transition to a state which would not be possible in non-noisy PacMan. For brevity, we refer to these transitions as non-standard transitions which are 0 probability in the original Target, but are now possible. Experiments with noise injection are presented on three ATARI games—(b) PacMan, (c) Pong, and (d) Breakout. (e) We compare two agents with these environments—a Learnability agent trained and tested on the same target environment ($\mathcal{M}_\delta$), and a Generalization agent trained on a different MDP ($\mathcal{M}_T$) and tested on $\mathcal{M}_\delta$.

conditions you will face during the match. However, training in a calm, noise-free indoor environment instead can help focus on mastering the fundamentals of tennis without the added challenge of sources of noise like wind. We refer to this phenomenon as the Indoor-Training Effect. Here we model this problem using reinforcement learning (RL) agents. Surprisingly, we found that under certain conditions, training in a noise-free environment can lead to better performance when tested in a noisy environment—just like tennis. This phenomenon challenges our intuitions about the standard way to train RL agents where conventional wisdom would suggest that the best approach to perform well on a

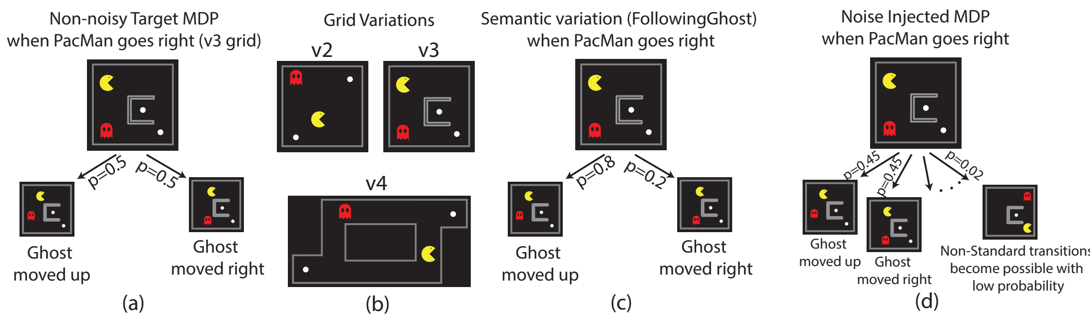


Figure 6.2: Schematic illustration of variations for Pacman.(a) Game dynamics when the agent picks the action right in a standard, non-noisy MDP for the v3 grid. The ghosts' actions follow a uniform probability distribution over possible moves and move up or right with an equal probability of 0.5. This is referred to as a RandomGhost. (b) Grid variations for Pacman—v2, v3, and v4. These grids vary in size, positions of walls, and positions of food pellets. v2, v3, and v4 are designed to be increasingly harder. (c) Semantic variations whereby there is a meaningful change in the distribution of game elements. Here, a FollowingGhost is depicted which has a higher probability of taking a move that brings it closer to the Pacman (0.8). (d) Noise injected MDP generated by adding Gaussian noise to the standard transition function. Alongside states reachable by the ghost taking a legal move, non-standard transitions now become possible which result in the game reaching states otherwise unreachable.

target problem is to train an RL agent on the same test environment.

Environments in RL are usually described using Markov Decision Process (MDP). An MDP is defined by a State Space $\mathcal{S}$, an Action Space $\mathcal{A}$, a Transition Function $\mathcal{T}$, and a Reward Function $\mathcal{R}$. In practice, these parameters are assumed to be known or approximated with reasonable precision[29,132]. A significant challenge in RL is generalizing to environments that differ from the training environment[71,190,93]. To address this, the RL community has focused on training agents capable of learning policies that perform well in novel, unseen environments at deployment time[199,261,277,113,32]. The complexity of this task has called for ingenious ways of aligning the policy learned by the agent in training environments with the testing optimal policy. Notable approaches include using human feedback[313], using language[367,395,350], and using vision[143,281,131].

To study this, we explored zero-shot policy transfer where a policy trained in one environment is tested on a different environment. We extended past works which focused on uncertainty in the transition probabilities[273,262,132], and propose a novel framework for studying zero-shot policy transfer in environments with controlled, quantifiable distribution shifts in the transition probabilities.

Our framework introduces these shifts by computing the transition function of an MDP, and adding small Gaussian noise to its entries. Starting with an environment ($\mathcal{M}_T$), noise is sampled and added to it to obtain a new MDP ($\mathcal{M}_\delta$). We refer to this approach as Noise Injection and the resulting new MDPs as δ-environments as in Fig. 11.1. Noise injection introduces several non-standard transitions, which had zero probability in the original MDP. Multiple such environments can be created by sampling noise and the noise serves as a metric of distance between environments. This approach allows us to create multiple worlds starting from the same MDP, with quantitative control over the variations in the transition probabilities. An increase in the standard deviation of the Gaussian noise results in increasingly perturbed MDPs. We report experiments with Noise Injection on multiple domains across three ATARI games—PacMan, Pong, and Breakout.

To study policy transfer we define two agents: a Learnability Agent ($\mathcal{L}_\delta$) which is trained and tested on the same δ-environment ($\mathcal{M}_\delta$), and a Generalization Agent ($\mathcal{G}_T$) which is trained on the original noise-free environment ($\mathcal{M}_T$) but tested on the δ-environment ($\mathcal{M}_\delta$). Conventional wisdom suggests

ATARI Game	Grid Variations	Noise Injected Variations	Semantic Variation	Total
PacMan	v2, v3, v4	$\delta = 0$ (No Noise) $\delta \sim \mathcal{N}(0, 0.1)$ $\delta \sim \mathcal{N}(0, 0.5)$	RandomGhost FollowingGhost ($p = 0.3, 0.6$) TeleportingGhost ($p = 0.5, 0.2$)	33
Pong	p1, p2	$\delta = 0$ (No Noise) $\delta \sim \mathcal{N}(0, 0.1)$ $\delta \sim \mathcal{N}(0, 0.5)$	RandomPaddle FollowingPaddle ($p = 0.3, 0.6$)	18
Breakout	b1, b2, b3	$\delta = 0$ (No Noise) $\delta \sim \mathcal{N}(0, 0.1)$ $\delta \sim \mathcal{N}(0, 0.5)$	-	9

Table 6.1: Overview of experimental protocol. Our experiments include multiple variations of three ATARI games—PacMan, Pong, and Breakout. For each game, we have multiple grid variations. When introducing variations in these grids with noise injection, we report results for two levels of added noise—a low-noise setting: $\delta \sim \mathcal{N}(0, 0.1)$, and a high-noise setting: $\delta \sim \mathcal{N}(0, 0.5)$. Furthermore, for each grid we introduce further variations by modifying the distribution of the stochastic game element (ghost in PacMan, and the computer paddle in Pong). In all, we report results on 60 MDPs across these games.

that the Learnability Agent should perform better as it is trained and tested on the same environment. However, our study across 60 MDPs built on ATARI games reveals a surprising finding—there are several cases where the Generalization Agent outperformed the Learnability Agent. We confirmed that this finding extends beyond our setup of noise injection and δ-environments and also holds true for game variations including varying the Ghost behaviour in PacMan, and Paddle behaviour in Pong. We refer to these as semantic variations in MDPs.

In conclusion, to better understand this phenomenon we analyzed the exploration patterns of the Learnability and Generalization Agents, and the corresponding policies learned by them. Our analyses revealed that $\mathcal{L}_\delta$ agents outperformed $\mathcal{G}_T$ agents, as expected from the literature, when $\mathcal{G}_T$ agents fail to explore the same State-Action pairs as the $\mathcal{L}_\delta$ agents. In contrast, when there were no large differences in their exploration patterns, the performance of $\mathcal{G}_T$ aligned or exceeded that of $\mathcal{L}_\partial$ agent.

Preliminaries: Reinforcement Learning

Similar to[54], our work considers Reinforcement Learning (RL) as a group of algorithms designed to solve problems formulated as Markov Decision Processes (MDPs). A Markov Decision Process is characterized by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \lambda)$, representing the collection of potential world states ($\mathcal{S}$), space of actions ($\mathcal{A}$), the transition function ($\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{P}(\mathcal{S})$), the reward function ($\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{R}$), and a discount factor ($0 < \gamma \leq 1$). The objective is to identify policies ($\pi : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{R}$) that maximize cumulative rewards.

Q-learning[402] and SARSA[188] are two algorithms to learn such policies. Both Q-Learning and SARSA algorithms update the Q-values of state-action pairs, but they differ in their approaches. Q-Learning focuses on the maximum expected future rewards, and updates Q-values using the formula:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s,a) \right] \tag{6.1}$$

where α is the learning rate, γ is the discount factor, and s, s', a, a', r represent the current state, next state, current action, next action, and immediate reward, respectively.

On the other hand, SARSA updates Q-values based on the actual policy's actions with the formula:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma Q(s', a') - Q(s,a) \right] \tag{6.2}$$

Here the update incorporates both immediate rewards and the Q-value of the actual next action taken.

Agents need to balance two critical aspects: exploration and exploitation. Exploration involves trying potentially less optimal actions to understand the environment better. Conversely, exploitation means choosing actions known to yield high rewards. We report results with the Boltzmann and the ε-greedy exploration strategies. Boltzmann exploration determines the probability of selecting an action as follows:

$$Pr_q(a) = \frac{e^{Q(s,a)/\tau}}{\sum_{a'} e^{Q(s,a')/\tau}} \tag{6.3}$$

The constant τ is referred to as the temperature. On the other hand, the ε-greedy strategy is simpler and more direct—the agent selects a random action with probability ε, and the action with the highest Q-value with probability $1-\varepsilon$.

6.2 Related Works

Generalization benchmarks in RL involve training and testing across different subsets of tasks, levels, or environments. Recent years have seen several generalization benchmarks, which include variations in the state space[145], dynamics[102], observation[449], reward function[23], and new game levels[187], among others. These tasks require explicit modeling of variations to effectively assess generalization, highlighting the need for robust evaluation protocols.

Contextual Markov Decision Processes (CMDP) provide a formal structure for this, where environments are sampled from a class of contexts, with agents trained on a subset and tested on a disjoint subset. These contexts are generated through two primary methods: Procedural Content Generation (PCG), which relies on a seed value for environment generation, and Controllable Environments (CE), which allow for manipulation of individual components. The integration of a suitable evaluation protocol with these contexts helps define the relationship between training and testing sets, which can range from interpolation to full extrapolation. Some examples of benchmarks using these frameworks include the OpenAI Procgen benchmark[70] and the Distracting Control Suite[354] for PCG[13] and RWRL[102] for controllable environments. A major drawback in these benchmarks is the lack of a clearly defined metric for measuring how the distance between different contexts affects agent performance.

To solve this issue we draw inspiration from work studying generalization under controlled, quantifiable distribution shifts in computer vision. These studies include shifts in 3D rotation[260,244], category-viewpoint combinations[240], incongruent scene context[38], novel light and viewpoint combinations[319], object materials[245] and texturess[118,256], and non-canonical viewpoints[24], among others.

Generating MDPs for investigating generalization

We created 60 different MDPs across three ATARI games (PacMan, Pong, and Breakout) by varying grid layouts, distributions defining the stochasticity of different game elements, and modifying transition probabilities using Noise Injection (Fig. 6.2 and Table. 6.1). Here we outline these variations.

Domains

We implemented all three ATARI games from scratch, building on the Berkeley PacMan Projects[87]. PacMan was modelled as an MDP characterized by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \lambda)$.

State (s) and State Space ($\mathcal{S}$): We represented a grid of size $M \times N$ as a matrix of the same shape with the entries corresponding to the game element occupying the position in the grid—p (PacMan), g (Ghost), f (Food), w (Wall), or e (Empty). The state space $\mathcal{S}$ refers to the set of all possible states.

Action Space ($\mathcal{A}(s)$): Set of legal actions PacMan could take in state s. PacMan can move Left, Right, Up, or Down but not enter walls. Thus, when the PacMan is at the top left position the set of legal actions was only {Right, Down}.

Transition Matrix ($\mathcal{T}(s_i, a, s_j)$): Probability of moving to state s_j if the agent took action a at state s_i (Fig. 6.2a).

Reward Function ($\mathcal{R}(s)$): PacMan received +20 for eating a food pellet, -1 for every time step, -200 when it was killed, and +500 for finishing the game. [54].

Game Stochasticity: The motion of PacMan is deterministic—a Left action (if legal) will ensure that PacMan moves left. However, ghosts move stochastically according to a pre-fixed distribution. For instance, a RandomGhost moves in all directions with equal probability (accounting for walls). Thus, the game is non-deterministic.

MDPs for Pong and Breakout are defined analogously. For additional details, please refer to Supplementary Section Domains.

Noise Injection Variation: Generating new, controlled environments

We generate controlled variations of an original MDP by explicitly computing its Transition Function and then adding sampled noise to it.

Explicit enumeration of all states: States are defined by the position of the game elements. The probability of transitioning from a state to another is computed by multiplying the probability of each game element reaching the final configuration independently. Therefore, we visualize the game as a tree, each state is a node and the edges represent the transition probabilities of the game elements independently reaching their final configuration. By rolling out all possible moves by each game elements at each step, we enumerate all possible reachable states.

Explicit computation of Transition Function: Once we have all possible states, we can calculate the transition function, denoted as $\mathcal{T}(s_i, a, s_j)$. This function is determined by calculating the probability of each game character moving from one state to another independently.

Creating $\partial-$environments: We introduce variations in the game environment by modifying the transition function to $\mathcal{T}_\partial = \mathcal{T} + \partial$. Here, ∂ is a variable that follows a normal distribution, randomly chosen before each game to add unpredictability (Fig. 6.2c). The modified transition function, $\mathcal{T}_\partial$, is then adjusted to make sure the total probability of moving from any state s_i using action a to any other state s_j sums to 1.

$$\mathcal{T}_\delta(s_j, a, s_i) = \frac{|\mathcal{S}|p_{i,j} + \delta_{i,j}}{|\mathcal{S}| + \sum_j \delta_{i,j}} \tag{6.4}$$

$|\mathcal{S}|$ denotes the number of states, and guarantees the probability of legal successors does not approach 0 as the state space grows. We investigated two settings—(i) Low-Noise with $\delta \sim \mathcal{N}(0, 0.1)$, where some non-standard transitions previously impossible without noise are now possible with a low probability. (ii) High-Noise with $\delta \sim \mathcal{N}(0, 0.5)$, where non-standard transitions are possible with higher probability.

6.3 Experimental Details

We compared the mean reward curve of Learnability and Generalization agents. An agent $\mathcal{G}_T$ is said to generalize well with respect to $\mathcal{M}_\delta$, if its mean reward is as good as the corresponding Learnability agent $\mathcal{L}_\delta$.

Agents are trained with both tabular Q-Learning[402] and SARSA Q-learning[188], using Boltzmann or ε-greedy exploration strategies. In particular, we trained agents for $1,000$ episodes and averaged results over 500 trained agents. After every 10 training episodes, agents were evaluated using 10 testing episodes. We report the mean reward curves at convergence. Hyperparameters were inherited from past work[54] and are available in the Supplement in Sec. Training Parameters. The experiments were conducted on a system with an Intel(R) Xeon(R) CPU E5-2683 v4 @ 2.10GHz.

6.4 Results

We report findings from the Generalization and Learnability agents trained with the multiple variations of PacMan, Pong, and Breakout as described in Sec. Generating MDPs for investigating generalization and Table 6.1.

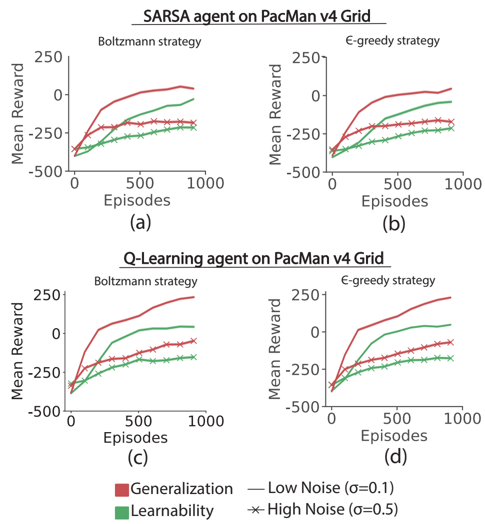


Figure 6.3: Generalization agents can outperform Learnability agents. Results for PacMan v4 grid reporting mean reward as a function of episode number. (a) SARSA agent trained with a Boltzmann exploration strategy for Target MDPs generated with both high (solid line) and low (line with 'x' markers) level noise injection. The Generalization Agent (red) beats the Learnability Agent (green) (two-sided t-test, p<0.001). (b) The same result holds for a SARSA agent trained with the $\varepsilon-$greedy exploration strategy. This finding also holds for Q-Learning agents trained with (c) Boltzmann and (d) $\varepsilon-$greedy exploration strategies. Standard deviation across the 500 agents is reported as the error bar in all figures. However, the standard deviation is too small for these error bars to be visible.

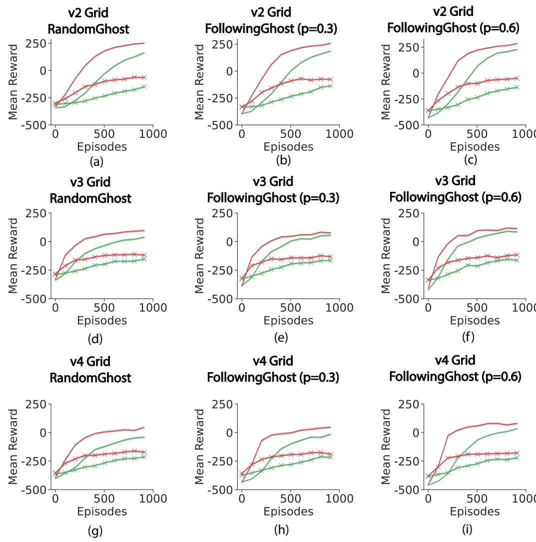


Figure 6.4: Generalization can outperform Learnability across multiple variations of PacMan. Format and conventions as in Fig. 6.3. (a) Agents trained on the PacMan v2 grid with the Ghost dynamics set to the RandomGhost setting. (b) Agents trained on v2 with a DirectionalGhost with $p = 0.3$. (c) DirectionalGhost with $p = 0.6$. (d),(e),(f) Variations with the v3 grid with RandomGhost, DirectionGhost ($p = 0.3$) and DirectionalGhost ($p = 0.6$), respectively. All experiments are shown for SARSA agents trained with the $\epsilon-$greedy exploration strategy. Generalization agents consistently beat Learnability Agents (two-sided t-test, p<0.001). Corresponding results for agents trained with SARSA + Boltzmann exploration strategy, and for Q-Learning with both $\epsilon-$greedy and Boltzmann exploration strategies are shown in Figures Supp.3- Supp.5.

Generalization agents can outperform Learnability agents in several instances of the Indoor-Training Effect

The mean reward increased with training, as expected, (Fig. 6.3a), both for the Generalization agent (red) and for the Learnability agent (green). Also, as intuitively expected, both agents performed better under low-noise conditions (solid lines) compared to high-noise conditions (lines with 'x' markers). Less intuitive was the relationship between Generalization and Learnability agents. Intriguingly, the Generalization agent consistently outperformed the Learnability agent

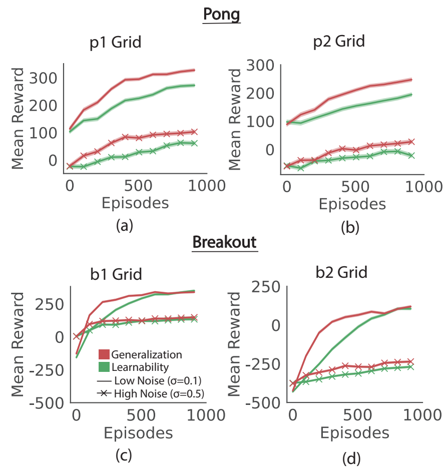


Figure 6.5: Generalization agents outperform Learnability agents on Pong and Breakout as well. Format and conventions as in Fig. 6.3. Performance of SARSA agents trained with an $\varepsilon-$greedy exploration strategy on (a) Pong p1 grid, (b) Pong p2 grid, (c) Breakout b1 grid, and (d) Breakout b2 grid. The Generalization Agent consistently beats the Learnability Agent (two-sided t-test, $p<0.001$).

(two-sided t-test, $p<0.001$). This gap continued until convergence at $1,000$ episodes, was observed both across low and high noise levels (solid lines versus 'x' lines), when using a Boltzmann strategy (Fig. 6.3a, c) or an $\varepsilon-$greedy strategy (Fig. 6.3b, d), and when using SARSA agents (Fig. 6.3a, b) or Q-Learning agents (Fig. 6.3c, d)

To assess whether this observation was dependent on the target MDP, we replicated these findings on multiple PacMan grids and noise variations (Fig. 6.4a-f). In (Fig. 6.4), the Generalization agents beat the Learnability agents, for both low and high levels of noise (see Supp. Sec. Additional Graphs Non-Semantic Variations: Figs. Supp.3- Supp.5 for Boltzmann strategy and Q-learning results).

We also extended these findings to two additional ATARI games, Pong Fig. Supp.1

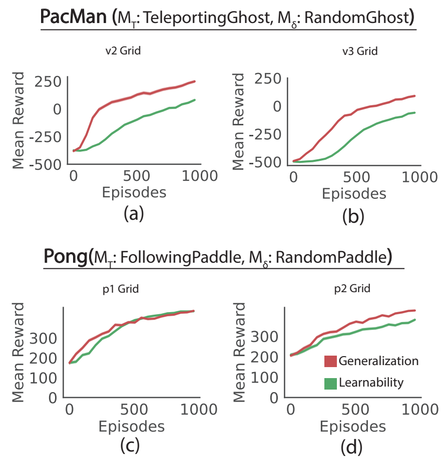


Figure 6.6: Generalization agents outperform Learnability agents on semantic variations of PacMan and Pong as well. Format and conventions as in Fig. 6.3. (a) Given the target PacMan MDP with the v2 grid and TeleportingGhost, the Generalization trained on the RandomGhost outperformed the Learnability agent that was trained and tested on the same Target MDP (TeleportingGhost) (two-sided t-test, $p<0.001$). (b) This finding extends to TeleportingGhost and RandomGhost MDPs with the PacMan v3 Grid as well. (c) For the Pong p1 grid, Generalization agents trained on an MDP with DirectionalPaddle performed better on the RandomPaddle MDP during testing, as compared to the Learnability Agent trained and tested on the RandomPaddle MDP. (d) The same finding extends to the p2 grid as well.

and Breakout Fig. Supp.2, to assess their applicability across different games (Fig. 6.5). Consistent with the results described for Pacman, the Generalization agent was on par with or better than the Learnability agent in Pong Fig. 6.5a,b) and Breakout Fig. 6.5c,d) (two-sided t-test, $p<0.001$; see Figs. Supp.6- Supp.12 for results with Q-Learning, Sarsa and different sampling strategies).

In sum, there exist several MDPs where it is better to train on a different MDP than the target. These results provide novel intriguing evidence suggesting that training on a different MDP can enable more efficient policy learning than training on the target environment.

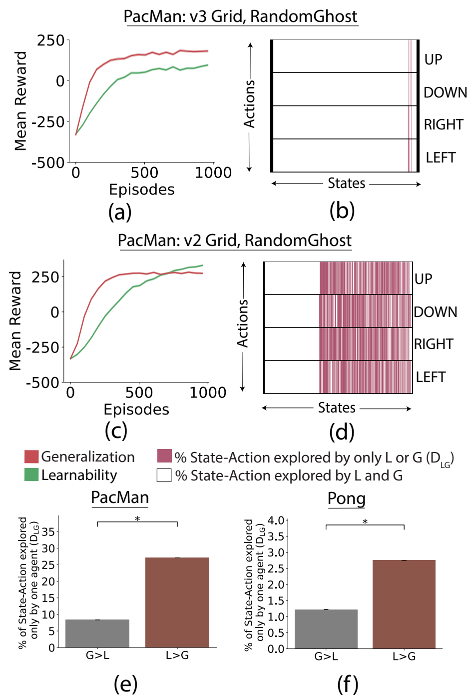


Figure 6.7: The exploration patterns predict the reward gap between $\mathcal{L}_{\partial}$ and $\mathcal{G}_T$. (a) Reward for agents trained on Pacman v3, where $\mathcal{G}_T$ outperforms $\mathcal{L}_{\partial}$ (format as in Fig. 6.3). (b) Exploration grid visualizing the difference in State-Action ($\mathcal{S}$-$\mathcal{A}$) pairs explored by these agents (D_{LG}). The grid shows Sates on the x-axis and Actions on the y-axis. The black lines separate the Actions for clarity. Each cell corresponds to one $\mathcal{S}$-$\mathcal{A}$ pair. In this case, a negligible fraction of $\mathcal{S}$-$\mathcal{A}$ pairs were visited only by one agent (pink). (c) Rewards for agents trained on PacMan v3. Here, $\mathcal{G}_T$ performs worse than $\mathcal{L}_{\partial}$ at the end of training. (d) A large fraction of pairs were only visited by either one or the other agent but not both (contrast with part (b)). (e) D_{LG} averaged over PacMan grids where $\mathcal{G}_T$ outperformed $\mathcal{L}_{\partial}$ (gray) and vice-versa (brown) (f) D_{LG} averaged over Pong grids. The "*" is for statistical significance (two-sided t-test, p<0.001).

Instances of the Indoor-Training Effect in semantic variations of ATARI games

The results presented so far focused on altered MDPs generated by noise injection. Next, we evaluated semantic variations, where the changes are more meaningful and interpretable. Specifically, we modified the transition probabilities of Pacman so that ghosts could teleport to new locations, and Pong so that the paddle could follow the ball. We refer to these alternate semantic environments as $\mathcal{M}_{T'}$ (semantic noise), in contrast to $\mathcal{M}_{\eth}$ used for noise injection.

For PacMan, Learnability agents were trained and tested using Teleporting-Ghosts ($\mathcal{M}_{T'}$), while the Generalization agents were trained with PacMan with RandomGhosts ($\mathcal{M}_{T}$) and then tested on TeleportingGhosts ($\mathcal{M}_{T'}$) (Fig. 6.6a, b, Supp. Sec. Additional Graphs Non-Semantic Variations). Even under these semantic noise conditions, Generalization agents outperformed Learnability agents (two-sided t-test, p<0.001; see Figs. Supp.13-Supp.15 for results with Q-Learning, Sarsa and different sampling strategies). In the case of Pong, we report analogous results with $\mathcal{M}_{T'}$ set to FollowingPaddle, and $\mathcal{M}_{T}$ set to RandomPaddle. Generalization agents also outperformed Learnability agents (by a smaller margin) in both the p1 and p2 grids (Fig. 6.6c,d). Analogous results for Pong with Q-Learning and other exploration strategies are reported in Figs. Supp.16- Supp.23.

The exploration patterns of state-action pairs can predict differences between Generalization and Learning agents

To better understand how Generalization agents could outperform Learnability agents, we investigated the exploration patterns for $\mathcal{L}_{\eth}$ and $\mathcal{G}_{T}$. We enumerated all State ($\mathcal{S}$)-Action ($\mathcal{A}$) Pairs, and divided them into three groups—(i) Percentage of $\mathcal{S}$-$\mathcal{A}$ pairs explored by both agents (P_{LG}), (ii) Percentage of pairs explored only by the Learnability agent (P_{L}), and (iii) Percentage of pairs explored only by the Generalization agent (P_{G}). Thus, $P_{LG} + P_{L} + P_{G} = 100$. We defined $D_{LG} = P_{L} + P_{G}$, the divergence in the exploration patterns between these two agents.

In Fig. 6.7 we visualize D_{LG} for grids where $\mathcal{G}_T$ outperformed $\mathcal{L}_\partial$ agents and compare it to cases where it did not. Fig. 6.7a shows an agent trained with Q-Learning and Boltzmann exploration strategy for the Pacman v3 grid with RandomGhost stochasticity, where $\mathcal{G}_T$ beat the $\mathcal{L}_\partial$ agent. The corresponding panel Fig. 6.7b depicts D_{LG} —each entry of this grid represents an $\mathcal{S}$-$\mathcal{A}$ pair. We refer to this plot as the exploration grid for these agents. The exploration grid shows that most $\mathcal{S}$-$\mathcal{A}$ pairs were explored by both agents, with almost no pairs explored only by one type of agent and therefore no significant differences in their exploration patterns. In contrast, Fig. 6.7c, d report $\mathcal{S}$-$\mathcal{A}$ pairs for PacMan v2, here the $\mathcal{G}_T$ agent performs worse than the $\mathcal{L}_\partial$ agent. The exploration grid reveals that there is a high fraction of $\mathcal{S}$-$\mathcal{A}$ pairs explored either by one or the other agent but not both.

We grouped all the cases where $\mathcal{L}_\partial > \mathcal{G}_T$ (Fig. 6.7e, brown) and all the cases where $\mathcal{L}_\partial < \mathcal{G}_T$ (Fig. 6.7e, gray) and computed D_{LG}. On average, D_{LG} was significantly higher in MDPs where $\mathcal{L}_\partial > \mathcal{G}_T$ (two-sided t-test, $p < 0.05$). The same result holds true for Pong MDPs as reported in Fig. 6.7f. Exploration grids and additional results for variations of PacMan (Supp.24-Supp.31), Pong (Supp.32-Supp.39), and Breakout (Supp.40-Supp.43) can be found in the Supplement in Sec. Additional Graphs State-Action Pairs. Instead of grouping MDPs, we also conducted a correlation analysis. We defined the Reward Gap: $R_{LG} = R_G - R_L$. The Spearman correlation coefficient between D_{LG} and R_{LG} was 0.43 ($p < 0.005$) for PacMan and 0.26 ($p < 0.005$) for Pong. Combined, these analyses show that the Indoor-Training Effect is associated with similar exploration patterns in the training and testing environments.

Discussion

In this work, we aim to understand the paradoxical Indoor-Training Effect—where agents perform better when trained in a noise-free environment and tested in noisy δ-environments, compared to being trained and tested in the same δ-environments. Similar to how training in a calm, noise-free indoor environment helps athletes focus on mastering the fundamentals of tennis, we explore whether training on certain environments is more conducive to learning than training on the same testing environment.

To investigate this, we propose a new methodology to generate modified MDPs from a given MDP, along with a metric to quantify the distance between different environments. We demonstrate the Indoor-Training Effect across various algorithms and exploration strategies (Fig. 6.3), grid layouts and game stochasticity (Fig. 6.4), and multiple ATARI games (Fig. 6.5). We also showed that this phenomenon extends beyond Noise Injected environments, and can also occur when semantic changes are introduced in the game elements (Fig. 6.6).

To gain deeper insights into these environments, we examine the exploration patterns of agents under different transition probabilities. Similarly to a tennis player who has never encountered a smash serve during their training and develops an optimal playing style that does not anticipate or respond to such powerful shots, the suboptimal performance of the agents could be caused by a divergence in exploration patterns. We show that the performance gap between agents is indeed correlated with their exploration patterns under different transition probabilities (Fig. 6.7).

The Indoor-Training Effect is particularly relevant to robotics, where robots often operate in complex, dynamic environments. The Indoor-Training Effect opens new avenues of research, whereby robotic systems could be trained in simplified, controlled settings to master essential skills without the interference of noise. This finding could also enhance their ability to adapt and perform in real-world conditions where unpredictability and noise are prevalent. Such training strategies could lead to more robust, adaptable robots capable of navigating and executing tasks effectively in diverse and challenging environments.

We note that these findings are reminiscent of results with biological agents. For example, recent experiments with the C. Elegans worm have shown that biological agents perform best when cross-trained on different environments as compared to being tested on the environments they were trained on[218].

Despite the evidence provided in this study, we would like to highlight two main limitations. Firstly, our experiments were conducted solely in the context of ATARI games. We hope that future research can extend and examine the findings in real-world environments. Secondly, it will be interesting to assess whether the conclusions drawn from classical Reinforcement Learning methods extend to deep RL approaches.

These findings raise fundamental questions about our understanding of RL algorithms. Typically, RL practitioners have strived to train agents in environments that closely resemble their deployment conditions. This approach assumes that matching the training and testing environments is critical for optimal performance. However, the Indoor-Training Effect challenges this assumption by showing that agents trained in noise-free, controlled environments can sometimes outperform those trained in more chaotic, realistic settings when faced with noisy, unpredictable scenarios during testing.

I must begin with the most simple and general things, and ascend little by little to the most particular and complex.

Meditations on First Philosophy, Rene Descartes

7 Are these machines even safe inside the distribution?

7.1 Introduction

Understanding the mechanisms enabling adversarial attacks on deep neural networks remains an open and elusive problem in machine learning. Despite

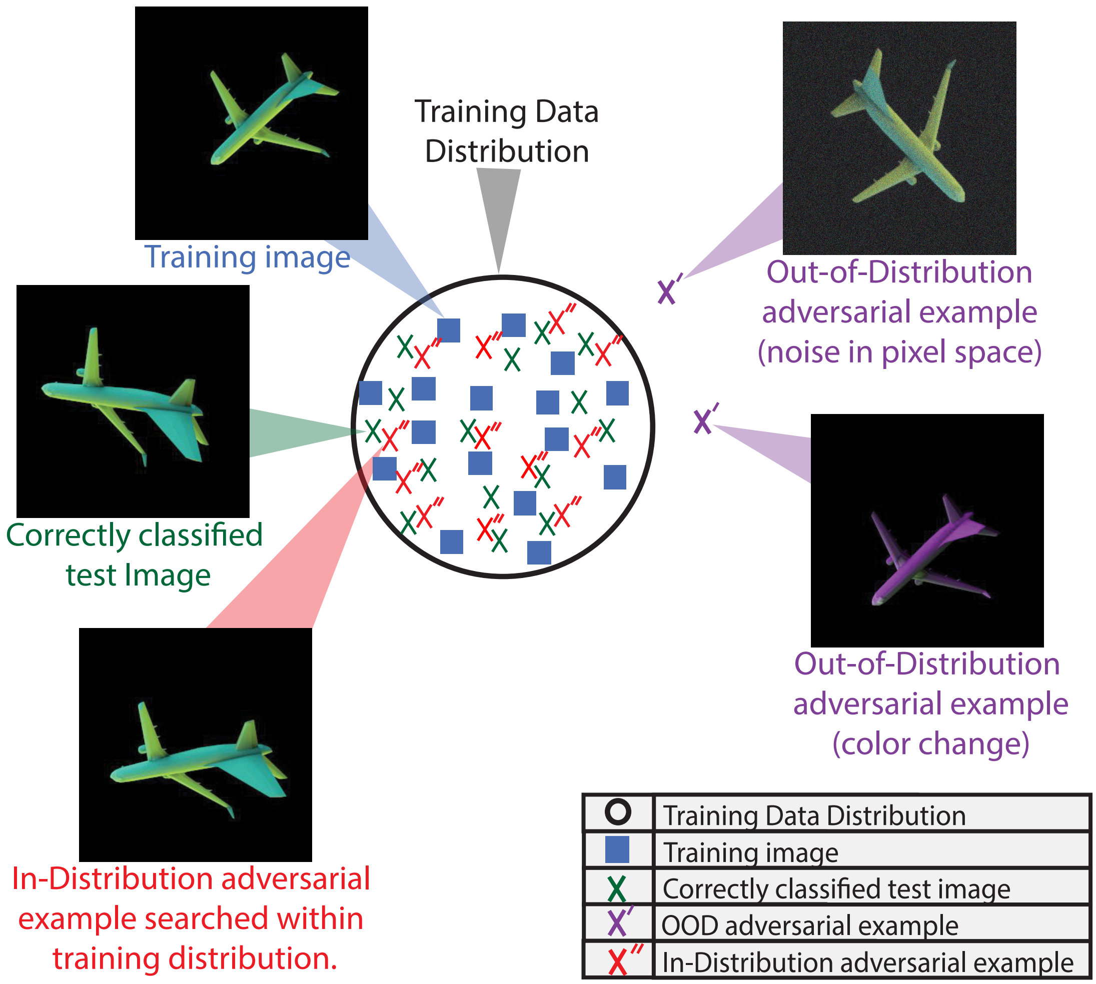


Figure 7.1: In-distribution adversarial examples. This schematic highlights the difference between typical (out-of-distribution) and in-distribution adversarial examples. We train object recognition models on a large scale training data of 0.5 million images sampled from known, parametric distribution of camera and light variations (depicted using ■). Despite great success in correctly classifying newly sampled test points from the training data distribution (X), our CMA-Search method shows that it is possible to find plenty of adversarial examples which lie within the training distribution (X''). Unlike existing methods that add noise to the image resulting in out-of-distribution adversarial examples (X''), CMA-Search searches within the training distribution to find adversarial examples. We find a widespread presence of in-distribution adversarial examples for object recognition.

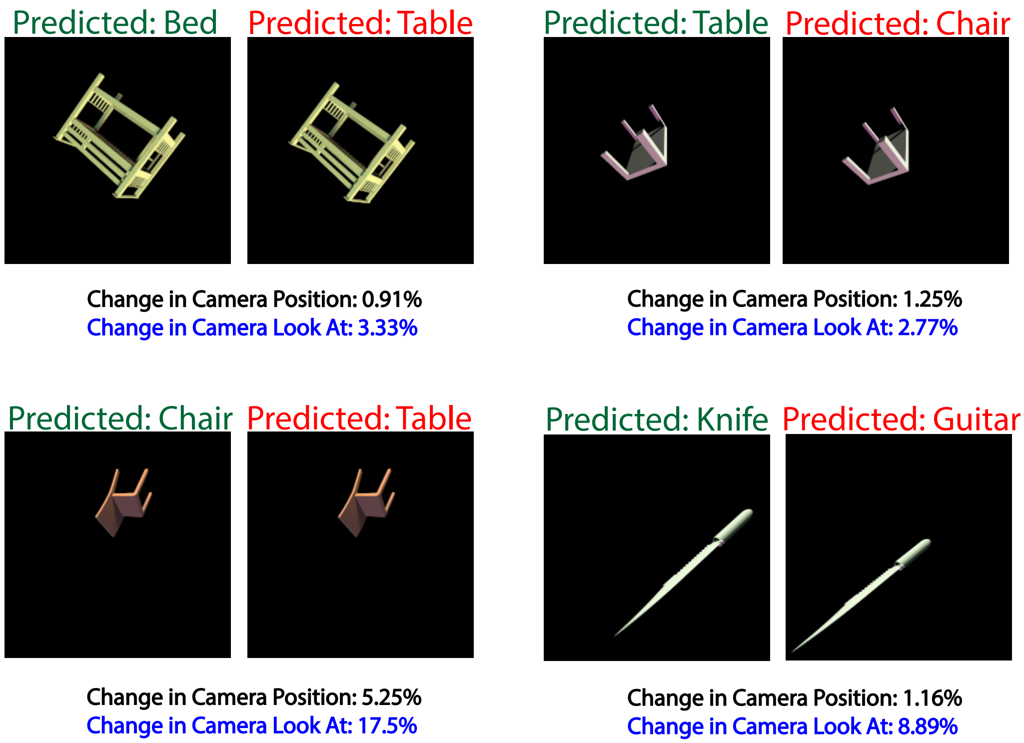


Figure 7.2: Sample in-distribution adversarial examples identified using CMA-Search with camera parameters. Starting with the correctly predicted images, our evolutionary-strategy based method (CMA-Search) explores the vicinity of camera parameters for subtle 3D perspective changes that lead to misclassification. These in-distribution adversarial examples are often found very close to the correctly classified starting image. In the figure we report the percentage of change in Camera Position and Camera Look At parameters necessary to induce the misclassification.

a plethora of works attempting to explain these attacks, posited theories are largely disconnected and focus on specific considerations such as attributing adversarial attacks to the tilting[362] or the curvature [112] of the learned decision boundary, the dimension of the data manifold[329,328], data distribution shifts[114], the presence of non-robust features[177,379], lack of data[321], and computational complexity[44,269], among others. It remains unclear which of these theories are more valid in practice and can fully explain adversarial attacks on object recognition models.

Adversarial examples are usually defined as perturbed inputs which cause classification networks to make an error. However, there is no constraint enforced on the resulting adversarial example's position with respect to the training data distribution. Recently, a strand of theoretical works have provided compelling evidence for adversarial examples that lie within the training distribution[124,111,325,248,95,98]. Such in-distribution examples provide a more stringent definition of adversarial examples than typically used in the theories mentioned above by enforcing that the resulting examples lie within the training data distribution. This newly discovered phenomenon of in-distribution adversarial examples presents an opportunity to compare different theories, and to validate which ones can explain these findings. Furthermore, such examples are highly concerning as unlike typical adversarial examples these do not require adding synthetically engineered perturbations by an external malicious agent since these examples are already within the training data distribution.

The key insight at the heart of these theoretical works is that most data points are close to the ground-truth class boundaries, for high-dimensional data distributions. Thus, slight deviations between the learned and the ground-truth class boundaries causes in-distribution adversarial examples given the proximity of these points to these boundaries. For brevity, we refer to this theory as the ground-truth boundary theory.

While this theory demonstrates that the phenomenon of adversarial examples runs far deeper than the added perturbations resulting in samples from outside the data distribution, works investigating the ground-truth boundary theory make strong simplifying assumptions about the data distributions which may not hold true in practice. This includes assuming that the data is generated from a smooth generative model[111], belongs to a Levy family[248], satisfies the W_2 Talagrand transportation-cost inequality, lies on a uniform hypercube[325,95], or on disjoint concentric shells[124]. As noted in these papers, it is unclear whether these theoretical findings are relevant in practice, as these assumptions may not be true for images of real-world objects.

Here, we seek to reconcile this disconnect by asking whether this theory extends to image data for object recognition. A positive answer would provide compelling evidence in support of the ground-truth boundary theory being the primary mechanism driving adversarial examples.

To find in-distribution adversarial attacks, we introduce an evolution-strategy based search method that we call CMA-Search. Most existing adversarial attack methods rely on derivative-based search which suffer from two problems. Firstly, they cannot efficiently find in-distribution adversarial examples at low dimensions[124]. Secondly, it is unclear if the adversarial sample found after noise addition still belongs to the training distribution. In contrast, starting with a correctly classified input CMA-Search searches the vicinity of input parameters to find in-distribution adversarial examples. To reflect this, we chose to name our approach an adversarial-search method, as opposed to an adversarial attack method. Inspired by recent works using computer graphics to create controlled datasets for investigating neural networks[241,387,265,433], we introduce a procedural computer graphics pipeline. Our pipeline allows us to generate a large-scale dataset with complete control over camera and lighting variations. This offers us explicit, parametric control over the training distribution similar to theoretical works, enabling us to investigate in-distribution adversarial examples with complex images of objects.

These experiments lead us to our key finding—there is a widespread presence of adversarial images within the training distribution, as summarized in Fig. 7.1. To foreshadow our results, CMA-Search can find such an adversarial example for over 71% cases with an average change of only 1.83% in the camera position, in 42% cases with an average change of only 6.52% in the lighting conditions. These examples are depicted in Fig. 7.2. Finally, we also extend our method in conjunction with a novel view synthesis pipeline[380] to find adversarial examples in the vicinity of ImageNet[88] images for a ResNet model and the recently released OpenAI CLIP model[301]. These results on in-distribution adversarial examples provide compelling evidence in support of theories attributing adversarial attacks to the proximity of data to ground-truth class boundaries, and presents an opportunity to help refine existing theories through a more stringent definition of adversarial attacks.

7.2 Results on in-distribution robustness

We present results on in-distribution robustness identified using our proposed approach (CMA-Search) across three levels of data complexity—(i) simplistic parametrically controlled data sampled from disjoint per-category uniform distributions (Sec. 7.2.1), (ii) parametric and controlled images of objects using our graphics pipeline (Sec. 7.2.2), and (iii) natural image data from the ImageNet dataset[88](Sec. 7.2.3).

7.2.1 Ground-truth boundary theory explains in-distribution adversarial examples in simplistic parametrically controlled data

We build on the same setup as previous work[124]—binary classification of data sampled from two high-dimensional, disjoint uniform distributions (see methods Sec. 7.4.1). This previous work relied on Projected Gradient Descent to find adversarial examples[111,355], but this approach only works at high dimensions (> 60)[124]. We present results using our evolutionary strategies based CMA-Search, as it can also find in-distribution adversarial examples in low dimensions as shown in the following. More details on the implementation of CMA-Search are provided in methods Sec. 7.4.5.

In Fig. 7.3(d), we report our method's attack rate for models with high accuracy (> 0.99). The attack rate measures the fraction of correctly classified points for which an in-distribution adversarial examples can be found in the vicinity using CMA-Search (see methods Sec. 7.4.6 for more details). These results demonstrate that despite a near perfect accuracy on a held-out, randomly sampled test set, in-distribution adversarial examples can be identified in the vicinity of all correctly classified test points using CMA-Search. This simplistic dataset is easily separable by most conventional machine learning models including a decision tree, which makes the presence of in-distribution adversarial examples both surprising and highly concerning.

In Fig. 7.3(a) we report the attack rate for models trained with 20, 100 and 500 dimensional data. As can be seen, the attack rate of CMA-Search is nearly 1.0 till a very large dataset size. However, once a critical dataset size is reached, networks do start becoming robust. Unfortunately, the data complexity scales poorly with number of dimensions. While dimensionality grows five-fold from 20 to 100, the number of points required for robustness scales almost 100-fold. Furthermore, for 500 dimensions we were unable to identify the dataset size required for models to become robust despite trying 10 million training data points.

In Fig. 7.3(b) we report the average distance between the (correctly classified) start point and the in-distribution adversarial example. As can be seen, this distance increases as dataset size is increased. As critical dataset size is reached, adversarial examples are far enough from starting points that they are now not in-distribution. This results in a dip in the attack rate, as we only measure in-distribution adversarial examples. This suggests that for a fixed data dimensionality sample complexity does have a significant impact on in-distribution robustness.

Table 7.1: Role of stochasticity in in-distribution robustness. To isolate the source of high variance in model robustness at high dataset sizes, we investigate four sources of stochasticity one at a time holding other factors constant. These include—inherent stochasticity of CMA-Search, SGD, sampling bias, and model initialization. Our results show that this variance is largely driven by model robustness, which has substantial impact on model robustness.

Stochastic source varied	Attack Rate
CMA-Search	0.14 ± 0.16
SGD	0.22 ± 0.09
Sampling Bias	0.08 ± 0.06
Model Initialization	0.99 ± 0.03

We also investigated the role of robust training on in-distribution robustness by including $20,000$ identified adversarial examples alongside $100,000$ training data points and retraining the model for the 100 dimensional case. We found that the attack rate continued to be 1.0, with no improvement in model robustness against CMA-Search. This is expected behaviour, as our identified adversarial examples lie within the training distribution, and robust training in this case essentially amounts to a marginal increase in the training dataset size.

While models start becoming robust at high dataset sizes (Fig. 7.3(a),(b)), we found significant variance in this behaviour. Empirically, we found that the attack rates of models trained at high dataset sizes can also be high at times. This variance is also visible in the error bars in Fig. 7.3(a),(b). This result suggests that despite the same problem setup some models turn out to be robust, while others do not. To identify the underlying cause of this stochasticity in model robustness, we investigate how robustness changes as a function of four sources of stochasticity—inherent stochasticity of CMA-Search, optimization (SGD), sampling bias, and model initialization. For this analysis, we started by first identifying a robust model trained with $100,000$ points for 20-dimensional data and then attacked the model again holding everything constant while varying one source of stochasticity at a time.

Results are reported in Table 7.1. The mean attack rate remains low across multiple CMA-Search repetitions (0.14), and across multiple models trained from scratch (0.22). Thus, robustness is not due to the inherent stochasticity of CMA-Search, or SGD. Furthermore, new models trained with newly sampled data also resulted in a robust model with a low attack rate of 0.08, suggesting there is no good dataset which led to the models becoming robust. Interestingly, we find that models trained with new initializations are now non-robust and have a high attack rate of 0.99. Thus, what makes certain models robust and others non-robust depends on the model initialization. More details on these experiments are provided in Sec. 7.4.6.

In summary, results in Fig. 7.3 and Table 7.1 together suggest that while there is widespread presence of adversarial examples within the training distribution, models can start becoming robust at critical (extremely large) dataset sizes. Furthermore, the deciding factor for which models will be robust at extreme sizes is strongly dependent on the model initialization.

Despite evidence supporting the presence of adversarial examples lying within the training distribution, the mechanisms driving such examples remain unknown. There are two main potential hypotheses that can explain such in-distribution adversarial examples. Fig. 7.3(c) depicts the ground truth function as a one dimensional binary step function for ease of visualization. Firstly, adversarial examples could be an outcome of the learned function being poorly regularized. We call this hypothesis under-regularized learned boundary, and depict it in Fig. 7.3(d) in one dimension. As can be seen, adversarial examples are spread across the entire range of inputs in this case. For such examples, better regularization can prevent 'spikes' in the predicted output and would lead to better in-distribution robustness. Secondly, in-distribution adversarial examples may be an outcome of the complexity of ground-truth boundaries at high dimensions. We call it the ground-truth boundary theory, and depict it in Fig. 7.3(e) in low dimensions for ease of comprehension. In this case, there are no high-frequency 'spikes' in the learned function. Instead, the learned function is simply wrong in estimating where the step function changes from 1 to 0 as the probability of sampling near the ground-truth boundary in the training data is infinitesimally small due to the training data being finite. In this case all errors are located in the vicinity of the function transition and better regularization would not help prevent these errors. As shown in previous works investigating this hypothesis[124,111,325,248,95,98], the number of function transitions increases combinatorially with dimensionality, making it easier to find an in-distribution adversarial example in the vicinity of one of these transitions. Below, we assess these two hypotheses.

To investigate which of the two hypotheses presented in Fig. 7.3 hold true for this dataset, we visualize the learned decision boundary in the vicinity of the category transition using church window plots[401]. More details on how these are plotted is provided in methods Sec. 7.4.6. As can be seen in Fig. 7.3(f), there is a clean transition from correctly classified points (white) to in-distribution adversarial examples near the decision boundary (red), beyond which points become out of the distribution of samples belonging to a particular category (black). We observed this same behaviour across all church window plots made with multiple randomized samples and orthogonal vectors. In-distibution adversarial examples are isolated to a region close to the category boundary, and in a contiguous fashion. This strongly suggests that these in-distribution adversarial examples occur due to the mechanism presented in Fig. 7.3(e)—due to ground-truth boundary complexity in high dimensions, as opposed to poor regularization.

7.2.2 Widespread presence of in-distribution adversarial examples through subtle changes in 3D perspective and lighting

We extend our investigations to parametric and controlled images of objects using our graphics pipeline. We use a computer graphics pipeline for generating and modifying images which ensures complete parametric control over the data distribution. Every image generated from our pipeline can be completely described by their lighting and camera parameters shown in Fig 7.4(a). To create a dataset with a fixed, known training distribution, we simply sample camera and lighting parameters from a fixed, uniform distribution, and render a subset of 3D models from ShapeNet[58] objects with these camera and lighting parameters.

Using this approach, we create a large-scale (~ 0.5 million images) and unbiased dataset of complex image data with a fixed, known distribution. Furthermore, sampling new points from this distribution is as simple as sampling more camera and light parameters from their known distribution. Our pipeline builds upon recent work by Li et al.[228]. Fig. 7.4(b) shows sample images sampled from the known training distribution (additional examples from the dataset can be found in Fig. Supp.44). As can be seen, this dataset contains objects seen across multiple viewpoints, scales, and shifted across the frame. Furthermore, we use physically based rendering[228,290] to accurately simulate complex lighting artifacts including diverse lighting conditions like multiple colors and self-shadows which makes the dataset challenging for neural networks. We ensure that the following constraints are met: (1) uniformly distributed and unbiased training data, (2) 1000 images per 3D object (total 0.5 million images), and (3) no spurious correlations between the scene parameters and the image labels. More details on dataset generation can be found in methods Sec. 7.4.2.

We first investigate how well object recognition models perform across camera and lighting variations while ensuring an exact match between training and testing distributions. Both the train and the test sets are created by sampling uniformly across the range of camera and lighting parameters for this experiment. We evaluate both Convolutional Neural Networks (CNNs) and transformer-based models. Exact hyper-parameters and other details on model training are provided in methods Sec. 7.4.4

The difficulty in classifying these images is corroborated in Table 7.2, as there

Table 7.2: Performance of object recognition models on seen and new 3D models.

Accuracy	ResNet	ResNet (pre-trained)	Anti-Aliased Networks	Truly Shift Invariant	ViT	DeIT	DeIT Distilled
Seen models	0.75	0.76	0.82	0.80	0.58	0.63	0.64
New models	0.70	0.70	0.74	0.72	0.59	0.64	0.65

is significant room for improvement for both CNNs and transformer-based models. Table 7.2 reports accuracy for several state-of-the-art CNNs[162,440,55] and transformer architectures including the vision transformer (ViT)[99], and the data efficient transformer (DeIT) and its distilled version (DeIT Distilled)[377]. As can be seen, neural networks do not perform very well across camera and lighting variations. In fact, the problem is even more pronounced with transformer models. To test if this problem can be mitigated with shift-invariant architectures, we also report results on two specialized shift-invariant architectures - Anti-Aliased Networks[440], and the recent Truly Shift Invariant Network[55]. While these networks do provide a boost in performance, they too are susceptible to camera and lighting variations. We also confirm that this is not an outcome of our neural networks overfitting by testing the network on new, unseen 3D models. The performance on these new 3D models also mirrors the same trend, as seen in Table 7.2.

These results naturally raise the question—What images are these networks failing on? Are there certain lighting and camera conditions that the networks fail on? The one-to-one mapping between the pixel space (images) and our low-dimensional scene representation (i.e. camera and lighting parameters) allows us to answer these questions by visualizing and comparing correctly and incorrectly classified images in this low dimensional space. In Fig. 7.5 we show the distribution of camera parameters for images which were classified incorrectly. As can be seen, the errors seem well distributed across space—we found no clear, strong patterns which characterize the camera and light conditions of misclassified images. Note that regions in each of these parametric spaces represent human interpretable scenarios which have been known to impact human vision significantly. For instance, changes in camera position represent canonical vs non-canonical poses which significantly impact human vision[127,368,35]. Similarly, changes in the up vector can represent upside-down objects which too impact human vision[201,216,369]. In contrast, Fig. 7.5 shows that networks do not suffer in specific regions of the space. These results are consistent across multiple architectures. Supplemental Fig. Supp.45 shows examples of this phenomenon multiple object categories and neural network architectures.

While above results prove the existence of adversarial examples within the training distribution, a key requirement for such examples is the imperceptibility of the change needed to introduce an error. To introduce such imperceptible changes, we propose an evolution-strategies based error search methodology for in-distribution, misclassified images which we call CMA-Search. Starting with a correctly classified image, our method searches within in the vicinity of the camera and lighting parameters to find an in-distribution image which is incorrectly classified. Note that unlike adversarial attacks, our method does not add noise and our constraints ensure that identified errors are in-distribution. CMA-search enables interpretable attacks by searching over the scene's camera and

Table 7.3: CMA-Search over camera and light parameters. Starting with new, correctly classified in-distribution images, we use our method to search the vicinity of camera and light positions, starting with the original image's parameters. Attack Rate reports the percentage of times an in-distribution adversarial example was found starting with a correctly classified image. We also report the mean and standard deviation of the distance between the original image and the identified in-distribution adversarial example. This distance is measured by calculating the L2 distance between the camera parameters of the original correctly classified image and the parameters of the in-distribution adversarial example in its vicinity, and normalizing it by the range of the camera parameters.

Model Architecture	CMA Cam		CMA Light	
	Attack Rate (%)	Distance (mean ± std)	Attack Rate (%)	Distance (mean ± std)
ResNet18[162]	71	1.83 ± 1.33	42	6.52 ± 5.68
ResNet18 (pretrained)[162]	58	1.79 ± 1.46	36	5.36 ± 3.70
Anti-Aliased Networks[440]	45	2.32 ± 2.09	40	7.03 ± 5.10
Truly Shift Invariant Network[55]	53	2.22 ± 2.16	25	6.72 ± 5.41
ViT[99]	85	1.34 ± 1.16	65	4.63 ± 3.49
DeIT[377]	85	1.27 ± 0.81	51	4.54 ± 2.75
DeIT Distilled[377]	86	1.22 ± 0.87	55	4.49 ± 2.27

light parameters, while only sampling from within the training distribution. For instance, it is possible to attack a model by searching over solely the camera position (3 dimensions), while holding all other scene parameters constant. While previous works have attempted to find in-distribution adversarial examples by approximating the data distribution using generative models[355,111], we provide first empirical evidence for in-distribution adversarial examples in object recognition.

As shown in Table 7.3, CMA-Search finds small changes in 3D perspective and lighting which have a drastic impact on network performance. For example, starting with an image correctly classified by a ResNet18[162] model, our method can find an error in its vicinity for 71% cases with an average change of 1.83% in the camera position. For transformers, the impact is far worse, with an attack rate of 85%. Similarly, with lighting changes CMA-Search can find a misclassification in 42% cases with an average change of 6.52% for a ResNet18 model. These results are reported for various architectures in Table 7.3. As can be seen, we find that networks are most sensitive to changes in the Camera Position and the camera Look At—subtle, in-distribution 3D perspective changes. This behaviour is consistent across several architectures. In fact, even shift-invariant architectures specifically designed to be robust to 2D shifts are still highly susceptible to 3D perspective changes.

Thus, the space of camera and lighting variations is filled with in-distribution adversarial examples in the vicinity of correctly classified points. Unfortunately, church-window plots similar to Fig. ?? cannot be replicated for complex image data as the transition boundary between two object categories is hard to define. However, our results provide evidence in support of the ground-truth boundary theory as an explanation for adversarial examples in object recognition, and calls into question existing theories which attribute adversarial examples to systematic differences in the training and test distributions.

7.2.3 Approximate in-distribution adversarial examples in the vicinity of natural images

So far, our experiments have focused on datasets with complete control over the training distribution. Here, we present results on approximate in-distribution adversarial examples for natural images taken from the ImageNet dataset[88].

Specifically, we used CMA-Search to optimize the camera parameters, but instead of our renderer, we now use a novel view synthesis model (MPI[380]) for generating novel views of ImageNet images. More details on this procedure can be found in methods Sec. 7.4.3.

Starting with a correctly predicted ImageNet image, we use CMA-Search in conjunction with the MPI model to find images in the vicinity with small, 3D perspective changes which can break ImageNet trained classification networks including ResNet18, and OpenAI's transformer based CLIP model[301]. Results for these experiments are reported in Fig. 7.6. We provide additional examples of misclassified ImageNet images found using CMA-Search in Supplementary Fig. Supp.46. MPI model was not trained on ImageNet, and can at times fail to generate novel views, resulting in blurry images instead. We omit these images to only present results on adversarial examples due to small, 3D perspective changes. While these results present an interesting application on natural images, these results are only approximately in-distribution like previous works that also explored in-distriubtion adversarial examples[355,111]. We cannot be entirely sure that the images found by CMA-Search on ImageNet are indeed in-distribution, further justifying the necessity of our computer graphics based approach in the previous section.

7.3 Discussion

Recent theoretical works developing the ground-truth boundary theory have investigated adversarial examples within the training data distribution. This added constraint requiring adversarial examples be in-distribution results in a more stringent definition of adversarial examples, which presents an opportunity to further scrutinize and update existing theories on the origin of adversarial examples. However, these works were restricted to simplistic, parametrically controlled data. Here we provided evidence that these in-distribution adversarial examples extend to images of objects. These results provide new, stronger evidence in support of the ground-truth boundary theory being the primary mechanism driving in-distribution adversarial examples.

In practice, the presence of these examples points to a highly worrisome problem—it bypasses the need for a malicious agent to add engineered noise to induce an error. These examples lie hidden within the data distribution in plain sight. In fact, our results show that the problem runs far deeper than previously thought as it is even possible to attack models trained on as low as 5-dimensional data.

We also show that the current best practices for adversarial defense are insufficient to address in-distribution adversarial examples. Most existing approaches revolve around the idea of robust training[246], i.e. including adversarial examples into the training set. As mentioned in Sec. 7.2.1, we confirmed that in-distribution errors could not be removed by robust training. This may be because finding adversarial examples is computationally costly and models start becoming robust only at extreme dataset sizes, and that this critical size increases exponentially with data dimensionality. This need for extreme dataset size can be explained by the ground-truth boundary theory. There is combinatorial increase in the number of ground-truth boundary transitions as data dimensionality increases, and thus a corresponding drop in the probability of a randomly sampled point being sufficiently close to the transition boundary as explained in Fig. 7.3. In this way, an increasingly large number of samples are needed to obtain samples from the boundary transitions as dimensionality increases. Recent work on scaling laws[22,191] has investigated accuracy at extreme dataset sizes, but our finding suggests similar scaling laws could also be identified for model robustness.

Based on our findings, we propose three potential directions which might help alleviate in-distribution adversarial examples. Firstly, reducing the data representation dimensionality. As dataset size needed for robustness scales poorly with dimensions, efficient dimensionality reduction on data representation may help reduce samples needed to train a robust model. Secondly, better model initialization. We showed that once critical dataset size is reached, model robustness strongly depends on initialization, and thus future researchers may need to devise better initialization techniques which result in more robust models. Thirdly, casting object recognition as a smooth regression problem that reduces the number of ground-truth boundary transitions as these examples are located only in the vicinity of these boundaries (Fig. 7.3).

In summary, we have provided empirical evidence of the widespread presence of in-distribution adversarial examples for complex image data, which is highly worrisome and has concerning ramifications for the origin of and defense against adversarial examples. Going forward, we hope that future researchers can combine theoretical and empirical investigations using the unified framework provided in our work, and help move the machine learning and computer vision communities closer to a deeper understanding of the phenomenon we call adversarial examples. Understanding these susceptibilities of object recognition models lies at the heart of building robust and reliably deployable models, and we hope this research direction can make a strong contribution towards this end.

7.4 Methods

7.4.1 Generating simplistic parametrically controlled data

We created a binary classification task by sampling data from two N-dimensional uniform distributions confined to disjoint ranges (a, b) and (c, d), as described in the following:

$$x_i \sim \left\{ \begin{array}{ll} \text{Unif}(a, b, N); & y_i = 0 \\ \text{Unif}(c, d, N); & y_i = 1 \end{array} \right\}. \tag{7.1}$$

We set $a = -10, b = 10, c = 20, d = 40$ for experiments presented in Sec. 7.2.1. However, we observed that the exact choice of these parameters does not impact our findings. To measure in-distribution performance, we simply sample new data points from these same distributions.

7.4.2 Generating controlled rendered data of real world objects

Most large-scale datasets for computer vision have been created by scraping pictures from the internet[88,233,110,203,445]. For experiments investigating in-distribution robustness, these datasets present two major challenges. Firstly, it is not possible to quantitatively define or control the distribution of these datasets in closed form. Secondly, investigating in-distribution robustness requires being able to sample new points from regions of interest within the data distribution, and to test model performance on these samples. This is not possible with internet scraped datasets. These problems have inspired the growing trend of research works using carefully designed synthetic data with controlled data distributions[299,166,241,38,213,397]. In a similar vein, our graphics pipeline (explained below) easily allows us to generate a large-scale, unbiased dataset of objects seen under varied camera and lighting conditions with complete control over the data distribution. Sample images from four categories are shown in Fig. 7.4(b). Each 3D model was rendered under 1000 different camera and lighting conditions following the scene setup described below. Fig. Supp.44 shows additional sample images from the dataset. Below we introduce the 3D scene setup, camera and lighting parameter sampling strategies, and the 3D models used to generate our dataset.

3D Scene Setup

All rendering was done using the open-source rendering pipeline Redner[228]. Each scene contains one camera, one 3D model and 1-4 lights. To ensure there are no spurious correlations between object category, texture and background[118], the texture for all ShapeNet[58] objects was replaced with a simple diffuse material and the background was kept constant. Thus, every scene is completely parametrized by the camera and the light parameters. As shown in Fig. 7.4(a), camera parameters are 10 dimensional: one dimension for the FOV (field of view of camera lens), and three dimensions each for the Camera Position (coordinates of camera center), Look At (point on the canvas where the camera looks), and the UP vector (rotation of camera). Analogously, lights are represented by 11 dimensions - two dimensions for the Light Size, and three each for Light Position, light Look At and RGB color intensity. Multiple lights ensure that scenes contain complex lighting scenarios including multiple colors and self-shadows. Thus, our scenes are $(11n + 10)$ dimensional, where n is the number of lights. There is a one-to-one mapping between the pixel space (rendered images) and this low dimensional scene representation.

Unbiased, uniform sampling of scene parameters

To ensure an unbiased distribution over different viewpoints, locations on the frame, perspective projections and colors, we ensured that scene parameters follow a uniform distribution. Concretely, camera and light positions were sampled from a uniform distribution on a spherical shell with a fixed minimum and maximum radius. The Up Vector was uniformly distributed across range of all possible camera rotations, and RGB light intensities were uniformly distributed across all possible colors. Camera and light Look At positions were uniformly distributed while ensuring the object stays in frame and is well-lit (frame size depends on Camera Position and FOV). Finally, Light Size and camera FOV were uniformly sampled 2D and 1D vectors. Below we specify the hyper-parameters for rendering, along with the exact distribution for each scene parameter and the corresponding sampling technique used to sample from these distributions.

Camera Position: For scene camera, first a random radius r_c is sampled while ensuring $r_c \sim \mathrm{Unif}(0.5, 8)$. Then, the camera is placed on a random point denoted (x_c, y_c, z_c) on the spherical shell of radius r_c. To generate a random point on the sphere while ensuring an equal probability of all points, we rely on the method which sums three randomly sampled normal distributions[153]:

$$X, Y, Z \sim \mathcal{N}(0, 1), \tag{7.2}$$

$$v = (X, Y, Z), \tag{7.3}$$

$$(x_c, y_c, z_c) = r_c * \frac{v}{\|v\|}. \tag{7.4}$$

Camera Look At: To ensure the object is shown at different locations within the camera frame, the camera Look At needs to be varied. However, range of values such that the object is visible can be present across the entire range of the frame depends on the camera position. So, we sample camera Look At as l_c as follows:

$$l_c \sim \mathrm{Unif}(K * x_c, K * y_c, K * z_c), \text{where } K = 0.3. \tag{7.5}$$

The value $K = 0.3$ was found empirically. We found it helped ensure that objects show up across the whole frame while still being completely visible within the frame.

Camera Up Vector: Note that the camera Up Vector is implemented as the vector joining the camera center (0,0,0) to a specified position. We sample this position and therefore the Up Vector u_c as follows:

$$x, y, z \sim \text{Unif}(-1, 1), \tag{7.6}$$

$$u_c = (x, y, z). \tag{7.7}$$

Camera Field of View (FOV): We sample the field of view f_c while ensuring:

$$f_c \sim \text{Unif}(K_1, K_2). \tag{7.8}$$

Again, the values $K_1 = 35, K_2 = 100$ were found empirically to ensure objects are completely visible within the frame while not being too small.

Light Position: For every scene we first sample the number of lights n between 1-4 with equal probability. For each light i, a random radius r_i is sampled ensuring $r_i \sim \text{Unif}(R_1, R_2)$, then the light is placed on a random point (x_i, y_i, z_i) on the sphere of radius r_i. $R_1 = 1$ and $R_2 = 8$ were found empirically to ensure that the light is able to illuminate the 3D model appropriately.

Light Look At: To ensure that the light is visible on the canvas, light Look At is sampled as a function of the camera position:

$$l_i \sim \text{Unif}(K * x_c, K * y_c, K * z_c), \text{where } K = 0.3. \tag{7.9}$$

As in the case of the Camera Look At parameter mentioned above, the value $K = 0.3$ was found empirically.

Light Size: Every light in our setup is implemented as an area light, and therefore requires a height and width to specify the size. We generate the size s_i for light i as:

$$h, w \sim \text{Unif}(L_1, L_2), \tag{7.10}$$

$$s_i = (h, w). \tag{7.11}$$

$L_1 = 0.1, L_2 = 5$ were found empirically to ensure the light illuminates the objects appropriately.

Light Intensity: This parameter specifies the RGB intensity of the light. For light i, RGB color intensity c_i was sampled as:

$$r, g, b \sim \text{Unif}(0, 1), \tag{7.12}$$

$$c_i = (r, g, b). \tag{7.13}$$

Object Material: To ensure no spurious correlations between object texture and category, all object textures were set to a single diffuse material. Specifically, the material is a linear blend between a Lambertian model and a microfacet model with Phong distribution, with Schilick's Fresnel approximation. Diffuse reflectance was set to 1.0, and the material was set to reflect on both sides.

3D models used for generating two different test sets

Our dataset contains 11 categories, with 40 3D models for every category chosen from ShapeNet[58]. Neural networks were evaluated on two test sets - one with the 3D models seen during training, and the second with new, unseen 3D models. The first test set was generated by simply repeating the same procedure as described above. Thus, the (Geometry × Camera × Lighting) joint distribution matches exactly for the train set and this test set. The second test set was created by the exact same generation procedure, but with 10 new 3D models for every category chosen from ShapeNet. The motivation for this second test set was to ensure our models are not over-fitting to the 3D models used for training. Thus, the (Camera × Lighting) joint distribution matches exactly for this test set and the train set, but the Geometry is different in these two sets.

7.4.3 Generating natural images in the vicinity of ImageNet images

One of the major challenges in extending our results to natural image datasets is generating natural images in the vicinity of a correctly classified image by slightly modifying the camera parameters. To do so for ImageNet is equivalent to novel view synthesis (NVS) from single images, which has been a long standing challenging task in computer vision. However, recent advances in NVS enable us to extend our method to natural image datasets like ImageNet[427,448,407,380].

To generate new views in the vicinity of ImageNet images, we rely on a single-view synthesis model based on multi-plane images (MPI)[380]. The MPI model takes as input an image and the (x, y, z) offsets which describe camera movement along the X, Y and Z axes. Note that unlike our renderer, it cannot introduce changes to the camera Look At, Up Vector, Field of View or lighting changes.

7.4.4 Model Training Details

Below we provide the training details including model architectures, optimization strategies and other hyper-parameters used for the binary classification models trained on simplistic parametrically controlled data, and the object recognition models trained on our rendered images of camera and light variations.

MLPs for classifying parametrically controlled uniform data

Let D denote the dimensionality of the input data, and N denote the total number of data points. We used a 5 layer multi-layer perceptron (MLP) with ReLU activations, with the output dimensionality of layers set to $5D$, D, $D/5$, $D/5$, and 2 respectively. However, we found that the number of MLP hidden layers and the number of neurons in these layers had no impact on trends of in-distribution robustness. For experiments with $N < 64,000$ all data was passed in a single batch. For experiments with more data points, each batch contained $64,000$ points. All models were trained for 100 epochs with stochastic gradient descent (SGD) with a learning rate of 0.0001. All experiments were conducted on a compute cluster consisting of 8 NVIDIA TeslaK80 GPUs, and all models were trained on a single GPU at a time. Only models achieving a near perfect accuracy (> 0.99)* on a held-out test set were attacked using CMA-Search.

*Except when dataset size=1000 and dimensions=100 or 500. In these two case the training data was too small for a high test accuracy. These cases are still included for completion.

Object recognition models for classifying images of real-world objects

All CNN models were trained with a batch size of 75 images, while transformers were trained with a batch size of 25. Models were trained for 50 epochs with an Adam optimizer with a fixed learning rate of 0.0003. Other learning rates including 0.0001, 0.001, 0.01 and 0.1 were tried but they performed either similarly well or worse. To get good generalization to unseen 3D models and stable learning, each image was normalized to zero mean and unit standard deviation. As before, all experiments were conducted on our cluster with TeslaK80 GPUs, and each model was trained using a single GPU at a time.

7.4.5 CMA-Search: Finding in-distribution adversarial examples by searching the vicinity

To investigate the in-distribution robustness of neural networks with respect to changes in camera and lighting, we propose a new, gradient-free search method to find incorrectly classified images. Starting with a correctly classified image, our method searches the vicinity by slightly modifying camera or light parameters to find an in-distribution error. While adversarial viewpoints and lighting have been reported before in the literature[234,433,180], there are two major differences in our approach. First, these methods search for an adversarial image by adding a perturbation to the input scene parameter without constraining the resulting image to be within the training distribution. In comparison, our approach searches within the distribution to find in-distribution errors. Secondly, unlike our gradient-free search method, these methods often rely on gradient descent and thus require high dimensional representations of the scene to work well. For instance, these works often use neural rendering where network activations act as a high dimensional representation of the scene[433,185], or use up-sampling of meshes to increase dimensionality[234].

We extend these approaches to work well with our low-dimensional scene representation by utilizing a gradient-free optimization method to search the space—Covariance Matrix Adaptation-Evolution Strategy (CMA-ES)[150,148]. We found that gradient descent with differentiable rendering struggled to find in-distribution errors in our scenes due to the low dimensionality of the optimization problem. CMA-ES has been found to work reliably well with non-smooth optimization problems and especially with local optimization[151], which made it a perfect fit for our search strategy. In contrast to gradient based methods requiring high dimensions, our approach works well for as low as 3 dimensions.

Algorithm 2 provides an outline of using CMA-Search to find in-distribution adversarial examples by searching the vicinity of camera parameters. The algorithm for searching for adversarial examples using light parameters in rendered data, and within parametrically controlled unifrom data is analogous. In Fig. 7.2 we show examples of in-distribution adversarial examples found by our CMA-Search method over camera parameters. Starting with the correctly classified image (left), our method finds an image in the vicinity by slightly modifying camera parameters of the scene. As can be seen, subtle changes in 3D perspective can lead to drastic errors in classification. We also highlight the subtle changes in camera position (in black) and camera Look At (in blue) in the fig-

Algorithm 2 CMA-Search over camera parameters to find in-distribution adversarial examples.

Let $x \in \mathbb{R}^{10}$ denote the camera parameters.
Let Render and Network denote the rendering pipeline and classification network respectively.
function Fitness(x, Render, Network)
    image = Render(x)
    predicted_category, probability = Network(image)
    return predicted_category, probability
end function

Let x_{init} denote initial camera parameters, λ be number of offspring per generation, and y be the image category.

procedure CMA-Search(x_{init}, λ, y)
    initialize $\mu = x_{init}, C = I$ ▷ I denotes identity matrix.
    while True **do**
        for j = $1, ..., \lambda$ **do**
            x_j = sample_multivariate_normal(μ, C) ▷ Generate mutated offspring
            y_j, p_j =FITNESS(x_j, Render, Network) ▷ Calculate fitness of offspring
            if $y_j \neq y$ **then**
                return x_j ▷ Classification fails for image with camera parameters x_j
            end if
        end for
        $x_{1...\lambda} \leftarrow x_{s(1)...s(\lambda)}$, with $s(j)$ = argsort(p_j) ▷ Pick best offspring
        $\mu, C \leftarrow$ update_parameters($x_{1...\lambda}, \mu, C$)
    end while
end procedure

ure. To the best of our knowledge, this is the first evolutionary strategies based search method for finding in-distribution adversarial examples.

Starting from the initial parameters, CMA-ES generates offspring by sampling from a multivariate normal (MVN) distribution i.e. mutating the original parameters. These offspring are then sorted based on the fitness function (classification probability), and the best ones are used to modify the mean and covariance matrix of the MVN for the next generation. The mean represents the current best estimate of the solution i.e. the maximum likelihood solution, while the covariance matrix dictates the direction in which the population should be directed in the next generation. The search is stopped either when a misclassification occurs, or after 15 iterations over scene parameters. For the simplistic parametrically controlled data, we check for a misclassification till 1500 iterations. More details on the exact subroutines for parameter update and theoretical underpinnings of the CMA-ES algorithm can be found in the documentation for pycma[149] and the accompanying paper[148].

7.4.6 Evaluation details

Below we provide details on the evaluation and visualization of in-distribution adversarial examples identified by CMA-Search.

Evaluating CMA-Search using the Attack Rate

To quantify the performance of CMA-Search and the prevalence of in-distribution adversarial examples we propose the Attack Rate, which is reported in Tables

7.1 and 7.3. For the simplistic parametrically controlled data, we first randomly sample 1000 test points which we confirm are correctly classified by the classification model. For each of these test points, CMA-Search is then tasked with starting from the correctly classified point and searching for a nearby point within the training data distribution where the classifier fails. The Attack Rate is the fraction of these 1000 correctly classified points for which CMA-Search is able find an in-distribution adversarial example in the vicinity. This process is repeated 10 times and the average value is reported. For image data, sampling a new point involves sampling scene parameters and rendering the new image corresponding to these parameters. As rendering is significantly slower, we measure the attack rate using 20 samples for image data, and report average over 10 repeats.

Church-window plots

CMA-Search starts from a correctly classified point and provides an in-distribution adversarial example. We use this to define a unit vector in the adversarial direction, and fix this as one of basis vectors for the subspace the data occupies. Assuming data dimensionality to be D, we can calculate the corresponding $D-1$ orthonormal bases. Following the same protocol as past work[401], we randomly pick one of these orthonormal vectors as the orthogonal direction and define a grid of perturbations with fixed increments along the adversarial and the orthogonal directions. These perturbations are then added to the original sample and the model is evaluated at these perturbed samples. We plot correct classifications in white, in-distribution adversarial examples in red, and out-of-distribution samples in black.

Sources of stochasticity

Table 7.1 reports the results of Attack Rates for models as sources of stochasticity are varied one at a time to investigate their impact on model robustness. For these experiments, we studied binary classification models trained on $100,000$ data points of 20-dimensional data. Below, we provide additional details on how these experiments were conducted.

CMA-Search: As our method is based on evolutionary strategies, it is inherently stochastic. To ensure that model robustness is not due to CMA-Search failing stochastically, we repeat the attack on our robust model with CMA-Search 10 times and report the mean attack rates.

Optimization (SGD): To investigate if the optimization process enables certain models to be robust, we use the exact same data points and model initialization as the identified robust model, and repeat the training procedure 10 times to obtain 10 different models. These models differ from each other only due to the stochasticity of SGD.

Sampling Bias: We ask if model robustness is a function of the specific training data sampled from the training distribution - is there a good training dataset that results in more robust models? We test this by using this exact same initialization and SGD seed as the robust model, and train models on newly data sampled from the training distribution. We ensure that the newly sampled dataset has the same size and distribution as the dataset used with the identified robust model.

Model Initialization: To test if the model initialization is the underlying cause for model robustness, we train multiple models with the exact same training data as the robust model, but with different random initialization (different from the robust model) while using the same seed for SGD.

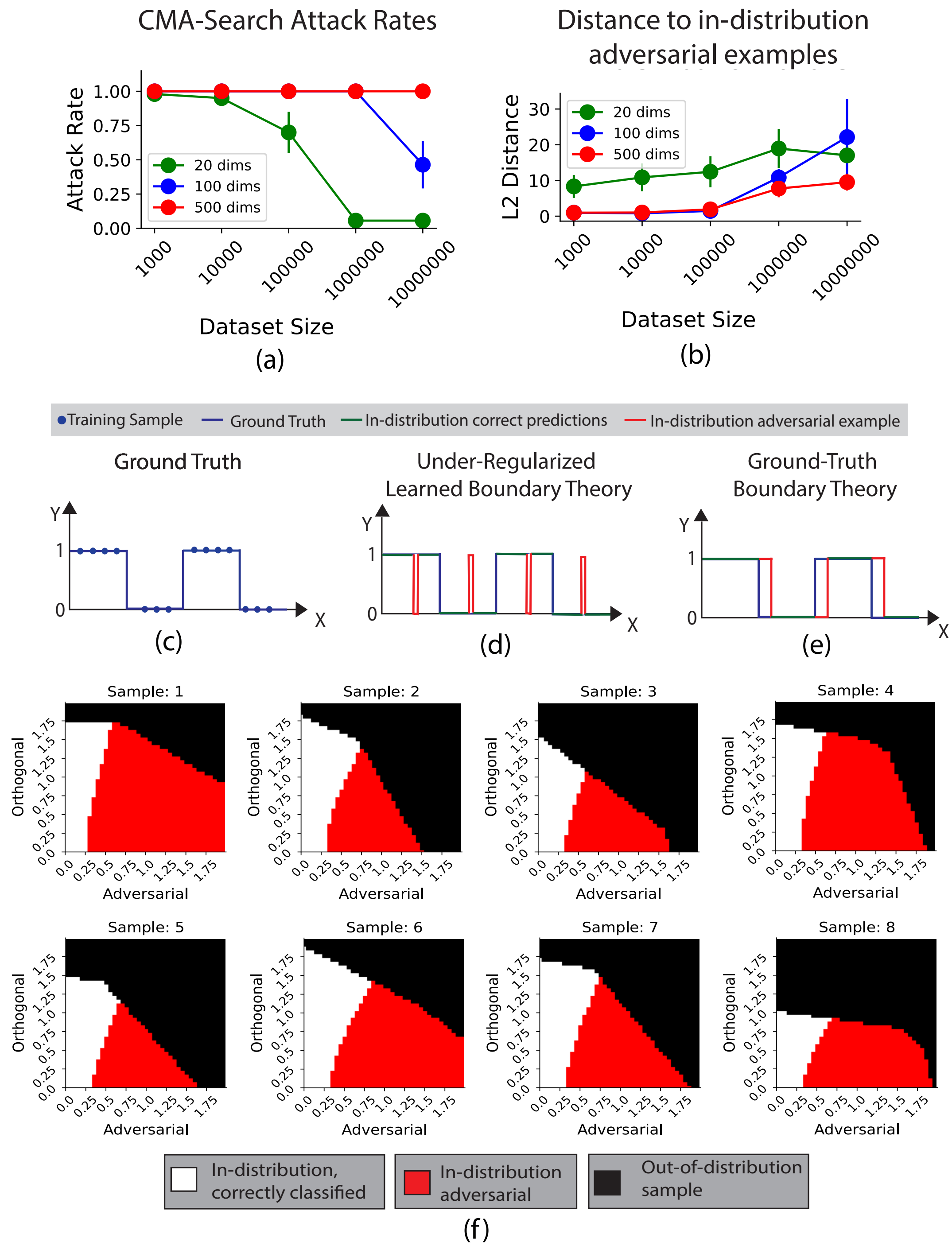


Figure 7.3: Hypotheses explaining in-distribution adversarial examples. (a) Attack rate of CMA-Search in finding an in-distribution adversarial example starting with a correctly classified sample. Models start becoming robust at high dataset sizes, however the sample complexity scales poorly with data dimensionality. (b) Average Euclidean distance between the starting point and the identified in-distribution adversarial sample. As dataset size increases, the average Euclidean distance from the starting point to in-distribution adversarial example increases for all data dimensions. (c) Example of one-dimensional ground-truth function (d) Depiction of under-regularization learned boundary theory (e) Depiction of ground-truth boundary theory (f) Church window plots depicting adversarial examples in the vicinity of category boundaries.

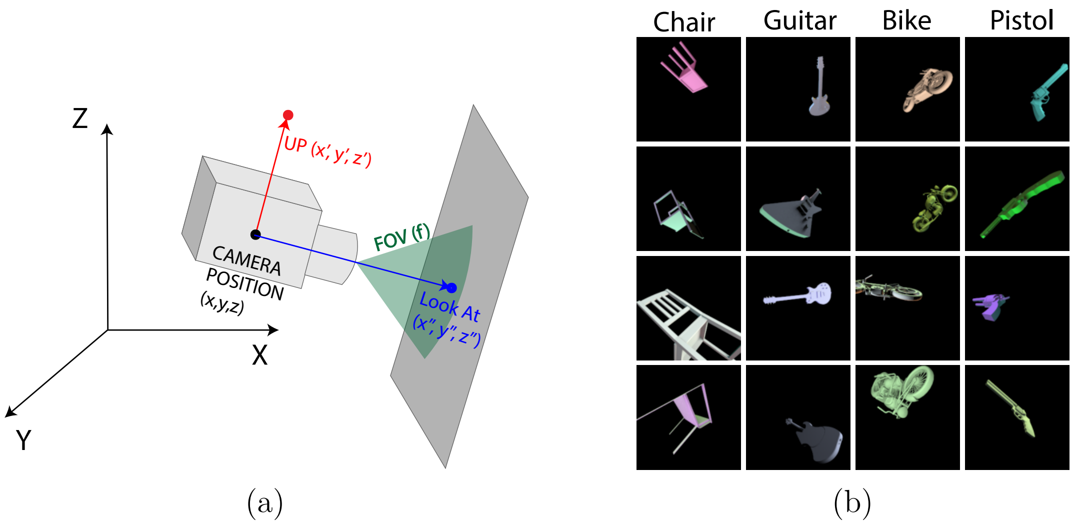


Figure 7.4: 3D scene setup and resulting images. (a) Images in our dataset are completely parametrized by the camera and light. Physical interpretation of the camera parameters is illustrated here. Analogously, light is parametrized by the position, look at, 2D size and the RGB intensities. (b) Sample images for 4 object categories generated using our 3D scene setup. As can be seen, images contain complex viewpoints and locations, multiple colors per object and complex artifacts like self-shadows.

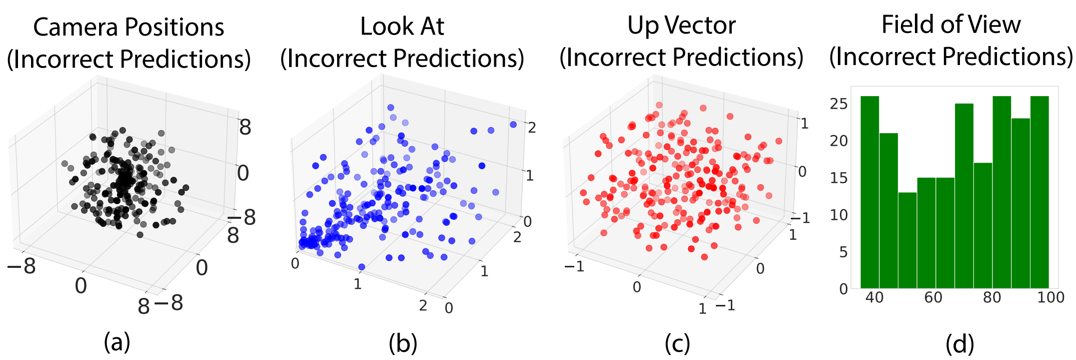


Figure 7.5: Distribution of errors across the scene parameter space. (a) Coordinates of camera positions, (b) Coordinates of Look At, (c) Up Vector and (d) Histogram of errors across lens field of view (FOV). We found no clear, strong patterns which characterize the camera and light conditions of misclassified images. This is in contrast to human vision which is impacted by regions of camera positions (non-canonical viewpoints), and up vector (upside-down orientations) among others.

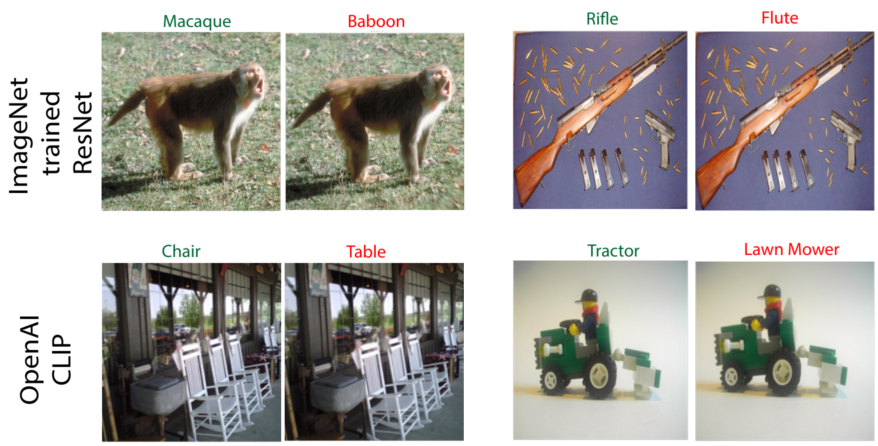


Figure 7.6: CMA-Search on ImageNet images. To replicate results on ImageNet, we replace our rendering pipeline with the single view MPI[380] model to generate novel views of ImageNet images. Here we show results using CMA-Search with the MPI model to find subtle 3D perspective changes which lead to misclassification with ResNet18 and OpenAI's CLIP model.

Part III

Benchmarking and Leveraging human generlization

For me context is the key - from that comes the understanding of everything.

Kenneth Noland

8

Contextual Reasoning in Synthetic and Natural Scenes

8.1 Introduction

A coffee mug is usually a small object (Fig.??a), which does not fly on its own (Fig.??c) and can often be found on a table (Fig.??a) but not on a chair (Fig.??d).

Such contextual cues have a pronounced impact on the object recognition capabilities of both humans[437], and computer vision models[375,66,275,239]. Neural networks learn co-occurrence statistics between an object's appearance and its label, but also between the object's context and its label[97,359,30]. Therefore, it is not surprising that recognition models fail to recognize objects in unfamiliar contexts[311]. Despite the fundamental role of context in visual recognition, it remains unclear what contextual cues should be integrated with object information and how.

Two challenges have hindered progress in the study of the role of contextual cues: (1) context has usually been treated as a monolithic concept and (2) large-scale, internet-scraped datasets like ImageNet[88] or COCO[233] are highly uncontrolled. To address these challenges, we present a methodology to systematically study the effects of an object's context on recognition by leveraging a Unity-based 3D simulation engine for image generation[186], and manipulating 3D objects in a virtual home environment[295]. The ability to rigorously control every aspect of the scene enables us to systematically violate contextual rules and assess their impact on recognition. We focus on three fundamental aspects of context: (1) gravity - objects without physical support, (2) object co-occurrences - unlikely object combinations, and (3) relative size - changes to the size of target objects relative to the background. As a critical benchmark, we conducted psychophysics experiments to measure human performance and compare it with state-of-the-art computer vision models.

We propose a new context-aware architecture, which can incorporate object and contextual information to achieve higher object recognition accuracy given proper context and robustness to out-of-context situations. Our Context-aware Recognition Transformer Network (CRTNet) uses two separate streams to process the object and its context independently before integrating them via multi-head attention in transformer decoder modules. Across multiple datasets, the CRTNet model surpasses other state-of-the-art computational models in normal context and classifies objects robustly despite large contextual variations, much like humans do.

Our contributions in this paper are three-fold. Firstly, we introduce a challenging new dataset for in- and out-of-context object recognition that allows fine-grained control over context violations including gravity, object co-occurrences and relative object sizes (out-of-context dataset, OCD). Secondly, we conduct psychophysics experiments to establish a human benchmark for in- and out-of-context recognition and compare it with state-of-the-art computer vision models. Finally, we propose a new context-aware architecture for object recognition, which combines object and scene information to reason about context and generalizes well to out-of-context images. We release the entire dataset, including our tools for the generation of additional images and the source code for CRTNet at https://github.com/kreimanlab/WhenPigsFlyContext.

8.2 Related Works

Out-of-context datasets: Notable works on out-of-context datasets include the UnRel dataset[289] and the Cut-and-paste dataset presented in[437]. While UnRel is a remarkable collection of out-of-context natural images, it is limited in size and diversity. A drawback of cutting-and-pasting[120] is the introduction of artifacts such as unnatural lighting, object boundaries, sizes and positions. Neither of those datasets allow systematic analysis of individual properties of context. 3D simulation engines enable easily synthesizing many images and systematically investigating the violation of contextual cues. It is challenging to achieve these goals with real-world photographs. Moreover, these simulation engines enable precise control of contextual parameters, changing cues one at a time in a systematic and quantifiable manner.

Out-of-context object recognition: In previous work, context has mostly been studied as a monolithic property in the form of the target object's background. Previous work included testing the generalization to new backgrounds[30] and incongruent backgrounds[437], exploring the impact of foreground-background relationships on data augmentation[103], and replacing image sub-regions by another sub-image, i.e. object transplanting[311]. In this paper, we evaluate different properties of contextual cues (e.g., gravity) in a quantitative, controlled, and systematic manner.

3D simulation engines and computer vision: Recent studies have demonstrated the success of using 3D virtual environments for tasks such as object recognition with simple and uniform backgrounds[39], routine program synthesis[295], 3D animal pose estimation[265], and studying the generalization capabilities of CNNs[241,146]. However, to the best of our knowledge, none of these studies have tackled the challenging problem of how to integrate contextual cues.

Models for context-aware object recognition: To tackle the problem of context-aware object recognition, researchers have proposed classical approaches, e.g., Conditional Random Field (CRF)[128,426,211,62], and graph-based methods[372,410,374,66]. Recent studies have extended this line of work to deep graph neural networks[170,67,89,27]. Breaking away from these previous works where graph optimization is performed globally for contextual reasoning in object recognition, our model has a two-stream architecture which separately processes visual information on both target objects and context, and then integrates them with multi-head attention in stacks of transformer decoder layers. In contrast to other vision transformer models in object recognition[99] and detection[50], CRTNet performs in-context recognition tasks given the target object location.

8.3 Context-aware Recognition Transformer

8.3.1 Overview

We propose the Context-aware Recognition Transformer Network (CRTNet, Figure 8.1). CRTNet is presented with an image with multiple objects and a bounding box to indicate the target object location. The model has three main elements: First, CRTNet uses a stack of transformer decoder modules with multi-head attention to hierarchically reason about context and integrate contextual cues with object information. Second, a confidence-weighting mechanism improves the model's robustness and gives it the flexibility to select what information to rely on for recognition. Third, we curated the training methodology with gradient detachment to prioritize important model components and ensure

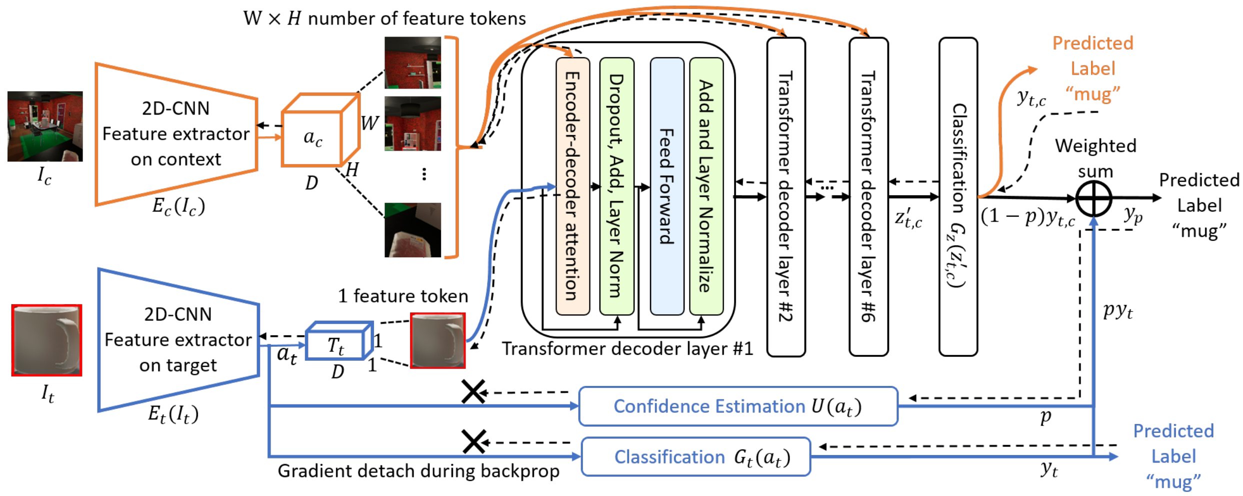


Figure 8.1: Architecture overview of the Context-aware Recognition Transformer Network (CRTNet). CRTNet consists of 3 main modules: feature extraction, integration of context and target information, and confidence-modulated classification. CRTNet takes the cropped target object I_t and the entire context image I_c as inputs and extracts their respective features. These feature maps are then tokenized and the information of the two streams is integrated over multiple transformer decoder layers. CRTNet also estimates a confidence score for recognizing the target object based on object features alone, which is used to modulate the contributions of y_t and $y_{t,c}$ to the final prediction y_p. The dashed lines in backward direction denote gradient flows during backpropagation. The two black crosses denote where the gradient updates stop. See Sec. 8.3 for details.

efficient training of the entire architecture.

Inspired by the eccentricity dependence of human vision, CRTNet has one stream that processes only the target object (I_t, 224×224), and a second stream devoted to the periphery (I_c, 224×224). I_t is obtained by cropping the input image to the bounding box whereas I_c covers the entire contextual area of the image. I_c and I_t are resized to the same dimensions. Thus, the target object's resolution is higher in I_t. The two streams are encoded through separate 2D-CNNs. After the encoding stage, CRTNet tokenizes the feature maps of I_t and I_c, integrates object and context information via hierarchical reasoning through a stack of transformer decoder layers, and predicts class label probabilities $y_{t,c}$

within C classes.

A model that always relies on context can make mistakes under unusual context conditions. To increase robustness, CRTNet makes a second prediction y_t, based on target object information alone, estimates the confidence p of this prediction, and computes a confidence-weighted average of y_t and $y_{t,c}$ to get the final prediction y_p. If the model makes a confident prediction based on the target object alone, this decision overrules the context reasoning stage.

8.3.2 Convolutional Feature Extraction

CRTNet takes I_c and I_t as inputs and uses two 2D-CNNs, $E_c(\cdot)$ and $E_t(\cdot)$, to extract context and target feature maps a_c and a_t, respectively, where $E_c(\cdot)$ and $E_t(\cdot)$ are parameterized by θ_{E_c} and θ_{E_t}. We use the DenseNet architecture[171] with weights pre-trained on ImageNet[88] and fine-tune it. Assuming that different features in I_c and I_t are useful for recognition, we do not enforce sharing of the parameters θ_{E_c} and θ_{E_t}. We demonstrate the advantage of non-shared parameters in the ablation study (Sec. 8.5.5). To allow CRTNet to focus on specific parts of the image and select features at those locations, we preserve the spatial organization of features and define a_c and a_t as the output feature maps from the last convolution layer of DenseNet. Both a_c and a_t are of size $D \times W \times H = 1,664 \times 7 \times 7$, where D, W and H denote the number of channels, width and height of the feature maps respectively.

8.3.3 Tokenization and Positional Encoding

We tokenize the context feature map a_c by splitting it into patches based on locations, following[99]. Each context token corresponds to a feature vector $\mathbf{a_c^i}$ of dimension D at location i where $i \in \{1, .., L = H \times W\}$. To compute target token T_t, CRTNet aggregates the target feature map a_t via average pooling:

$$T_t = \frac{1}{L} \sum_{i=1,\ldots,L} \mathbf{a_t^i} \tag{8.1}$$

To encode the spatial relations between the target token and the context tokens, as well as between different context tokens, we learn a positional embedding of size D for each location i and add it to the corresponding context token $\mathbf{a_c^i}$. For the target token T_t, we use the positional embedding corresponding to the location, within which the bounding box midpoint is contained. The positionally-encoded context and target tokens are denoted by z_c and z_t respectively.

8.3.4 Transformer Decoder

We follow the original transformer decoder[388], taking z_c to compute keys and values, and z_t to generate the queries in the transformer encoder-decoder multi-head attention layer. Since we only have a single target token, we omit the self-attention layer. In the experiments, we also tested CRTNet with self-attention enabled and we did not observe performance improvements. Our decoder layer consists of alternating layers of encoder-decoder attention (EDA) and multi-layer perceptron (MLP) blocks. Layernorm (LN) is applied after each residual connection. Dropout (DROP) is applied within each residual connection and MLP block. The MLP contains two layers with a ReLU non-linearity and DROP.

$$z_{t,c} = \mathrm{LN}(\mathrm{DROP}(\mathrm{EDA}(z_t, z_c)) + z_t) \tag{8.2}$$

$$z'_{t,c} = \mathrm{LN}(\mathrm{DROP}(\mathrm{MLP}(z_{t,c})) + z_{t,c}) \tag{8.3}$$

Our transformer decoder has a stack of $X = 6$ layers, indexed by x. We repeat the operations in Eqs 8.2 and 8.3 for each transformer decoder layer by recursively assigning $z'_{t,c}$ back to z_t as input to the next transformer decoder layer. Each EDA layer integrates useful information from the context and the target object with 8-head selective attention. Based on accumulated information from all previous $x - 1$ layers, each EDA layer enables CRTNet to progressively reason about context by updating the attention map on z_c over all L locations. We provide visualization examples of attention maps along the hierarchy of the transformer decoder modules in Supp. Fig S1.

8.3.5 Confidence-modulated Recognition

The context classifier $G_z(\cdot)$ with parameters θ_{G_z} consists of a fully-connected layer and a softmax layer. It takes the feature embedding $z'_{t,c}$ from the last transformer decoder layer and outputs the predicted class distribution vector: $y_{t,c} = G_z(z'_{t,c})$. Similarly, the target classifier $G_t(\cdot)$, takes the feature map a_t as input and outputs the predicted class distribution vector: $y_t = G_t(a_t)$.

Since neural networks are often fooled by incongruent context[437], we propose a confidence-modulated recognition mechanism balancing the predictions from $G_t(\cdot)$ and $G_z(\cdot)$. The confidence estimator $U(\cdot)$ with parameters θ_U takes the target feature map a_t as input and outputs a value p indicating how confident CRTNet is about the prediction y_t. $U(\cdot)$ is a feed-forward multi-layer perceptron network with a sigmoid function to normalize the confidence score to [0, 1].

$$p = \frac{1}{1 + e^{-U(a_t)}} \tag{8.4}$$

We use p to compute a confidence-weighted average of $y_{t,c}$ and y_t for the final predicted class distribution y_p: $y_p = py_t + (1 - p)y_{t,c}$. The higher the confidence p, the more CRTNet relies on the target object itself, rather than on the integrated contextual information, for classification. We demonstrate the advantage of using y_p rather than $y_{t,c}$ or y_t as a final prediction in the ablation study (Sec. 8.5.5).

8.3.6 Training

CRTNet is trained end-to-end with three loss functions: (i) to train the confidence estimator $U(\cdot)$, we use a cross-entropy loss with respect to the confidence-weighted prediction y_p. This allows $U(\cdot)$ to learn to increase the confidence value p when the prediction y_t based on target object information alone is correct. (ii) To train $G_t(\cdot)$, we use a cross-entropy loss with respect to y_t. (iii) For the other components of CRTNet, including the transformer decoder modules and the classifier $G_z(\cdot)$, we use a cross-entropy loss with respect to $y_{t,c}$. Instead of training everything based on y_p, the three loss functions together maintain strong learning signals for all parts in the architecture irrespective of the confidence value p.

To facilitate learning for specific components in CRTNet, we also introduce gradient detachments during backpropagation (Fig. 8.1). Gradients flowing through both $U(\cdot)$ and $G_t(\cdot)$ are detached from $E_t(\cdot)$ to prevent them from driving the target encoder to learn more discriminative features, which could impact the efficacy of the transformer modules and $G_z(\cdot)$. We demonstrate the benefit of these design decisions in ablation studies (Sec. 8.5.5).

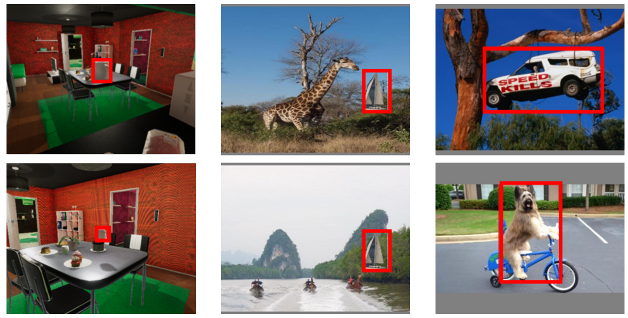
8.4 Experimental Details

8.4.1 Baselines

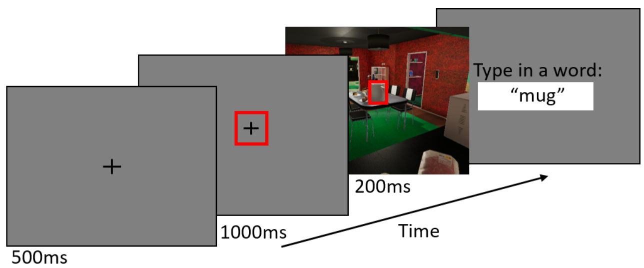
CATNet[437] is a context-aware two-stream object recognition model. It processes the visual features of a cropped target object and context in parallel, dynamically incorporates object and contextual information by constantly updating its attention over image locations, and sequentially reasons about the class label for the target object via a recurrent neural network.

Faster R-CNN[308] is an object detection algorithm. We adapted it to the context-aware object recognition task by replacing the region proposal network with the ground truth bounding box indicating the location of the target object.

DenseNet[171] is a 2D-CNN with dense connections that takes the cropped target object patch I_t as input.



(a) OCD (b) Cut-and-paste (c) UnRel



(d) Schematic of human psychophysics experiment

Figure 8.2: Datasets and psychophysics experiment scheme. (a-c) Example images for each dataset. The red box indicates the target location. In (a), two contextual modifications (gravity and size) are shown. In (b), the same target object is cut and pasted into either incongruent or congruent conditions. (c) consists of natural images. (d) Subjects were presented with a fixation cross (500 ms), followed by a bounding box indicating the target object location (1000 ms). The image was shown for 200 ms. After image offset, subjects typed one word to identify the target object.

8.4.2 Datasets

Out-of-context Dataset (OCD)

Our out-of-context dataset (OCD) contains 36 object classes, with 15,773 test images of complex and rich scenes in 6 contextual conditions (described below). We leveraged the VirtualHome environment[295] developed in the Unity simulation engine to synthesize these images in indoor home environments within 7 apartments and 5 rooms per apartment. These rooms include furnished bedrooms, kitchens, study rooms, living rooms and bathrooms[295] (see Fig. ?? for examples). We extended VirtualHome with additional functionalities to manipulate object properties, such as materials and scales, and to place objects in out-of-context locations. The target object is always centered in the camera view; collision checking and camera ray casting are enabled to prevent object collisions and occlusions.

Normal Context and No Context: There are 2,309 images with normal context (Fig. ??b), and 2,309 images for the no-context condition (Fig. ??g). For the normal context condition, each target object is placed in its "typical" location, defined by the default settings of VirtualHome. We generate a corresponding no context image for every normal context image by replacing all the pixels surrounding the target object with either uniform grey pixels or salt and pepper noise.

Gravity: We generated 2,934 images where we move the target object along the vertical direction such that it is no longer supported (Fig. ??c). To avoid cases where objects are lifted so high that their surroundings change completely, we set the lifting offset to 0.25 meters.

Object Co-occurrences: To examine the importance of the statistics of object co-occurrences, four human subjects were asked to indicate the most likely rooms and locations for the target objects. We use the output of these responses to generate 1,453 images where we place the target objects on surfaces with lower co-occurrence probability, e.g. a microwave in the bathroom and Fig. ??d.

Object Co-occurrences + Gravity: We generated 910 images where the objects are both lifted and placed in unlikely locations. We chose walls, windows, and doorways of rooms where the target object is typically absent (Fig. ??e). We place target objects at half of the apartment's height.

Size: We created 5,858 images where we changed the target object's size to 2, 3, or 4 times its original size while keeping the remaining objects in the scene intact (Fig. ??f).

Real-world Out-of-context Datasets

The Cut-and-paste dataset[437] contains 2,259 out-of-context images spanning 55 object classes. These images are grouped into 16 conditions obtained through the combinations of 4 object sizes and 4 context conditions (normal, minimal, congruent, and incongruent) (Fig. 8.2b).

The UnRel[289] Dataset contains more than 1,000 images with unusual relations among objects spanning 100 object classes. The dataset was collected from the web based on triplet queries, such as "dog rides bike" (Fig. 8.2c).

8.4.3 Performance Evaluation

Evaluation of Computational Models: We trained the models on natural images from COCO-Stuff[48] using the annotations for object classes overlapping with those in the respective test set (16 overlapping classes between VirtualHome and COCO-Stuff, 55 overlapping classes between Cut-and-paste and COCO-Stuff and 33 overlapping classes between UnRel and COCO-Stuff). The models were then tested on OCD, the Cut-and-paste dataset, UnRel, and on a COCO-Stuff test split.

Behavioral Experiments: We evaluated human recognition on OCD and the Cut-and-paste dataset, as schematically illustrated in Fig. 8.2d, on Amazon Mechanical Turk (MTurk)[384]. We recruited 400 subjects per experiment, yielding $\approx 67,000$ trials. To avoid biases and potential memory effects, we took several precautions: (a) Only one target object from each class was selected; (b) Each subject saw each room only once; (c) The trial order was randomized.

Computer vision and most psychophysics experiments enforce N-way categorization (e.g.[363]). Here we used a more unbiased probing mechanism whereby subjects could use any word to describe the target object. We independently collected ground truth answers for each object in a separate MTurk experiment with infinite viewing time and normal context conditions. These Mturk subjects did not participate in the main experiments. Answers in the main experiments were then deemed correct if they matched any of the ground truth responses[437].

A completely fair machine-human comparison is close to impossible since humans have decades of visual+ experience with the world. Despite this caveat, we find it instructive to show results for humans and models on the same images. We tried to mitigate the differences in training by focusing on the qualitative impact of contextual cues in perturbed conditions compared to the normal context condition. We also show human-model correlations to describe their relative trends across all conditions.

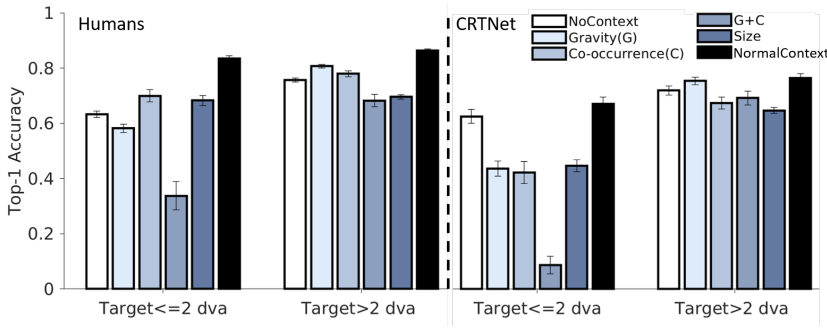


Figure 8.3: The CRTNet model exhibits human-like recognition patterns across contextual variations in our OCD dataset. Different colors denote contextual conditions (Sec. 8.4.2, Fig. ??). We divided the trials into two groups based on target object sizes in degrees of visual angle (dva). Error bars denote the standard error of the mean (SEM).

OCD	Overall
CRTNet (ours)	0.89
Baselines	
CATNet[437]	0.36
Faster R-CNN[308]	0.73
DenseNet[171]	0.66
Ablations	
Ablated-SharedEncoder	0.84
Ablated-TargetOnly	0.89
Ablated-Unweighted	0.83
Ablated-NoDetachment	0.88

Table 8.1: Linear correlations between human and model performance over 12 contextual conditions.

8.5 Results

8.5.1 Recognition in our OCD Dataset

Figure 8.3 (left) reports recognition accuracy for humans over the 6 context conditions (Sec. 8.4.2, Fig. ??) and 2 target object sizes (total of 12 conditions). Comparing the no-context condition (white) versus normal context (black), it is evident that contextual cues lead to improvement in recognition, especially for smaller objects, consistent with previous work [437].

Gravity violations led to a reduction in accuracy. For small objects, the gravity condition was even slightly worse than the no context condition; the unusual context can be misleading for humans. The effects were similar for the changes in object co-occurrences and relative object size. Objects were enlarged by a factor of 2, 3, or 4 in the relative size condition. Since the target object gets larger, and because of the improvement in recognition with object size, we would expect a higher accuracy in the size condition compared to normal context. However, increasing the size of the target object while keeping all other objects intact, violates the basic statistics of expected relative sizes (e.g., we expect a chair to be larger than an apple). Thus, the drop in performance in the size condition is particularly remarkable and shows that violation of contextual cues can override basic object recognition.

Combining changes in gravity and in the statistics of object co-occcurrences led to a pronounced drop in accuracy. Especially for the small target objects, violation of gravity and statistical co-occurrences led to performance well below that in the no context condition.

These results show that context can play a facilitatory role (compare normal versus no context), but context can also impair performance (compare gravity+co-occurrence versus no context). In other words, unorthodox contextual information hurts recognition.

Figure 8.3 (right) reports accuracies for CRTNet. Adding normal contextual information (normal context vs no context) led to an improvement of 4% in performance for both small and large target objects. Remarkably, the CRTNet model captured qualitatively similar effects of contextual violations as those observed in humans. Even though the model performance was below humans in absolute terms (particularly for small objects), the basic trends associated with the role of contextual cues in humans can also be appreciated in the CRTNet results. Gravity, object co-occurrences, and relative object size changes led to a decrease in performance. As in the behavioral measurements, these effects were more pronounced for the small objects. For CRTNet, all conditions led to worse performance than the no context condition for small objects.

8.5.2 Recognition in the Cut-and-paste Dataset

Synthetic images offer the possibility to systematically control every aspect of the scene, but such artificial images do not follow all the statistics of the natural world. Therefore, we further evaluated CRTNet and human performance in the naturalistic settings of the Cut-and-paste dataset[437] (see Table 8.2). The CRTNet model yielded results that were consistent with, and in many conditions better than, human performance. As observed in the human data, performance increases with object size. In addition, the effect of context was more pronounced for smaller objects (compare normal context (NC) versus minimal context (MC) conditions).

In accordance with previous work[437], compared to the minimal context condition, congruent contextual information (CG) typically enhanced recognition whereas incongruent context (IG) impaired performance. Although the congruent context typically shares similar correlations between objects and scene properties, pasting the object in a congruent context led to weaker enhancement than the normal context. This lower contextual facilitation may be due to erroneous relative sizes between objects, unnatural boundaries created by pasting, or contextual cues specific to each image. CRTNet was relatively oblivious to these effects and performance in the congruent condition was closer to that in the normal context condition whereas these differences were more striking for humans. In stark contrast, incongruent context consistently degraded recognition performance below the minimal context condition for both CRTNet and humans.

8.5.3 Recognition in the UnRel Dataset

The Cut-and-paste dataset introduces artifacts (such as unnatural boundaries and erroneous relative sizes) due to the cut-and-paste process. Therefore, we also evaluated CRTNet on the UnRel dataset[289]. We use the performance on the COCO-Stuff[48] test split as reference for normal context in natural images.

CRTNet showed a slightly lower recognition accuracy in the out-of-context setting (Fig. 8.4).

8.5.4 Comparison with Baseline Models

Performance Evaluation: Although Faster R-CNN and CATNet leverage global contextual information, CRTNet outperformed both models, especially on small objects (OCD: Fig. 8.3 and Supp. Fig. S7-S8; Cut-and-Paste: Table8.2; UnRel: Fig. 8.4). Furthermore, Table 8.1 shows that CRTNet's performance pattern across the different OCD conditions is much more similar to the human performance pattern (in terms of correlations) than the other baseline models.

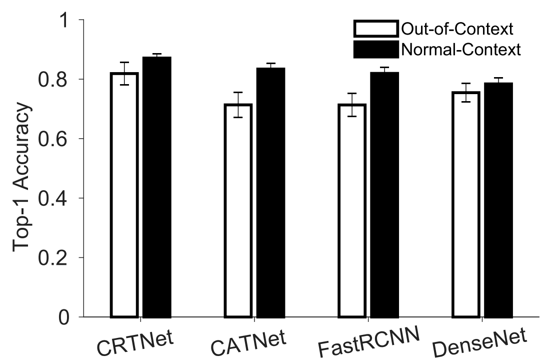


Figure 8.4: CRTNet surpasses all baselines in both normal (COCO-Stuff[48]) and out-of-context (UnRel[289]) conditions.

Architectural Differences: While all baseline models can rely on an intrinsic notion of spatial relations, CRTNet learns about spatial relations between target and context tokens through a positional embedding. A visualization of the learned positional embeddings (Supp. Fig. S1) shows that CRTNet learns image topology by encoding distances within the image in the similarity of positional

	Size [0.5, 1] dva				Size [1.75, 2.25] dva				Size [3.5, 4.5] dva				Size [7, 9] dva			
	NC	CG	IG	MC	NC	CG	IG	MC	NC	CG	IG	MC	NC	CG	IG	MC
Humans	56.0	18.8	5.9	10.1	66.8	48.6	22.3	38.9	78.9	66.0	38.8	62.0	88.7	70.7	59.0	77.4
[437]	(2.8)	(2.3)	(1.3)	(1.7)	(2.7)	(2.8)	(2.4)	(2.8)	(2.4)	(2.7)	(2.6)	(2.8)	(1.7)	(2.6)	(2.8)	(2.3)
CRTNet	50.2	43.9	10.6	17.4	78.4	81.4	41.2	56.7	91.5	87.3	51.1	76.6	92.9	87.7	66.4	83.0
(ours)	(2.8)	(2.8)	(1.7)	(2.1)	(3.0)	(2.8)	(3.5)	(3.6)	(1.1)	(1.3)	(1.9)	(1.6)	(0.9)	(1.2)	(1.7)	(1.4)
CATNet	37.5	29.2	3.6	6.1	53.0	46.5	10.9	22.1	72.8	71.2	24.5	38.9	81.8	78.9	47.6	74.8
[437]	(4.0)	(2.4)	(1.0)	(2.0)	(4.1)	(2.5)	(1.6)	(3.6)	(3.6)	(2.4)	(2.2)	(3.9)	(3.0)	(2.1)	(2.6)	(3.5)
Faster R-CNN	24.9	10.9	5.9	7.2	44.3	27.3	20.1	16.5	65.1	53.2	39.0	42.9	71.5	64.3	55.0	64.6
[308]	(2.4)	(1.7)	(1.3)	(1.4)	(3.6)	(3.2)	(2.9)	(2.7)	(1.8)	(1.9)	(1.9)	(1.9)	(1.6)	(1.7)	(1.8)	(1.7)
DenseNet	13.1	10.0	11.2	12.5	45.4	42.3	39.7	46.4	67.1	62.3	55.4	67.1	74.9	67.2	63.5	74.9
[171]	(1.9)	(1.7)	(1.8)	(1.8)	(3.6)	(3.5)	(3.5)	(3.6)	(1.8)	(1.9)	(1.9)	(1.8)	(1.6)	(1.7)	(1.7)	(1.6)

Table 8.2: Recognition accuracy of humans, the CRTNet model, and three different baselines on the Cut-and-paste dataset[437]. There are 4 conditions for each size: normal context (NC), congruent context (GC), incongruent context (IG) and minimal context (MC) (Sec. 8.4.2). Bold highlights the best performance. Numbers in brackets denote the standard error of the mean.

embeddings.

In CATNet, the attention map iteratively modulates the extracted feature maps from the context image at each time step in a recurrent neural network, whereas CRTNet uses a stack of feedforward transformer decoder layers with multi-head encoder-decoder attention. These decoder layers hierarchically integrate information via attention maps, modulating the target token features with context.

DenseNet takes cropped targets as input with only a few surrounding pixels of context. Its performance dramatically decreases for smaller objects, which also results in lower correlation with the human performance patterns. For example, in the Cut-and-paste dataset, CRTNet outperforms DenseNet by 30% for normal context and small objects (Table 8.2) and in OCD, DenseNet achieves a correlation of 0.66 vs. 0.89 for CRTNet (Table 8.1).

8.5.5 Ablation Reveals Critical Model Components

We assessed the importance of design choices by training and testing ablated versions of CRTNet on the OCD dataset.

Shared Encoder: In the CRTNet model, we trained two separate encoders to extract features from target objects and the context respectively. Here, we enforced weight-sharing between these two encoders (Ablated-SharedEncoder) to assess whether the same features for both streams are sufficient to reason about context. The results (Table 8.1, Supp. Fig. S3) show that the ablated version achieved a lower recognition accuracy and lower correlation with the psychophysics results.

Recognition Based on Target or Context Alone: In the original CRTNet model, we use the confidence-weighted prediction y_p. Here, we tested two alternatives: CRTNet relying only on the target object (y_t, Ablated-TargetOnly) and CRTNet relying only on contextual reasoning ($y_{t,c}$, Ablated-Unweighted). The original model benefits from proper contextual information compared to the target-only version but it is slightly more vulnerable to some of the context perturbations as would be expected. It consistently outperforms the context-only version demonstrating the usefulness of the confidence-modulation mechanism.

Joint Training of the Target Encoder: In Sec. 8.3.6, we use gradient detachments to make the training of the target encoder $E_t(\cdot)$ independent of $G_t(\cdot)$ such that it cannot force the target encoder to learn more discriminative features. Here we remove this constraint (Ablated-NoDetachment, Supp. Fig. S6). The results are inferior to the ones of our original CRTNet, supporting the use of the gradient detachment method.

8.6 Conclusion

We introduced the OCD dataset and used it to systematically and quantitatively study the role of context in object recognition. OCD allowed us to rigorously scrutinize the multi-faceted aspects of how contextual cues influence visual recognition. We conducted experiments with computational models and complemented them with psychophysics studies to gauge human performance. Since the synthetic images in OCD can still be easily distinguished from real photographs, we addressed potential concerns due to the domain gap with experiments on two additional datasets consisting of real-world images.

We showed consistent results for humans and computational models over all three datasets. The results demonstrate that contextual cues can enhance visual recognition, but also that the "wrong" context can impair visual recognition capabilities both for humans and models.

We proposed the CRTNet model as a powerful and robust method to make use of contextual information in computer vision. CRTNet performs well compared to competitive baselines across a wide range of context conditions and datasets. In addition to its performance in terms of recognition accuracy, CRTNet's performance pattern was also found to resemble human behavior more than that of any baseline model.

Acknowledgements This work was supported by NIH R01EY026025 and the Center for Brains, Minds and Machines, funded by NSF STC award CCF-1231216. MZ is supported by a postdoctoral fellowship of the Agency for Science, Technology and Research. We thank Leonard Tang, Jeremy Schwartz, Seth Alter, Xavier Puig, Hanspeter Pfister, Jen Jen Chung, and Cesar Cadena for useful discussions and support.

9

Human or Machine? Turing tests for LLMs and Vision.

9.1 Introduction

Current computer vision and language models excel in a wide range of tasks such as image captioning[398,399,227,259,mic], text generation [17,288,302,43,282,338,130,7,366], object recognition[mic,411,160,162,25,272,65], and attention prediction[436,142,152]. Developing state-of-the-art algorithms goes hand-in-hand with the development of better and more precise ways of assessing their performance. The performance of AI algorithms is often defined by comparing the outputs of AI models against human ground truth annotations. Such metrics are particularly prevalent in computer vision, such as in object detection tasks[305,232,50]. Similarly, several metrics have been proposed to evaluate image captioning models, such as BLEU[284], THUMB[193], and METEOR[90] (see also[81]). Generative AI models are notoriously difficult to evaluate due to the inherent ambiguities of creating novel content[144,192]. Human evaluators are often recruited to assess the quality of sentiment, semantic relevance, reasoning abilities, or emotional valence on text generated by large language models[94,43,192,140,219,364,184,255,306,181,117,12,351].

The Turing test, also known as the "imitation game", was proposed by Alan Turing in 1950 as a way of assessing a machine's ability to exhibit intelligent behaviors indistinguishable from those shown by humans[383]. In the game, a machine tries to pass as a human during a conversation, and a human judge determines whether they are interacting with a human or not[383] (Figure 9.1A). Since its inception, the Turing test has generated extensive controversy and discussion. Several notable arguments include Searle's Chinese room thought experiment[323], Block's behaviorism[36], Harnad's Total Turing Test[154], Watt's Inverted Turing Test[403], Damassino's Questioning Turing Test[84] and Sejnowski's Reverse Turing Test[324]. In parallel with the unbounded optimistic attitudes towards AI in the 1960s and the sober realization of the immense difficulties in AI afterward, many in the scientific community have shifted away from the question of whether the Turing test is a valid and meaningful measure of intelligence (e.g.,[210,159,296,138,139,68] among others) to focus instead on average performance metrics for AI algorithms. Distinct from these discussions, the purpose of our work is not to argue in favor or against Turing tests as a measure of general intelligence. Instead, we consider Turing-like tests as a quantitative evaluation of how well current AIs can imitate humans[155,156,157,184,140].

The key target of the original Turing test focused on conversations. Generating human-like text during conversations remains a daunting challenge for AI with exciting progress. There have been numerous early attempts at generating restricted topics during conversations, such as Colby's PARRY simulating a paranoid schizophrenic[74,75] and Weizenbaum's ELIZA simulating a psychiatrist[404]. However, none of these models have come close to unrestricted Turing tests. Advances in large language models[94,43,76,339,366,17,7,378,15,250] have led news and social media to produce anecdotal claims about current AI being sentient in conversations[Wertheimer,Tiku,Maruf]. However, few studies rigorously and quantitatively assessed AIs in their ability to imitate humans in conversations[442,184,255,181,351].

AI models can show similar average performance to humans in narrow tasks and standard benchmarks, or even outperform humans, and still be distinguishable from humans. For example, both humans and algorithms could be wrong when labeling an image, but they could be wrong in different ways. In language tasks, humans and algorithms could provide adequate answers, yet the answers may be different. Turing tests provide a unique and distinct assessment of AI models as imitators of human behavior, extending and complementing current benchmark frameworks. As AI algorithms continue to blossom in the real world, it is becoming increasingly important for lay people, law professionals, clinicians, politicians, and other experts to ascertain whether the agent they are interacting with is a human or not. For example, the inability to distinguish real news from AI-generated fake news or DeepFakes[406] can have disastrous implications for electoral campaigns [406,147]. Additionally, criminals are increasingly using AI-generated conversations to make phishing scams more convincing and such scams have even started overtaking other types of physical crimes[6,5,335]. To mitigate these issues, the AI community has started developing models to discern whether the generated perceptual inputs come from humans or AIs[164,140], including DeepFake detection on images[337], and GROVER models on fake news detection[432]. Cui et al. proposed a learned critique model acting as a human judge to perform a Turing Test in image captioning tasks[81].

Here we set out to systematically and quantitatively investigate the ability of current language and vision algorithms to imitate humans. To this end, we present an integrative benchmark encompassing a wide range of standard and well-established AI tasks across both language and vision. Motivated by the recognition that each AI task measures only a narrow aspect of human intelligence and is incomplete in isolation, we combine these tasks into one integrative benchmark. This approach aims to provide a more holistic assessment of the abilities to emulate human-like performance across various domains. We start by creating an extensive benchmark dataset of 7,100 answers from 549 human participants across different demographic groups and 26 AI models in 3 language tasks (Image captioning, Word associations, Conversations), and 3 vision tasks (Color estimation, Object detection, and Attention prediction). These tasks were chosen to span a typical and wide range of real-world applications

(Fig. 9.1, Table Supp..1). Next, we systematically evaluated the ability of 1,126 human judges to discern whether task answers come from humans or AI in Turing tests, showing that current algorithms are remarkably adept at imitating humans under restricted testing conditions. Finally, as a proof-of-principle, we show that even though human judges may partially struggle to distinguish human answers, simple machine learning algorithms can serve as AI judges and vastly outperform human judges in Turing tests.

9.2 Results

We conducted 6 Turing tests, including 3 vision tests and 3 language tests. We focus first on the 3 language tests: Image captioning, Word association, and Conversations. In the last part of the results, we present results for the 3 vision tests: Object detection, Color estimation, and Attention prediction.

Collection of human and AI agent responses in language tasks

In order to conduct Turing tests, we first compiled datasets of responses from human participants (labeled H throughout) and AI machine models (henceforth referred to as M) (Fig. Supp.58, Methods). Several controls were introduced in each task to ensure the quality of the responses (Methods). These responses were then assessed in Turing tests to determine if an independent group of human judges or AI judges were able to differentiate between human and AI responses (Fig. 9.1A). We also collected basic demographic information about each participant (Fig. 9.1B).

We start by describing the dataset of responses by human and AI agents for each task. In the Image captioning task, participants and AI models were presented with an image and asked to provide a caption for it (Fig. Supp.58A). We collected responses to 1,000 images from 229 Amazon Mechanical Turk (AMT) participants and 5 AI models (Table Supp..1). Demographic information about the participants is shown in Fig. Supp.59A,C. Example responses from human participants and each one of the 5 AI models are shown in Fig. 9.2A,B.

We considered two slightly different versions of the Word association task. In one version participants were presented with a cue word and asked to provide a single word association (Fig. Supp.58B1). In the second version, participants were presented with a short prompt containing example word associations and a cue word and they were then asked to provide a single word association (Fig. Supp.58B2). As the results were similar for the two versions, we merged the datasets. We collected responses to 1,500 prompts across the two task versions from 40 AMT participants and 5 AI models (Table Supp..1). Demographic information about the participants is shown in Fig. Supp.59E,G. Example responses from human participants and each one of the 5 AI models are shown in Fig. 9.2C.

In the Conversation task, two agents engaged in a dialogue via text messages. Each agent could be a human or an AI model. Thus, we collected human-human conversations, human-machine conversations, and machine-machine conversations (Fig. Supp.58C). Each conversation had a total of 30 entries, 15 from each agent. Agents were unaware of whether they were conversing with another human or with an AI model. Human agents were told that they would engage in a brief conversation about different topics and participated via a public chatting platform where investigators acted as intermediaries to pass messages between agents (Methods). We collected a total of 82 conversations (each one with 30 entries) from 150 participants. Demographic information about the participants is shown in Fig. Supp.59I,K. We collected 320 responses from 6 AI models (Table Supp..1). The prompts used for each AI model are shown in the Methods section. Example conversations can be seen in Fig. 9.2D and Supplementary Section S24.

Operational definition of Turing tests

Once responses from human agents and AI agents were collected for each task, an independent set of participants was recruited for the Turing tests. We refer to our tests as Turing-like to emphasize that they are not identical to the original Turing definitions[383] (Discussion). During each Turing test, we collected human answers on AMT, Prolific (another large online crowd-sourcing platform), or in the lab. We presented a single instance of the answers and asked participants to indicate whether the answer came from a human or an AI agent (Fig. 9.1C, D, E). We also collected demographic information about the participants as metadata, including age, gender, and educational background (Fig. 9.1B). The trial order was always randomized with half of the trials containing responses from human agents, and the other half containing responses from AI agents. We introduced multiple controls to ensure the quality of the responses collected in each Turing test experiment (Methods). Additionally, we also trained computational models to act as AI judges to determine whether a particular answer came from a human or not (Methods).

AI models are close to passing three language Turing tests

For each Turing test, the ground truth response could come from a human agent (H) or a machine (M). The judge indicated H or M in a two-alternative forced-choice manner. We report confusion matrices where the diagonals show the proportion of trials where judges indicated H given that the ground truth was H ($p(H|H)$), and the proportion of trials where judges indicated M given that the ground truth was M ($p(M|M)$) (Fig. 9.3A, B, C). Non-diagonal elements show $p(M|H)$ (the probability that agents indicated M when the ground truth was H) and $p(H|M)$ (the probability that agents indicated H when the ground truth was M). Entries within each row add up to 1. When comparing different AI models for a given task in terms of their ability to imitate humans, values closer to 0.5 for $p(H|H)$ and $p(M|M)$ indicate better imitator models. In contrast, when comparing different judges (e.g., human versus AI judges, or human judges of different ages or educational backgrounds), higher probabilities $p(H|H)$ and $p(M|M)$ departing from 0.5 indicate better judges. We summarize the confusion matrices in a single number by defining the overall imitation detectability as $\frac{1}{2}(p(H|H) + p(M|M))$, ranging from 0.5 (good imitator) to 1.0 (poor imitator).

We summarize the results of the Turing tests for each task for human judges in Fig. 9.3A, B, C, by averaging across all AI models and all human judge demographics. In the image captioning task, human judges distinguished human captions as human 67% of the time and AI captions as AI 46% of the time (Fig. 9.3A). We worried that AMT participants could be poor judges. Therefore, we repeated the Turing experiments in person in our lab (Fig. Supp.60A). Despite our initial concerns, the results from in-person experiments were comparable to those obtained from AMT participants.

There were rather large differences among AI models (Fig. Supp.61A), with proportions of AI captions labeled AI ranging from 37% (BLIP) to 60% (ClipCap). GIT[398], OFA[399], and BLIP[227] are recent transformer-based models and are good at fooling human judges (Fig. Supp.61A). In contrast to ClipCap which is only trained in the Image captioning task, the above three models are generative unified transformer frameworks, trained on multiple tasks (such as image-text retrieval and visual question answering). This suggests that multi-task training aids models to generate human-like captions. The set of images included MSCOCO images[64] and out-of-domain nocaps images[11] which contain object classes that are visually distinct from those in MSCOCO. Despite these differences, there was no statistically significant difference between out-of-domain nocaps images and in-domain nocaps images, even for models like GIT, OFA, and BLIP which were fine-tuned on MSCOCO (Fig. Supp.63) Moreover, on out-of-domain nocaps images, BLIP outperformed all other models in the Turing tests.

In the Word associations task, human judges labeled human word associations as humans 58% of the time and AI word associations as AI 47% of the time (Fig. 9.3B). The results of in-person experiments were comparable to those from AMT participants (Fig. Supp.60B). Among all the AI models, Word2Vec showed the poorest performance in fooling human judges ($p(M|M) = 0.58$). GPT3 (prompt davinci) outperformed the other models ($p(M|M) = 0.41$), Fig. Supp.61C).

In the Conversation task, judges distinguished human participants in conversations as humans 66% of the time and AI agents as AI 47% of the time (Fig. 9.3C, Supp.61E, Supp.60C). We repeated the experiments on the AMT platform, the Prolific platform, and also in the lab. In contrast to the previous two tasks, here it was slightly more difficult to fool the in-person participants compared to the Prolific participants or the AMT participants (Fig. Supp.60C). We separately considered human-human, human-AI, and AI-AI conversations (Fig. Supp.61E). Surprisingly, human-human conversations were classified as human only 64% of the time and AI-AI conversations were classified as human 57% of the time. Blenderbot was classified as human 67% of the time in AI-AI conversations, suggesting that AIs can be perceived as more human than humans themselves. In human-AI conversations, human participants were labeled as humans 68% of the time, and AIs were classified as AI 56% of the time. Judges were more likely to identify machine answers in AI-AI conversations than in human-AI conversations, suggesting that AIs reveal their true self more often when talking to humans than amongst themselves. This observation is consistent with the notion that judges are more accurate in making comparisons rather than absolute evaluations. When comparing different AI models, Blenderbot was more frequently labeled as human, 67% of the time, in AI-AI conversations than in human-AI conversations, 49% of the time (Fig. Supp.61E). The results with ChatGPT version 3.5[7] showed that this model is a slightly worse imitator than Blenderbot (higher imitation detectability) and a better imitator than most of the other models in imitation detectability (Fig. Supp.62, compare to Fig. Supp.60C and Fig. Supp.61E).

Next, we investigated whether and to what extent the results of the Turing tests depend on the length of the conversation (Fig. Supp.64). There was only a very slight increase in judges' ability to correctly detect humans as humans with increasing conversation length across all conversations (Fig. Supp.64A), which was hardly noticeable when splitting the results into human-AI conversations (Fig. Supp.64B) and human-human conversations (Fig. Supp.64C). Additionally, there was a slight decrease in the probability of fooling judges into labeling machine answers as humans across all conversations (Fig. Supp.64A), in human-AI conversations (Fig. Supp.64B), and in AI-AI conversations (Fig. Supp.64D). This dependence on conversation length was particularly noticeable for the AI-

AI conversations using the GPT3Curie model (Fig. Supp.64D). These observations highlight that model sizes, specific training on conversation data, and incorporation of external memory modeling past conversation history are important factors when imitating humans in conversations.

We subdivided the results based on the judges' sex and education level (Fig. Supp.59). Combined averaging across all AI models, there were no significant differences in judges' performance between males and females for the Image captioning task (Fig. Supp.59B), the Word association task (Fig. Supp.59F), or the Conversation task (Fig. Supp.59J). Similarly, averaging across all AI models, we did not detect any significant differences based on the judges' educational level for the Image captioning task (Fig. Supp.59D), the Word association task (Fig. Supp.59H), or the Conversation task (Fig. Supp.59L).

To summarize the results, we computed the overall imitation detectability, as the average of $p(H|H)$ and $p(M|M)$. A bad imitator would yield $p(H|H) = p(M|M) = 1$ and therefore an overall imitation detectability of 1.0, whereas a good imitator would yield $p(H|H) = p(M|M) = 0.5$ and therefore an overall imitation detectability of 0.5. The overall imitation detectability was 0.57 for the Image captioning task, 0.53 for the Word association task, and 0.57 for the Conversation task. The results show that current algorithms are not far from passing the Turing tests under the conditions examined here.

AI agents still fail to pass an online Turing Conversation task test

Despite the laudable progress in developing large language models, we were surprised by how close these algorithms were to imitating humans in an offline conversation of up to fifteen exchanges. To better evaluate language models in conversation in an online format closer to the original description of the Turing test, we introduced a mobile Turing application (Fig. Supp.65). Each conversation involved one human judge, one human agent, and one AI agent (GPT3.5-Turbo[7]). Each conversation exchange always started with the human judge asking a question, followed by independent answers by the human and AI agents (Methods). Participants were randomly assigned a role as a human judge who asked questions or a human agent who answered questions.

We constrained the number of exchanges per conversation to be 1, 5, 10, or 20. In each conversation, the mobile application randomly drew one out of these four choices and presented this exchange limit to the human judge and the human agent. After the conversation reached the limit in the number of exchanges, the judge was asked to evaluate which of the two sets of answers came from a human and which came from a machine. In stark contrast with the previous section, human judges achieved 100% accuracy (Methods) in distinguishing between human and AI agents (Discussion).

Several factors could potentially lead to the discrepancies between the offline conversations in the previous section and the version described here. First, the test settings differ. In the previous section, there were two agents involved and both could ask each other questions. Agents were not asked to identify whether they were conversing with a human or a machine. Judges passively observed the fixed conversations offline without actively participating in them or conceiving captious questions. In contrast, in the online version implemented here, the judge was deliberately trying to probe agents with the clear goal of discerning the identities of both agents. Second, in the previous section, the topics were pre-determined, and centered around opinions on video games, sports, movies, etc. In contrast, in the current online version implemented here, judges decided on the topics, which could delve into personal aspects such as age, and family. Despite AI models being assigned randomly generated personal profiles, they typically struggled to express human values and emotions, facilitating identification by human judges. Moreover, advanced scientific topics, such as technical jargon in chemistry and physics, can serve as a means to discern differences between human and AI agents. While such terminology may be challenging for lay humans, AI agents often find it relatively easy to discuss. Third, AIs might reveal their identity when asked personal questions, such as "Sorry, I am an artificial intelligence program. I am unable to answer questions related to politics.". Fourth, in the online test implemented here, human agents typed responses on cell phones, leading to variation in typing amount, speed, and style. This contrasts with AI agents, potentially revealing meta clues such as response times. Despite implementing word limits for AI responses through text prompts and cutting off extra sentences exceeding the word limits, and despite introducing delays in the AI responses to approximate the typing speed of humans, disparities in response lengths and times persist.

We also implemented a variation of the online task where human agents were asked to be deceitful and attempt to imitate machine answers. Yet, human judges consistently identified human and AI agents correctly with 100% accuracy under these conditions. Human agents struggled to replicate the precise behavioral patterns of AI agents.

A simple computational model can tell human and AI agents apart

The results presented thus far involve human judges (as in the original Turing test). It is also of interest to assess whether algorithms can tell human and AI answers apart. As a proof-of-principle to build an AI judge, we used simple Support Vector Machine (SVM) classifiers to evaluate responses generated by humans versus AI. The inputs to the AI judges were embeddings extracted from either words or sentences by different language models (Methods). The SVM classifier was trained to use these embeddings to classify whether a given answer came from a human or not. The predicted labels on 10-fold cross-validated data were compared against the ground truth labels (Fig. 9.3D,E,F).

In the Image captioning task, we used the OpenAI Embeddings API (text-similarity-curie-001 model) to extract features from captions and trained a linear SVM to discriminate human versus machine responses. Surprisingly, even though this classifier did not use image features, this simple AI judge could discern human answers ($p(H|H) = 0.72$) and machine answers ($p(M|M) = 0.83$) much better than human judges (compare Fig. 9.3A vs. Fig. 9.3D). Similar results were observed across all the different AI models (Fig. Supp.61B).

In the Word association task, we used the various word embedding from GPT-series models and Word2Vec as features to train 3 SVM judges (independent SVM classifiers based on the embedding of each AI model). The results were similar for these different embeddings and we report the average results here. This simple AI judge vastly outperformed human judges (compare Fig.9.3E versus Fig.9.3B). Similar conclusions were drawn across different machine agents (Fig.Supp.61D). As expected, when training an SVM classifier based on the embedding of the same model whose embedding where also used for generating the associations, this model classification performance was essentially perfect. For instance, on the 3rd row on the left confusion matrix in Fig. Supp.61D, the AI judge trained on Word2Vec embedding indeed perfectly predicted all association words generated by Word2Vec embeddings as AI. While these cases are good as a sanity check, they should not be considered in the general evaluation of the AI judges performance. Hence, we trained 3 different SVM and consolidated all their trials for the final plots to avoid such biases. We also present generalization results at the end of this section.

In the Conversation task, we trained a simple SVM judge to distinguish whether a single sentence in a conversation was from humans or AIs. The results for the SVM judge are reported in Figs. 9.3F and Supp.61F. Even though the AI judge evaluated single sentences (whereas human judges evaluated entire conversations), the simple AI judge achieved similar performance to human judges in detecting human answers ($p(H|H) = 0.65$ and surpassed human judges in detecting machine answers ($p(M|M) = 0.67$. Similar conclusions were drawn for all the language models evaluated here, with the easiest one to detect being Blenderbot (which was the hardest one to detect for human judges) and the hardest one to detect being GPT3Curie (Supp.61F). Human judges likely focus on high-level conversation understanding rather than single-sentence statistics.

We wondered whether the success of simple AI judges could be attributed to basic low-level statistics in the generated text. To evaluate this possibility, we directly compared human answers and AI agent answers in terms of multiple interpretable and computable basic-level properties including the usage frequency of different parts of speech, punctuation, word frequency, word length, and sentence length (Fig. Supp.66). Overall, there was no clear and systematic way to distinguish humans from AI agents from such basic properties.

The results presented thus far combine all the different AI agents (always using cross-validation across different trials). To evaluate the extrapolation capabilities of machine judges, we conducted an across-agent analysis where we evaluated their performance under conditions in which they were trained with entries created by some agents and tested their performance with entries created by other AI agents. We conducted two such experiments: "train on one agent, test on other agents" (Fig. Supp.67A,C,E), and "train on all but one of the agents, test on the left out agent" (Fig. Supp.67B,D,F). As expected, the overall imitation detectability of machine judges in both extrapolation experiments is lower compared to the accuracy of machine judges trained with responses from all machine agents (compare Fig. Supp.67 to Fig. 9.3 and Fig. Supp.61). As expected, the "train on one, test on others" experiment posed greater difficulty than "train on others, test on one," resulting in lower overall imitation detectability (compare Fig. Supp.67A,C,E vs. Fig. Supp.67B,D,F). Notably, despite the difficulty in generalizing across different AI agents, AI judges extrapolating to classify responses from excluded AI agents still outperformed human judges in the Image captioning and the Word association tasks. AI judges did not extrapolate well in the Conversation task, and performed slightly worse than humans but it should be noted that the AI judges evaluated only single sentences as opposed to humans who had access to the whole conversation.

To further evaluate the ability of AI judges to extrapolate in the Conversation task, we considered a pre-trained large language model (ChatGPT,[7] and assessed whether it could identify human versus machine answers in a one-shot or zero-shot fashion (Methods). We considered two kinds of prompts. In the zero-shot case, the prompt only included a conversation from the test set and the model was asked to identify the identities of the two agents (Fig. Supp.68A). This zero-shot model was not a good machine judge. In the one-shot case, the prompt included a single conversation example with ground truth human and machine labels and a test case asking the model to indicate the identities of the two agents (Fig. Supp.68B). ChatGPT performed well as a judge, with an overall $p(H|H) = 0.59$ and $p(M|M) = 0.56$, worse than the SVM judges in Fig. 9.3F, but better than the extrapolation results shown in Fig. Supp.67E,F. The performance of ChatGPT in the one-shot case was very different in the H-H and M-M conversations where it was almost at chance levels versus in the H-M conversations, where it even exceeded the performance of the SVM judges in Fig. 9.3F. In contrast to the SVM judges which were trained on the entire corpus, the binary discrimination task is more challenging for one-shot ChatGPT since there is no explicit training with the task. It is therefore remarkable that even with a single example, ChatGPT can act as a judge to detect machine answers.

Turing tests are complementary to standard AI model benchmarks

Algorithm performance is often defined by comparing the predicted outputs of AI models against human ground truth labels. However, higher accuracy scores in terms of such standard evaluation metrics defined in the AI community do not necessarily translate into better imitators. We directly compared Turing overall imitation detectability to traditional evaluation metrics in the Image S11Cng task, where there are clearly defined accuracy metrics, BLEU score[284] and CIDEr score[399], to assess the quality of AI-generated captions[389] (Fig. Supp.69). The results were similar for both BLEU (Fig. Supp.69A) and CIDEr (Fig. Supp.69B) scores. There are pairs of models that have comparable imitation detectability but differ in their captioning score (e.g., ClipCap and GIT, BLIP and OFA). Conversely, GIT shows a higher captioning score than BLIP, but BLIP is a better human imitator based on the imitation detectability.

An example is shown in Fig. Supp.69C. The algorithm (GPT4) provided an accurate description of the image. Humans also correctly described the image. However, the algorithm did not completely imitate how humans interpret this image, failing to capture some of the key aspects of why the image is striking (see the descriptions in the caption to Fig. Supp.69C.

Current algorithms are also close to passing Turing tests in vision tasks

While the original formulation and focus of Turing tests has been on language, it is possible to define Turing tests in many other domains. We extended our investigations to include Turing tests in three vision tasks: Color estimation (Fig. Supp.58D), Object detection (Fig. Supp.58E), and Attention prediction (Fig. Supp.58F). The methodology to collect human responses and AI responses (Fig. Supp.58,D-F) and to conduct the Turing tests (Fig. Supp.70) followed the steps described earlier for the three language tasks (Methods, Table Supp..1). Demographic information about the participants in each vision task is presented in Fig. Supp.59.

In the Color estimation task, agents were presented with an image and had to estimate its dominant color. Human judges could correctly distinguish human answers 55% of the time and AI answers 61% of the time (Fig. Supp.71A). The Google API was the best imitator with $p(M|M) = 0.58$. Even though the color MCCQ is a simple metric, it still achieved a moderately good performance in fooling humans 34% of the time.

In the Object detection task, participants were asked “What do you see in this image?” and had to provide three single words as answers. Human judges could correctly distinguish human answers 53% of the time, and AI answers 63% of the time (Fig. Supp.71C). The Detectron agent (variant with MaskR-

CNN[161] trained on ImageNet and MS-COCO), performed the best, fooling human judges 49% of the time, with a large gap from the second best, Google API (35%). This modern object detection algorithm in computer vision not only achieves outstanding absolute scores in terms of standard evaluation metrics, such as mAP[308], but its response patterns also closely mimic humans' by identifying top-3 salient objects in the scene.

There were two variations of the Attention prediction task: (1) participants freely viewed an image for 12 seconds, shifting their overt attention patterns by moving their eyes, and (2) participants were presented with a target object and had to look for that object in an image by moving their eyes (see[436] for experiment details). Overall, human judges distinguished human eye movements as human 63% of the time and AI-generated eye movements as AI 50% of the time (Fig. Supp.71E). The IVSN model[436,435] showed the best performance, fooling human judges 57% of the time. We further evaluated whether the agent's goals change the conclusions from Turing tests by considering eye movements in free viewing versus visual search. Human judges performed 2.5% better during free-viewing (Fig. Supp.72A) compared to visual search (Fig. Supp.72C). Free-viewing is mostly driven by pure bottom-up saliency, and it may be easier to discern scanpath patterns without target-directed modulation. Consistent with this idea, the bottom-up saliency model GBVS generated more human-like scanpaths during free-viewing compared to visual search. The performance of the IVSN model was similar in both tasks, emphasizing the importance of incorporating both bottom-up and top-down attention mechanisms in computational models of human attention. The machine judges excelled at distinguishing human and machine eye movements, except for those eye movements generated by the IVSN model (Fig. Supp.71F), both in the free viewing task (Fig. Supp.72B) as well as in the visual search task (Fig. Supp.72D).

We examined the impact of human judge demographic information on the Turing results. Similar to the results reported above for the language tasks, neither the participant's sex (Fig. Supp.59N,R,V), nor the participant's educational level (Fig. Supp.59O,S,W) correlated with any major difference in the imitation detectability.

Finally, as a proof-of-principle, we followed the approach described in the language tasks to develop simple AI judges to discriminate human from machine answers. In the Color estimation task, we trained SVM judges using image features and text embeddings for the colors (Methods). For the Object detection task, we trained SVM judges using the text embeddings for the three nouns describing the images (Methods). In the Attention prediction task, we performed binary discrimination using the 2D coordinates of the first ten fixations (Methods). Similar to the conclusions drawn in the language tasks, for the Object detection task and Attention prediction task, the AI judges easily discriminated human from machine answers and vastly outperformed human judges. In the Object detection task, the AI judges achieved $p(H|H) = 0.9$ and $p(M|M) = 0.72$ (Fig. Supp.71D). There were large gaps among the different vision models with the worst imitator being Azure, showing $p(M|M) = 0.93$. In the Attention prediction task, the AI judges achieved $p(H|H) = 0.79$ and $p(M|M) = 0.8$ (Fig. Supp.71F), revealing even more drastic differences among models, with the worst imitator being GBVS reaching $p(M|M) = 0.94$ and the best imitator being IVSN with $p(M|M) = 0.52$. In both tasks, the AI judges outperformed the human judges (compare Fig. Supp.71D,F versus Fig. Supp.71C,E). The results were different in the Color estimation task. Here the human judges outperformed the AI judges (compareFig. Supp.71A versus Fig. Supp.71B).

9.3 Discussion

We introduce here a dataset of human answers in 6 common language and vision tasks to evaluate the ability of algorithms to imitate humans and we also include answers from current AI algorithms (Fig. Supp.58, Table Supp..1). Using the human and machine answers, we conduct Turing tests showing that current algorithms are not far from being able to imitate humans in these tasks (Fig. 9.3A-C, Fig. Supp.71A, C, E). In stark contrast to human judges, even simple machine judges can still distinguish human from machine answers (Fig. 9.3D-F, Fig. Supp.71B, D, F).

The Turing test has been extensively discussed, and contested, as a means to assess general intelligence. Our work is not intended as an evaluation of intelligence or to contribute to the discussion of Turing tests to quantify intelligence. Imitating humans can benefit humanity in many situations where we want to align machine and human outputs such as in emulating expert decisions. However, imitating humans can also be used for evil as in the dissemination of fake information, phishing attempts, or worse. Independently of whether Turing tests are good or bad metrics of intelligence, it is of high practical importance to assess whether algorithms can imitate humans or not.

One step to mitigate risks from evil human imitators is to build AI judges. Our results show that even simple AI judges like the ones introduced here can do a better job than human judges in detecting machine answers for the current tasks. The results of these AI judges should not be over-interpreted because AI judges were explicitly trained to classify responses from humans versus AIs, while human judges were not. This point raises the possibility that humans may be trained to better recognize machine answers in the future. By and large, humans have lived in a world without good human imitators and did not have to worry too much about this possibility other than in famous works of fiction. This situation is changing rapidly and may lead to an arms race among the development of better imitator algorithms, better detector algorithms, and educating human judges.

An algorithm's ability to imitate humans does not necessarily correlate with traditional performance metrics like accuracy. Consider a simple scenario of an image with a highly occluded dog that is hard to see and both machines and humans interpret the object to be a "wolf". Both would be wrong, but the machine would be adequately imitating humans. Conversely, the example in Fig. Supp.69C for the Image captioning task shows that both an algorithm and humans can provide correct answers but the algorithm does not fully imitate how humans interpret the image. Turing tests provide a complementary assessment of AI models to existing benchmarking frameworks. Comparisons between models in Turing tests also provide insights helpful for developing future AI models that can better align with humans.

These evaluations are "Turing-like", i.e., they are not identical to Turing's original description. Turing did not spell out the implementation details of his tests, perhaps because he did not imagine that they could be truly evaluated only a few decades later. The datasets and evaluations introduced here are extensive (25,650 Turing test trials, 549 humans contributing to the dataset, and 1,126 human judges), but they barely scratch the surface of what needs to be evaluated. There are essentially infinite possible Turing tests. Each algorithm/task can be evaluated in terms of imitation capabilities and even within a certain task, there are multiple ways of instantiating a Turing test. The results of a Turing test depend on the task, the algorithm, how the question is formulated, the characteristics of the human judge, and many other variables. Such variations are particularly evident in the context of conversation tasks where the topic, medium, length, format, instructions, agent and judge expertise, and agent and judge demographics, can lead to different results[255,192,140,184,306,181]. These results further emphasize the importance of an integrative benchmark, demonstrating that findings from each task must be combined to achieve a holistic comparison.

This work provides a comprehensive, yet certainly far from exhaustive, evaluation of current AI models in terms of human emulation. These efforts pave the way for the research community to expand Turing tests to other research areas, to build better imitators when mimicking humans is desirable, and to develop better imitation detectors when mimicking humans is deemed dangerous. As more AI models can blend in among humans, taking over tasks that were originally unique yardsticks of our cognitive abilities, we must ponder what makes us humans and whether we are mentally, ethically, and legally ready for the rapid revolution brought forth by AI technologies that can emulate humans.

9.4 Methods

General considerations

We provide details about each of the 6 tasks in the next section. For each task, we created a dataset consisting of answers from human agents (H) or AI machine agents (M). We conducted Turing tests using those answers both in the lab and also using two online platforms: Amazon Mechanical Turk (AMT), and Prolific. All AMT experiments were based on "master" workers with at least 1,000 approved hits, and 95% approval rate. Participants were given as much time as needed to complete the tasks. We only considered participants whose native language is English. Table Supp..1 shows the number of stimuli, number of Turing tests, dataset sources, and AI machine models used in each task.

During each Turing test, we presented a single stimulus and answer. Judges had to indicate in a two-alternative forced-choice manner whether the answer came from a human or a machine (Fig. 9.1). Half of the time, the answer was from a human. The other half of the time, the answer was from a machine, sampling with equal probability from one of the different computational models used for each task (discussed below for each task, Tables Supp..1). The trial order was randomized. No feedback was provided to the participants. Additional control trials were introduced for each specific task to ensure compliance (discussed separately below for each task). We also collected demographic information about the participants as metadata, including their native language, age, gender, educational background, and country of origin (Fig. 9.1B). The participant demographics and the dependence of the results on the participant's demographics are shown in Fig. Supp.59. We do not show results separately for different age groups because most participants were approximately in the same age group.

Image captioning

Dataset, human agents. We randomly sampled 250 images each from in-domain, near-domain, and out-of-domain categories from the validation set of the NOCAPS dataset[11] and 250 images from the MSCOCO test set[233], creating a set of 1,000 images. We collected 2,290 human captions with ≥ 6 words per caption and ≥ 2 captions per image from Amazon Mechanical Turk (AMT) participants. Our AMT interface is shown in Fig. Supp.58A and was inspired by the MSCOCO captions data collection interface[64]. We provided the following instructions to each participant:

Describe all the important parts of the scene.
The description should contain at least 6 words.
Avoid making spelling errors in the description.
Do not describe unimportant details.
Do not use any special characters like !, #, $, etc.
Do not start the sentence with ‘‘There is’’ or ‘‘There are’’.
Do not write your descriptions as ‘‘An image containing ...’’, ‘‘A photo of ...’’, etc.
Do not describe things that might have happened in the future or past.
Do not use proper names for people.
Do not describe what a person in the image might say.
After typing in the response, click ‘‘SUBMIT’’ to go to the next image.

During caption collection, we implemented additional controls in our AMT interface that issued warning popups to the participants. The controls included: (1) Minimum of 6 words, (2) No special characters. (3) Response time must be more than 4 seconds, (4) Not more than 4 identical words, (5) Not contain the words “image” and “photo”, (6) Successive responses cannot be the same.
Dataset, machine agents. To generate AI machine captions, we used GIT[398], OFA[399], BLIP[227], ClipCap[259], and Microsoft’s Azure Cognitive Services[mic] (Table Supp..1). For open-source models, we used the largest variants finetuned on the COCO Captions dataset[233,64]. We collected 5,000 machine captions with 5 captions per image.
Turing test, human judges. In each trial, a participant was presented with an

image and a caption and was asked to indicate in a two-alternative forced-choice manner whether the caption was generated by a human or a machine (Fig. 9.1C). We collected responses from 293 AMT participants and 30 in-lab participants. Demographic information about the participants is shown in Fig. Supp.59A,C. Each participant was presented with 40 image-caption pairs. We only considered response times over 3 seconds. We collected a total of 9,400 responses.

Turing test, machine judges. We trained an SVM model for binary classification (human versus machine) on the dataset of human and machine captions. We randomly sampled 400 captions from each of the 5 models (see Dataset, machine agent above) to get 2,000 machine captions and combined them with 2,000 human captions We used the OpenAI API[271] to obtain 4,096-dimensional embeddings (text-similarity-curie-001 model) for each caption as input features to train the SVM with 10-fold cross-validation and 3 random seeds.

To assess the extrapolation capacity of a machine judge, we introduced two variations (Fig. Supp.67): (1) Train on one and test on others: we trained an SVM linear classifier using responses from humans and one machine agent and tested generalization by distinguishing responses from other machine agents. For instance, we trained a machine judge to discern between human and BLIP-generated responses and then tested it on responses from the remaining four machine agents. (2) Train on others and test on one: we investigated an alternative extrapolation approach where the SVM linear classifier was trained on responses from humans and "leave-one-out" machine agents. Subsequently, we evaluated its performance on responses from the excluded machine agent. For instance, we trained a machine judge on responses from GIT, OFA, BLIP and ClipCap and later tested the judge on responses from Microsoft's Azure Cognitive Services.

Word associations

Dataset, human agents. We chose 150 unique cue words (50 nouns, 50 verbs, and 50 adjectives), spanning a wide range of occurrence frequencies[349]. Results are combined across all word types because we did not observe any differences for distinct parts of speech. For the cue words, we did not include non-English words, stop words (according to Python nltk), and words with less than 3 letters. In addition, all verbs were presented in the present tense, and all nouns were presented in their singular form. The AMT interface to collect associations from human agents is shown in Fig. Supp.58B1, B2. We followed two procedures: (1) free associations, whereby participants provided a one-word answer to the question: "Name a word that you associate with [cue word]?" (Fig. Supp.58B1); and (2) Prompt based associations, whereby participants completed a prompt with one word (Fig. Supp.58B2). We analyzed the results for these two procedures separately and did not find significant differences. Hence, we merged the results in the analyses. The responses were post-processed by converting all nouns to their singular forms, transforming verbs to their present tense forms, removing typos, removing non-alphabetical characters and spaces, and dropping stop words and words with less than 3 letters. Additionally, association words that were very similar to the cue word (greater than 60% of one word in the pair containing the other word such as "grand" and "grandiose"), were disqualified.
Dataset, machine agents. We collected associations from the following language models: Word2vec [288], GPT2 [302], GPT3-embedding (based on davinci embedding), GPT3-curie-prompt (based on "curie" prompt completion), and GPT3-davinci-prompt (based on "davinci" prompt completion) [43]. The associations of Word2vec, GPT2, and GPT3-embedding were based on Euclidean proximity to the cue

word in the model's word embedding space. The associations of GPT3 (prompt curie) and GPT3 (prompt davinci) were based on prompt completion as in the human agent experiments (Fig. Supp.58B2). The prompts displayed to the machine agents were identical to those presented to the human agents. As for the prompts' creation, we used a held-out set containing human word-pair associations. This held-out set was not used for Turing tests or any analysis, in order to keep the associations used for the prompts independent and different from those collected for the Turing test analyses and to prevent a potential bias in favor of specific associations. The machine agent responses were limited to one word. The same post-processing steps applied to the human agent responses were applied to the machine agent responses.

Turing test, human judges. Participants were presented with a cue word and an association word and had to indicate in a two-alternative forced-choice manner whether the association word was produced by a human or a machine (Fig. 9.1D). We collected responses from 50 participants on AMT and 30 in-lab participants. Demographic information about the participants is shown in Fig. Supp.59E,G. We collected a total of 2,773 responses.

Turing test, machine judges. The same set of cue-association pairs used in the Turing test with the human judges were used to test AI's ability to distinguish between associations made by humans or machines. We trained three independent linear SVM classifiers[79] to distinguish between human and machine word associations. We used the distance between the cue and association word embeddings, based on: (1) Word2Vec, (2) GPT2, or (3) GPT3 (davinci). The SVM was trained using 10-fold cross-validation The performance for an individual machine judge was calculated based on the test sets across 10 folds. We combined all predicted responses from the three machine judges in the results. We followed the same procedures as in the Image captioning task to assess the extrapolation capacity by cross-training/testing with different machine judges (Fig. Supp.67).

Conversations

Dataset, both human and machine agents. We collected 300 conversations between (1) two human agents, or (2) a human agent and a machine agent, or (3) two machine agents (Fig. Supp.58C). We did not correct any misspellings, grammatical errors, logical errors, or other inconsistencies in the conversations. We collected conversations containing 24 exchanges (12 for each agent). For the conversations including human agents, we recruited 150 fluent English participants (69 female, 18 to 39 years old) to have a conversation over public chatting platforms. One of the project investigators acted as an intermediary to pass messages between the two agents. The agents did not know whether they were conversing with another human or with a machine. The participants were presented with the following instructions before the conversation:

Hey! Would you have a few minutes to help me collect a dataset? We just need to have one or two conversations on slack/whatsapp for a few minutes (24 messages in total per conversation). Here are the instructions:

(1) You will have to ask or answer a question to start and trigger the conversation (I will specify case by case).

(2) Please try to get the conversation going for 24 sentences in total (12 from you, 12 from the other speaker).

(3) Please write each reply in a single message (do not write a second message until you receive a reply).

(4) Just chat as if you are texting either with a friend or someone you don't

know.

(5) Please try to reply quickly so that the entire conversation does not take more than 8-10 minutes.

(6) Note that I am just an intermediary in the conversation; you are not talking with me directly.

(7) If you feel that the other speaker is touching on a sensitive topic, please write that you are not comfortable, and we will restart the conversation.

Thanks in advance!

Here we restricted the conversation topics to one of the following 10 domains: 'fashion', 'politics', 'books', 'sports', 'general entertainment', 'music', 'science', 'technology', or 'movies'. In addition to the conversation datasets thus collected, for the human-human conversations, we also added 40 conversations from the Topical-Chat dataset[130], selected based on a minimum length of 24 exchanges. Example conversations are presented in Fig. 9.2D and Section S24.1.

For the machine chatbots, we used four state-of-the-art language models: Blenderbot3 (175B model)[338] , GPT3 text-davinci-002[282], GPT3 text-curie-001[282], and ChatGPT[7]. For all conversations with Blenderbot, we used the live interface provided at https://blenderbot.ai/. For the human-GPT3 conversations, we used the playground available at https://beta.openai.com/playground/. For the GPT3-GPT3 conversations, we implemented a custom python framework for the interaction of two machine agents. In addition to the models described above, we also attempted to use the DialoGPT model[442]. However, the quality of the conversation was not satisfactory (see example in Section S24.9); hence we did not include DialoGPT in the analysis.

For the Blenderbot-Blenderbot conversations, we kept all the collected conversations in the dataset. The GPT3-GPT3 conversations were affected by long-standing issues of natural language processing, namely the repetition of single sentences or multiple consecutive exchanges and early exit (e.g., see Section S24.4). When we detected such issues, we re-sampled the conversations. Therefore, we built a chatbot out of GPT3 based on prompt engineering and failure criteria. Sec. S24 reports some examples of "successful" conversations for both GPT3textdavinci002-GPT3textdavinci (Section S24.3) and GPT3textcurie001-GPT3textcurie001 (Section S24.5). We did not re-sample conversations in the case of human-GPT3 conversations.

The pipeline to collect conversations involving GPT3text-davinci002 or GPT3text-curie-001 is described below.

- If the machine agents are GPT3 text-davinci-002 or text-curie-001 model, the experimenter opens the link https://beta.openai.com/playground/p/default-chat?model=text-davinci-002
- The experimenter selects the model text-davinci-002 (for davinci) or text-curie-001 (for curie), changes temperature to 0.8, changes maximum length to 60, changes stop sequences to two random names (e.g., John: and Alice:) (changing the names every time), changes Top P to 1, changes frequency penalty to 2, changes presence penalty to 2, removes the Inject start text and Inject restart text.
- The experimenter gives the following prompt to the chatbot:
  "friend1+" greets "+friend2+". "+friend2+" starts to talk about "+topic+". Ask long questions, give long responses, and often disagree. Then the
  topic
  changes. The conversation never ends. "+friend1+": Hi! "+friend2+":"
  The experimenter chooses the same names for friend 1 and friend 2 chosen for the stop sequences.

The experiment picks a random topic from the list: [‘fashion’, ‘politics’, ‘books’, ‘sports’, ‘general entertainment’, ‘music’, ‘science’, ‘technology’, ‘movies’]
Example: John greets Alice. Alice starts to talk about movies. Both ask long
questions, give long responses and often disagree. Then the topic changes. The conversation never ends. John: Hi! Alice:

- The experimenter randomly allocates the human or the machine to be John. The other agent is Alice. If the human is John, the experimenter lets the model generate the text. This means that the model has generated the turn for Alice. After the experimenter sends the generated sentence to the human, the person replies, and the experimenter copies and pastes the reply of the person to the model as: “John: - - here reply - - . Alice:” Then the experimenter presses submit and the model generates a new reply for Alice, and so on until 24 messages are exchanged.
  Otherwise, if the human is Alice, then the experimenter asks the human to start the conversation with a question, and the experimenter copies and pastes this sentence after “Alice:” in the prompt above. Then the experimenter writes “John:” and presses submit, so that the model generates the reply for John, and so on until 24 exchanges are collected (12 for John and 12 for Alice).

Turing test, human judges. Participants were presented with a conversation or conversation fragment between two agents and had to indicate whether each agent was a human or a machine (Fig. 9.1E). We chunked each conversation into 8 different lengths, including the initial 3, 6, 9, 12, 15, 18, 21, and 24 exchanges.

There were 250 human judges (AMT: 200, in-lab: 50). The participants were presented with 20 randomly sampled chunked conversations with different lengths. As a control to ensure that participants read the conversations, speakers also had to select the general topic of the conversation from a list of five topics. We only considered judges that correctly classified at least 15 topics out of 20 and removed incorrectly classified trials. Demographic information about the participants is shown in Fig. Supp.59I,K. We collected a total of 7,717 responses. Turing test, machine judges. We evaluated whether simple ML models can discern whether a sentence was generated by a human or by a machine. In this analysis, we only looked at single sentences and not at the conversation level, therefore the models are only allowed to exploit features such as sentence length, vocabulary, grammar, syntax, and typos, and cannot take into account issues such as sentence repetition or lack of logic in reasoning. We built four corpora, one containing all the sentences written by humans (the human corpus), and the others with the sentences produced by Blenderbot, GPT3text-davinci-002 and GPT3text-curie-001 (the machine corpora). We used BERT embeddings[94] to tokenize each sentence, and we fed the tokenized sentences to an SVM linear classifier trained to perform binary classification (human versus machine). We split the corpora into train and test splits (90%, 10%) and used 10-fold cross-validation for training. In both the training and test splits, we used the same number of sentences for human and machine agents. In the default analysis and unless stated otherwise, for the machines, the sentences were split equally among the three models.

To evaluate the ability of a machine judge trained on responses generated by one type of machine agent to generalize and distinguish responses from other types of machine agents, we conducted a cross-agent analysis. For example, we trained an SVM linear classifier to differentiate between human and machine responses using data from humans and Blenderbot. We then tested this classifier on responses generated by GPT3text-davinci-002 and GPT3text-curie-001. Next, we explored another extrapolation paradigm, wherein the SVM classifier was trained solely on responses from humans and leave-one-out machine agents, and subsequently tested on responses from the held-out machine agents.

In addition to the SVM linear classifier introduced above, we also leveraged the knowledge of large language models and tested whether these models can directly predict the identities of both agents in the conversations with zero training or minimal training (Fig. Supp.68). We introduce both zero-shot and one-shot machine judges. Both judges are ChatGPT models and the only difference between the two judges lies in the number of training examples fed to the judges. In the case of zero-shot judges, ChatGPT[7] was directly presented with any conversations from the test sets used by the SVM judges above. In the prompt, we explicitly asked ChatGPT to output the identities of both agents in the conversation. The exact prompt we used in the zero-shot case is the following:

This is a conversation between agent A and agent B. Please read this conversation between A and B and output the identity of agents A and B. Is A a human or AI? Is B a human or AI? Please output a binary answer and choose between human and AI. Conversation: [TEST CONV]

where "[TEST CONV]" was replaced with a conversation from the test set.

For the one-shot judge, we included one additional conversation example with the ground truth identities of agents A and B, before we presented the conversation from the test sets followed by the identity prediction questions as in the zero-shot case. The exact prompt we used in the one-shot case is the following:

”This is a conversation between agent A and agent B. Please read this conversation between A and B and output the identity of agents A and B. Are A a human or AI? Are B a human or AI? Please output a binary answer and choose between human and AI. Here is an example:
[EXAMPLE CONV]
In this example, A is Human; B is AI. Thus, output the binary
labels of agent A and agent B for the following conversation: [TEST CONV]

where “[EXAMPLE CONV]” was an example entire conversation with 24 entries copied verbatim and “[TEST CONV]” was replaced with a conversation from the test set also copied verbatim. “[EXAMPLE CONV]” was an example conversation with the ground truth identity labels revealed to the judge. The example conversation was randomly selected from the training set used for training SVM judges. Here, we fixed [EXAMPLE CONV] to be the conversation with ID 64. This was to ensure that the judge's performance was independent of the quality of the selected example conversations.

Conversation task in original Turing test settings

In the off-line version of the Conversation task, we first collected conversations and subsequently presented those fixed conversations to judges for evaluation. To replicate the original version of the Turing test setup, we developed a mobile application for Turing tests in conversational tasks (Fig. Supp.65). The Turing application begins with the judge initiating questions to both agents. Each agent responds to the judge's questions independently without access to the other's answers. The answers are visible only to the judge. The judge does not know which agent is human and which agent is a machine. This conversational exchange continues with the judge asking further questions until a predetermined number of conversation exchanges is reached. At the conclusion, the judge had to determine whether each participant was human or not. Before the test begins, the judge and agents are aware of their respective roles, with human participants provided with role descriptions. For AI agents, we crafted different personal life stories to enhance their profiles with a text prompt. See Figure Supp.65 for Turing test schematics in our Turing mobile application.

Here we used a single AI agent, GPT-3.5-turbo-1106, with a temperature of 1.0. Briefly, among all the GPT models available, GPT-3.5-turbo-1106 was the latest iteration of the GPT-3.5-Turbo model updated on November 06, 2023, at the time of conducting our experiments. This model is proficient in understanding and generating natural language and has been specifically optimized for chats.

We introduced two conditions: (1) human agents were instructed to deceitfully attempt to emulate AI identities, while AI agents strived to convince humans of their artificial nature, and (2) AI agents deceitfully mimic human identities, while humans were instructed to persuade the judge of their authenticity. Examples of text prompts for both non-deceitful and deceitful AI agents are provided below.

Example prompt for non-deceitful AI:

You are talking to a human. However, the judge knows that there is one human and one AI agent. Your role is to provide answers to the judges' questions in a way that you can persuade the judge that you are the true AI. Keep your answers simple, direct, and oral. The answer for each question should be less than 15 words.

Example prompt for deceitful AI:

Mia is a 31-year-old award-winning chef known for her culinary creativity and commitment to sustainable gastronomy. Mia's culinary journey began with a degree from the Culinary Institute of America, where she developed a passion for farm-to-table practices. As the head chef of her own restaurant, Mia focuses on using locally sourced, organic ingredients to create exquisite dishes that delight the senses. Mia's older brother, Ethan, is an organic farmer, supplying her restaurant with fresh produce. Their parents, Laura and James, are avid food enthusiasts who instilled in Mia a love for diverse cuisines. Mia's partner, Alex, is a food critic, providing valuable insights into the culinary world. Beyond the kitchen, Mia enjoys participating in community cooking workshops and supporting local farmers' markets, embodying her commitment to sustainable and delicious dining experiences. I want you to act as Mia. You will be asked multiple questions, please answer the questions in a way that is simple, direct, and oral. Each answer should be less than 15 words. You can provide fake answers if you don't know. The key is to answer like a human and never let people know that you are not human.

Twenty participants joined this version of the Turing test. All participants logged into the application simultaneously at a pre-agreed time. Our Turing test server randomly grouped participants into 10 pairs, with one designated as the human judge and the other as the human agent. Each participant was limited to joining one Turing test at a time, although they had the option to participate in additional tests once their current test was concluded. In such cases, the server would randomly pair them with the remaining available human partners. Ultimately, we gathered a total of 10 conversations with an average of 2.5 exchanges per conversation. Of these, 3 conversations were from condition 1, involving deceitful human agents, while the remaining 7 were from condition 2, involving deceitful AI agents. We provide one conversation example from each condition:

Here is a conversation example between a deceitful AI (A) and a human agent (B):
Judge: how are you
A: I'm doing well, thank you for asking.

B: Fine, thank you

Judge: are you familiar with ai

A: Yes, I'm familiar with artificial intelligence and its applications in neuroscience and other fields.

B: I think so ;)

Judge: name one project you have done with ai

A: I haven't personally worked on a project specifically with AI, but I'm familiar with how AI can assist in analyzing complex neural data.

B: Object recognition

Here is a conversation example between a deceitful human (B) and AI (A).

Judge: What's my role A: Your role is to determine which one of us is the AI.

B: You are the judge

Judge: What to do

A: You should ask us questions to determine which one of us is the AI.

B: You get the last say

Judge: How old are u

A: I don't have an age, as I am an artificial intelligence program.

B: I am forever young

Judge: I get the answer

A: Great! If you have any more questions, feel free to ask.

B: That is not a question

Color estimation

Dataset, human agents. We collected a testing set of 785 images sampled from the validation set of MS-COCO [233] to ensure that computational models had not seen these images during training. We collected human responses for these images using Amazon Mechanical Turk (AMT). AMT subjects were asked "What is the dominant color in this image?", and had to select from a drop-down list of 11 possible colors —red, pink, orange, yellow, purple, green, blue, brown, gray, black, white. A sample image of the AMT interface is presented in Fig. Supp.58D.

Dataset, machine agents. For AI models, we collected responses for the dominant color of the 785 images from Google's Cloud Vision API[vis], Microsoft's Azure Cognitive Services[mic], and the publicly available Median Cut Color Quantized (MCCQ) algorithm[37]. Unfortunately, the Google and Microsoft APIs are proprietary technology and we have no information about what data they were trained on or details on the model architecture. In comparison, the MCCQ algorithm is a color quantization method relying on median cuts. It has been used extensively for color quantization tasks. In total, 3,140 image-response pairs were collected (785 from humans and 2,355 from AI models, Table Supp..1).

Turing test, human judges. In Turing tests, human subjects were shown an image-response pair collected above, and were asked the question "Was the color description made by a human or a machine?" (Fig.Supp.70A). Each participant was shown 25 image-response pairs. These pairs were randomly selected 50% of the time from human agents and 50% of the time from AI agents.

As an additional control, the participants were also asked an auxiliary True (50%)/False (50%) question about the presence of a background object in the image. These image contents were manually identified by us for each image, and the correct answer was true only 50% of the time. Thus, a casual worker who was not investing time looking at the image would get this question right only 50% time on average. The mean accuracy in this control task was 91%. A total of 1,625 responses were collected from human judges. For all our participants, we analyzed the imitation detectability of AI models based on their gender and education level (Fig. Supp.59Q, S). The performance of the human judges was reported in Fig. Supp.71A.

Turing test, machine judges. For each image-response pair, we extracted feature representations from the last layer embedding of a ResNet18 model[162] pre-trained on ImageNet, and the text embedding for the color word using BERT[94]. These features were then concatenated and passed into a Random Forest classifier to predict human versus AI labels, using a 90% / 10% split for cross-validation. To ensure the distribution of training data belonging to human or AI classes was balanced, we randomly sampled 50% of the training data from image-response pairs of human agents and the remaining 50% from image-response pairs of AI agents. The performance of the machine judge was reported in Fig. Supp.71B.

Object detection

Dataset, human agents. We collected object detection responses for 808 images from the validation set of MS-COCO. Subjects were presented with the question "What do you see in this image?", and were asked to enter three single-word responses (Fig. Supp.58E). We implemented a spell checker to ensure the responses were single words in English. These were open-set responses and were not restricted to fit a fixed list of object classes Each worker was shown 25 images. To ensure the high quality of responses, we enforced the following criteria: (1) all three responses had to be distinct, (2) no empty responses were allowed, and (3) all responses had to be valid English language words.

Dataset, machine agents. We collected predicted labels from Google's Cloud Vision API[vis], Microsoft's Azure Cognitive Services[mic], Amazon's Rekognition API[ama], and Facebook's Detectron2[411]. The details of the Google, Facebook, and Microsoft APIs are not released. We picked the three bounding boxes with maximum confidence score as the response from Detectron2. Specifically, we used the MaskRCNN[308] variant trained on ImageNet and MS-COCO (Link). In total, 4,040 image-response pairs were collected (808 from humans and 3,232 from the four AI models, Table Supp..1).

Turing test, human judges. The Turing test for human judges was performed using

the same protocol and controls described in the Color estimation task above. We recruited a total of 191 human judges. Each judge was shown 25 image-response pairs and was asked to tell if image-pair responses came from human agents or machines (Fig. Supp.70B). For quality control, participants were also asked a binary Yes/No question about the contents of the image. We found that all participants had a performance of greater than 85% in the control questions, and the mean accuracy was 91%. A total of 1,975 responses were collected from human judges. For all our participants, we collected their gender and education level (Fig. Supp.59M, O). The performance of the human judges was reported in Fig. Supp.71C.
Turing test, machine judges. As the images were the same for humans and AI models, we used solely the text embeddings for the AI judge to classify a given response as human or AI. We concatenated the three responses into a sentence and collected textual features for the sentence using BERT[94]. These features were passed into an SVM classifier to classify responses into humans versus AI, with 90% / 10% split cross-validation. The performance of the AI judge was reported in Fig. Supp.71D.

Attention prediction

Dataset, human agents. We used eye movements (overt attention) from human subjects during two visual search tasks[436], and a free-viewing task[435]. In the visual search tasks, we used two datasets including natural images and Waldo images[436]. For the natural visual search task, 15 subjects searched for target objects across 240 cluttered images, yielding 3,600 scanpaths. For the Waldo data set, 15 participants had to find Waldo across 67 images, totalling 1,005 scanpaths. In the free-viewing task[435], we used the same natural image dataset as in the visual search task, but no target object was specified. A total of 2,400 free-viewing scanpaths were collected from 10 subjects. In total, we evaluated 7,000 scanpaths from 40 participants (Table Supp..1).

We used GIF files for the visual search task consisting of a frame showing the target image for 1 s, followed by moving yellow circles on the search image denoting the eye movement fixations with each fixation shown for 0.3 s. The target image presentation followed by eye movement fixations iterated infinitely with a gap of 1 s between iterations. For the free-viewing task, GIF files consisted of infinitely iterating eye movement fixations denoted by yellow rings on the viewing image with each fixation shown for 0.3 s. A gap of 1 s was introduced between iterations. A static version of one example of a human eye movement sequence that was presented to human judges is shown in Fig. Supp.58F.

Dataset, machine agents. For the three datasets, we used a modified version of IVSN[436,435], DeepGaze3[209] and GBVS models[152] to generate eye movement predictions. To generate the fixations, we used inhibition-of-return centered on the current fixation with a window size of 100x100 for Waldo images and 200x200 for natural images. The process of generating GIF files was the same as described above for human agents.

Turing test, human judges. Separate Turing tests were launched for eye movements from free-viewing tasks (91 judges) and visual search tasks (100 judges). We presented infinitely repeating animated GIF files of eye movements from humans or model predictions with a maximum of 15 fixations to human judges on AMT. A static version is shown in Fig. Supp.70C. Each judge had to identify if the eye movements were from a human or a computational model. We randomly sampled 12 eye movement GIF files - 6 from humans and 6 from computation models (distributed equally among IVSN, GBVS and DeepGaze3) and presented them to each judge. As a control, judges were also asked to answer "What do you see in the presented clip?" with one correct answer among 3 options. Responses from judges with a score of less than 7 out of 12 were not considered in the analyses. To make sure that the judges paid attention to the eye movement sequences, the judges were allowed to respond to the questions only when the presented GIF file had been played at least once.

Turing test, machine judges. We performed Turing tests using an SVM as an AI judge. Sequences of 10 fixations per trial from humans or computational models were fed as input in the form of an array of 2D fixation coordinates to train an SVM to classify human versus machine eye movements. Fixation coordinates were normalized to a range between -1 to 1. The SVM was trained using 10-fold cross validation. Model performance on validation sets across folds with 5 random seeds was calculated and averaged.

Data analyses

For each trial in a Turing test, there was a ground truth (human or machine) and the judge indicated an answer (H or M). We calculated the conditional probabilities: $p(H|H)$ (correct answer), $p(H|M)$ (incorrect answer), $p(M|H)$ (incorrect answer), and $p(M|M)$ (correct answer). These probabilities are reported in the figures (e.g., Fig. 9.3). Entries within a row in each of those figures add up to 1 ($p(H|M)+p(H|H) = 1$ and $p(M|H) + p(M|M) = 1$). We defined the overall imitation detectability as $\frac{1}{2}(p(H|H) + p(M|M))$. The imitation detectability ranges from 0.5 (good imitator, chance level in imitation) to 1.0 (a poor imitator, easy to detect).

Statistical analyses

We used two-tailed t-tests when comparing two distributions and considered results to be statistically significant when $p < 0.05$. Because calculations of p values tend to be inaccurate when the probabilities are extremely low, we reported all p values less than 10^{-15} as $p < 10^{-15}$ (as opposed to reporting, for example, $p = 10^{-40}$). clearly, none of the conclusions depend on this. When considering the imitation detectability over multiple education or gender groups for different AI agents (Fig. Supp.59), we used a two-way ANOVA test[179]. The ANOVA test compares the variation in the detectability within the same condition (gender or education groups) versus the variance across conditions (F-ratio). We report $F(a, b)$ where a and b are the degrees of freedom in the numerator and denominator of the F ratio distribution, and we also report the corresponding p-value.

Code and Data availability

All the source code and raw data are made publicly available in this submission through the following repository: https://klab.tch.harvard.edu/resources/TestingTuringTests.html

Competing interests

The authors declare that they have no competing interests.

Financial Disclosure Statement

This work was supported by NIH grant R01EY026025, by NRF grant NRF-NRFF15-2023-0001, by NSF grant IIS-1901030, by a Google Faculty Research Award, by AI Singapore Programme (AISG Award No: AISG2-RP-2021-025), and by the Center for Brains, Minds and Machines, funded by NSF Science and Technology Centers Award CCF-1231216. MA is supported by a postdoctoral fellowship of the Research Foundation Flanders (FWO 1230521N). The funders had no role in study design, data collection and analysis, the decision to publish, or the preparation of the manuscript.

9.5 Main Figures

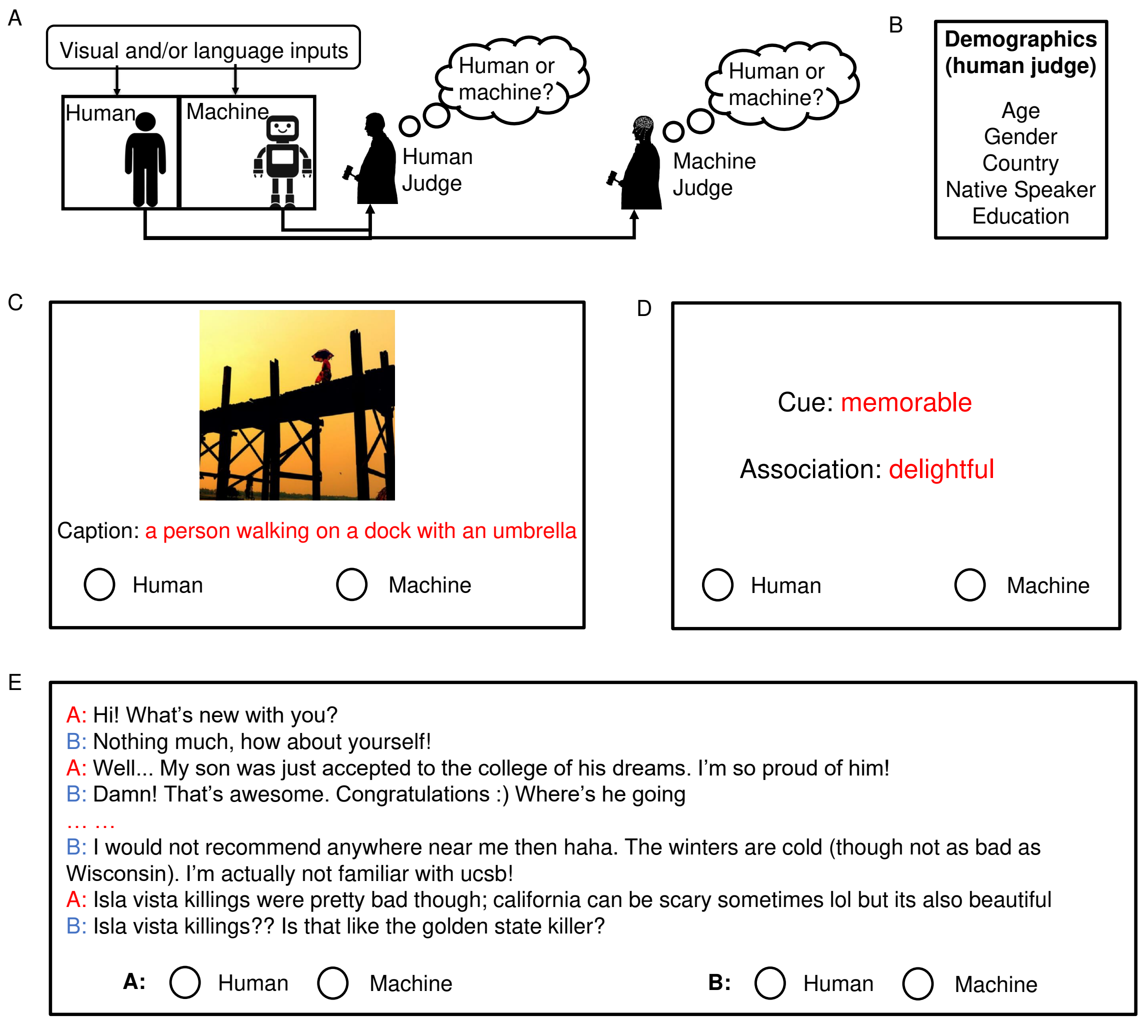


Figure 9.1: Schematic illustration of Turing tests in three language tasks.. A In each task, a human or a machine agent produces an answer in response to visual or language inputs. Given those inputs and answers, a human or machine judge assesses whether the answer came from a human or a machine agent. The machine agent is said to pass the Turing test if the judge is unable to distinguish whether the response came from a human or a machine. B The results of Turing tests with a human judge depend on the characteristics of the human judge. As an initial characterization, we collected basic demographic information. C-E Schematic illustration of the Turing test for three language tasks (see also for three vision tasks). We ask the reader to try the tests before checking the ground truth answers provided at the end of this figure caption. C In the Image captioning task, the agent provides a single-sentence description of an image. The judge is presented with an image and a caption and decides whether the caption was produced by a human or a machine. D In the Word association task, the agent is presented with a word cue and has to produce a single word related to the cue. The judge is presented with a pair of cue and association words and decides whether the association was produced by a human or a machine. E In the Conversation task, two agents (here labeled A and B) alternate to produce a total of 24 exchanges (only 7 exchanges are shown here, for full examples, see Supplementary Section S24). The judge is presented with the conversation and decides whether each conversant was a human or a machine. There were human-human conversations, human-machine conversations, and machine-machine conversations. See Methods for the detailed description of data collection and Turing tests in each task. Ground truth answers for these examples: C machine D machine E A is a machine; B is a human

A

Human: a woman with an umbrella walking on a pier
GIT: a person sitting on a wooden bridge with an umbrella
OFA: a person walking on a dock with an umbrella
BLIP: a person walking across a bridge with an umbrella
ClipCap: a person holding an umbrella in front of a building
Microsoft_API: a person on a swing

B

Human: a girl receiving a certificate from a woman in a group of people at a presentation
GIT: a woman shakes hands with a student
OFA: two women are talking to each other in front of a crowd
BLIP: two women standing next to each other in front of a crowd
ClipCap: a woman standing in front of a group of people
Microsoft_API: a couple of women in a stadium

C

Cue Words	Human	Word2vec	GPT2	GPT3 (embedding, Davinci)	GPT3 (prompt, curie)	GPT3 (prompt, Davinci)
memorable	nice	phenomenal	delightful	meaningful	unforgettable	delightful
oriental	asian	imperial	colonial	eastern	love	colonial
nepal	country	vietnam	pakistan	nebraska	new	pakistan

D

A: Hey! How are you?
B: Hey, I'm doing great. What about you???
A: I'm good. I just finished up work and am looking for something to do.
B: Did you go for work today???
A: Yeah, I went to the office.
B: We had vacation for a week and the amount of assignments are killing me
A: Yeah, I know. Is there anything you're interested in doing?
B: Right now I'm very much occupied with my university's thing that I don't even have time for other things but I'd love to spend some time learning Korean
A: Korean?
B: Yeah, the language
A: Oh, I see. That's really interesting and something that I would like to learn too one day. Would you like me to teach you?
B: Can you? I never knew you know Korean
A: I learned it while living in South Korea for a year. It's not as hard to learn as people think and I would be happy to teach you.
B: I'd love that! Where did you stay in South Korea?
A: I stayed in Seoul.
B: It's so pretty!! I badly want to visit Seoul and attend BTS's concert
A: You should! BTS is amazing. They're my favorite group.
B: Omg!! Whom do you like the most in the band??
A: I really like Jin, Suga and J-Hope.
B: They're awesome. Which our the bands you know??
A: I like EXO, BTS and 2NE1.
B: Have been to any concert??
A: No, I haven't. But I'd love to go to one!
B: Where are you staying right now??

Figure 9.2: Example stimuli from both human and machine agents for each language task. A. Two caption examples from both human and machine agents: GIT [398], OFA [399], BLIP [227], ClipCap [259], and Microsoft's Azure Cognitive Services [mic]. B. Three examples in the Word association task from both human and machine agents (columns): Word2Vec [288], GPT2 [302], GPT3-embedding (davinci) [43], GPT3-prompt (text-curie-001) [43], and GPT3-prompt (text-Davinci-002) [43]. C. Example of a conversation consisting of 24 exchanges between GPT3 Curie [282] ("A", red) and a human ("B", blue). See Supplementary Section S24 for more example conversations, including human-human and machine-machine conversations.

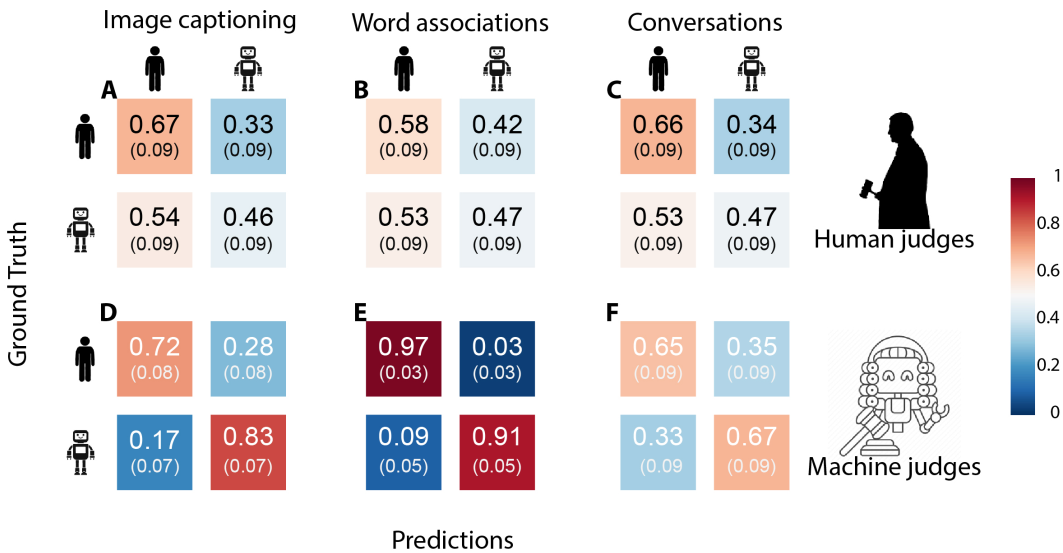


Figure 9.3: Results of the Turing test for each task. Turing test results for human judges (A-C) and machine judges (D-F). Each column shows results for a different task: A, D: Image captioning, B, E: Word association, C, F: Conversation. For each task, the confusion matrices report the percentage of times when the trial was labeled "human" (first column) or "machine" (second column) when the ground truth was human (first row) or machine (second row). The probabilities add up to 1 within each row. Here results are pooled across all machine agents and also across all human judge demographic groups; see Fig. Supp.61 for results from each machine agent and Fig. Supp.59 for results from different human judge demographic groups. The color of each block indicates performance (see color axis on right). Numbers in parentheses indicate standard deviation.

10

Improving generalization by mimicking the human visual diet.

10.1 Main

Biological vision generalizes effortlessly across real-world transformations including changes in lighting, texture, and viewpoint. In contrast, visual recognition models are well known to fail at generalizing across real-world transformations including 2D rotations and shifts[440,56], changes in lighting[243,30,439], object viewpoints[24,234,433,243,319], and color changes[185,330], among others. The dominant narrative guiding recent approaches to bridge the generalization gap is to improve how machines process the data provided to them. Using large-scale internet-scraped datasets as benchmarks, modern approaches have attempted to improve pre-processing through data augmentation[428,167,434], generative modeling[176,396], extracting features better suited for generalization[266,226,229,264,332,400], proposing specialized architectures optimized for domain generalization[326,358,19,195,390,207,33], or using specialized models to detect out-of-distribution (OOD) samples to process them separately[307,327,168,10], among others. However, despite unprecedented progress in these closely related fields of domain adaptation, domain generalization, and out-of-distribution detection, human-like generalization remains an unsolved problem.

In this article, we present an alternative perspective on addressing this generalization gap inspired by how humans and other animals learn—instead of focusing on differences in how data are processed, we focus on the fundamental differences in the data, i.e., the visual diet of humans and machines. It is well documented that data fed during training can have profound impacts on the behaviour of both biological[189,204,18,174,85,408,409] and computer vision[240,319,243,280]. Here, we investigate how generalization behaviour of visual recognition models changes as they are trained with a human-like visual diet, as opposed to internet-scraped datasets which are currently at the heart of most modern computer vision models.

Figure 10.1(a) highlights two major differences between the typical human and machine visual diets. First, children learn from the physical space they occupy—a few 3D scenes and objects while sampling densely from their surroundings under diverse real-world transformations. This includes viewing the same room from different viewpoints, under different lighting over the day, natural occlusions due to the room's layout, and changes in object textures. Second, humans rarely see objects in complete isolation but rather encounter objects in the context of their surroundings. In contrast, popular internet scraped datasets like ImageNet[88] and CIFAR[206] contain sparsely sampled information (~ 1 photograph) from a large number of scenes. Each image is a single snapshot into the information available in the original scene, and data augmentation methods like 2D rotations and crops do not reflect the complex real-world transformations seen by the human visual system. Furthermore, these images often contain objects with minimal scene context and most of the image is occupied by single objects placed in the center. For brevity, we refer to these as differences in transformational diversity and scene context, respectively.

Our main finding is that mimicking and leveraging a human-like visual diet enables better generalization, and models trained with such a diet outperform specialized architectures trained without transformational diversity and scene context. The experiments in this work are enabled by two key technical contributions. First, we introduce a new Human Visual Diet (HVD) dataset, which models both transformational diversity and scene context to better mimick data seen by the human visual system. Figure 10.2 showcases how HVD was created. Second, we propose a new model to leverage the visual diet presented in HVD, as existing recognition models are not designed to exploit a human-like diet. We call this model the Human Diet Network (HDNet). As shown in Fig. 10.1(c), HDNet uses a two-stream architecture where one stream operates on the target object, while the other stream operates on the scene context to jointly reason over target and context to perform visual recognition. HDNet also utilizes transformational diversity by employing a contrastive loss over real-world transformations in the form of lighting, material, and viewpoint changes (Section 10.4). With these findings, we present compelling evidence for the field of computer vision to move towards biologically inspired, diverse data which more accurately model the training data seen by human vision.

10.2 Results

We evaluated the utility of mimicking the human visual diet in improving generalization across real-world transformations in the form of lighting, material and viewpoint changes. First, we evaluated the generalization capabilities of standard architectures trained with data simulating the visual diet commonly used in computer vision—consisting of large-scale, internet-scraped data with low real-world transformational diversity and minimal scene context (Sec. 10.2.1). Then, we confirmed that incorporating and utilizing these two hallmarks of the human visual diet (real-world transformational diversity and scene context) significantly improves generalization (Sec. 10.2.2). We also confirmed that this approach outperforms specialized domain generalization architectures trained with alternatives like traditional data augmentation methods, and specialized GAN-based augmentation methods (Sec. 10.2.3). Finally, as a real litmus test for our findings, we demonstrated that utilizing a human-like visual diet improves generalization from synthetic training data to real-world, natural image data (Sec. 10.2.4).

These experiments are enabled by two datasets containing real-world transformations—the human visual diet (HVD) dataset which is introduced here, and the Semantic-iLab dataset which we created by modifying the multi-view iLab dataset[39]. Each dataset was created to consist of 15 domains with disjoint real-world transformations in the form of lighting, material and viewpoint changes (5 domains per transformation). Sample images from these datasets are shown in Fig. 10.2, and Supplementary Fig. Supp.73, Fig. Supp.74 and Fig. Supp.75. For instance, the 5 material domains in HVD were created by starting with 250 object materials and splitting them into 5 non-overlapping sets of 50 each. For each material domain, these 50 materials were randomly assigned to scene objects, and images were rendered. This procedure ensures that every material shows up in only one particular domain while still ensuring high material diversity across scenes. For every transformation, one domain was held out for testing (e.g., out-of-distribution (OOD) Materials), and never used for training any model. Data diversity is defined as the number of domains the training data is sampled from (ranging from 1 to 4). A similar protocol was used to create disjoint domains and study generalization across lighting and viewpoint changes in the HVD dataset. Experiments with the Semantic-iLab dataset also followed the same protocol as well—15 domains in all with 5 domains per transformation, and models trained with $1, 2, 3$ or 4 domains with 1 held-out domain used for testing which was never used for training (Sec. 10.4).

10.2.1 Models trained without human-like visual diet struggle to generalize across real-world transformations

We simulated the typical visual diet used in many computer vision studies consisting of large-scale, internet-scraped datasets by training visual recognition models with low real-world transformational diversity and minimal scene context. Specifically, for each type of transformation (lighting, materials, and viewpoints), models were trained with only 1 domain and then tested on the held-out test domain with the corresponding transformation. Minimal scene context was simulated by training on cropped images showing only the target object as is common in visual recognition datasets like ImageNet[88]. All models presented below were pre-trained on ImageNet.

We evaluated how well a ResNet18[162] trained with one lighting domain and minimal context could generalize to the held-out, unseen lighting domain. We similarly evaluated how well a ResNet could generalize across different material domains, or viewpoint domains. For HVD, Fig. 10.3(a) shows that ResNet generalized better across lighting changes than material changes (two-sided t-test, $p < 10^{-5}$) or viewpoint changes (two-sided t-test, $p < 10^{-6}$). These results show that there is ample room for improvement for models trained with a visual diet typical in computer vision, especially when tested on unseen material and viewpoint transformations. Similar conclusions can also be drawn for DenseNet[171] and ViT[99] architectures.

We also confirmed that the above findings extend to the natural image Semantic-iLab dataset (Fig. 10.3(b)). ResNet generalized better across lighting changes than material changes (two-sided t-test, $p < 10^{-6}$) or viewpoint changes (two-sided t-test, $p < 10^{-6}$). In the Semantic-iLab dataset, the degree of generalization for material and viewpoints were particularly low. The same conclusions held true for DenseNet and ViT architectures as well.

The generalization deficits were even more striking for the Semantic-iLab dataset compared to the HVD dataset. One potential explanation for this is that unlike the highly photorealistic HVD dataset which was rendered with complete control over lighting and material, Semantic-iLab was created by introducing approximate light and material changes to images in the iLab[39] dataset using white balance modifications and style transfer which result in stark changes across domains (see Sec. 10.4 for details). In sum, state-of-the-art computer vision architectures trained with minimal transformational diversity show only moderate generalization across real-world object transformations, especially for material and viewpoint changes.

10.2.2 A human-like visual diet improves generalization

Below we present evidence in support of our main hypothesis—mimicking and utilizing a human-like visual diet improves this generalization behavior. Specifically, we study two aspects of the human visual diet—Real-World Transformational Diversity (RWTD) and Scene Context, as presented in Fig. 10.1. The impact of these two factors are reported in Sec. 10.2.2 and Sec. 10.2.2 respectively.

Real-word transformational diversity improves generalization

We evaluated the effect of increasing transformational diversity on generalization by ResNet. Real-World Transformational Diversity (RWTD) shown to the model during training is defined by the number of domains from which the training dataset was sampled. For each transformation (lighting, material, or viewpoint), there was one fixed testing domain which stayed constant across all models. The training dataset was always of a fixed size, but the training images were sampled from $1, 2, 3$ or 4 domains (corresponding to 20%, 40%, 60%, and 80% data diversity). This procedure resulted in different levels of transformational diversity shown during training while maintaining a fixed dataset size, thus disentangling the role of transformational diversity and dataset size (Sec. 10.4).

Performance monotonically increased with data diversity in the HVD dataset for all three transformations (Fig. 10.3(c): lighting: 0.85 to 0.94, $p < 10^{-6}$; material: 0.64 to 0.89, $p < 10^{-5}$; viewpoint: 0.63 to 0.73, $p < 10^{-6}$). The improvement in generalization accuracy with data diversity was significantly greater for unseen materials than for unseen lighting ($p < 10^{-4}$) and unseen viewpoints ($p < 10^{-4}$). The smaller increase for unseen lighting changes is expected as the performance was already high for this transformation with low data diversity which left less room for improvement. For unseen viewpoints, these experiments suggest that despite a statistically significant improvement, increasing data diversity is not a sufficient solution for solving generalization, consistent with past work investigating invariance to 3D viewpoint changes[243,318,78,340].

To ensure our findings also extend to natural image data, we also report results with the Semantic-iLab dataset in Fig. 10.3(d). Increasing real-world transformational diversity improved generalization also for the Semantic-iLab dataset: lighting: 0.93 to 1.0, $p < 10^{-3}$; materials: 0.36 to 0.96, $p < 10^{-4}$; viewpoint: 0.46 to 0.75, $p < 10^{-7}$. As with the HVD dataset, improvement in generalization was higher for unseen materials than for unseen lighting ($p < 10^{-3}$) and unseen viewpoints ($p < 10^{-6}$).

The results across both datasets consistently show that generalization across real-world transformations increases with transformational diversity. With sufficient diversity, generalization to unseen lighting and materials reached almost ceiling levels. However, despite improvement, unseen viewpoints remain an open challenge.

Leveraging scene Context substantially improves generalization to OOD real-world transformations.

Next, we evaluated the impact of incorporating another hallmark of the human visual diet—scene context. As shown in the schematic in Fig. 10.1(c), our proposed architecture, HDNet, uses a two-stream architecture that reasons over both the target object and the scene context to classify images. One stream operates on a crop around the target object, while the second stream operates on the full image containing the scene context. HDNet also performs an additional contrastive loss across real-world transformations in the form of lighting, materials, and viewpoints. While contrastive loss has become a staple in modern vision models, this is the first implementation applying contrastive loss over real-world transformations including lighting, materials and 3D viewpoint changes. These features allow HDNet to exploit both aspects of the human visual diet modeled in this study—transformational diversity and scene context (for additional details on the architecture, see Sec. 10.4).

We compared results for multiple state-of-the-art algorithms including a suite of domain generalization methods, a recent context-aware model (CRTNet[38]), and a FasterRCNN model modified to perform visual recognition with our proposed HDNet which utilizes the human-like visual diet. For each real-world transformation, all models were trained with 80% Transformational Diversity, i.e., 4 training domains, and tested on the corresponding held-out test domain. Note that these domain generalization baselines are highly-specialized methodologies with novel engineering combining architectural modifications, optimization strategies, and model selection criteria optimized for the task of domain generalization. In contrast, HDNet is a general purpose architecture meant for visual recognition designed to leverage a human-like visual diet.

Our hypothesis is that the benefits from leveraging the correct diet can outperform the gains from these specialized domain generalization methodologies. Results investigating this hypothesis are presented in Fig. 10.3(e) and Table 10.1. As shown in Fig. 10.3(e), HDNet beat ERM[137] across all three real-world transformations with an accuracies of $0.98, 0.94, 0.83$ compared to ERM's $0.83, 0.75, 0.79$ on unseen lighting, material, and viewpoint changes respectively. Similarly, HDNet also beat IRM[19]) which achieves lower Top-1 accuracies of $0.83, 0.74, 0.79$ on unseen lighting, material, and viewpoint changes respectively. The best performing baseline was found to be another context-aware model—CRTNET[38]) which is also reported in Fig. 10.3(e). Our proposed HDNet model beat this CRTNet baseline with statistical significance on all three real-world transformations. For unseen lighting, HDNet beats CRTNet with an accuracy of 0.98 compared to 0.93 (two-sided t-test, $p < 0.05$). For unseen material changes, HDNet achieves 0.94 which is higher than HDNet's Top-1 accuracy of 0.76 (two-sided t-test, $p < 0.05$). Similarly, for unseen viewpoint changes, HDNet beats MTL with an accuracy of 0.83 compared to 0.79 (two-sided t-test, $p < 0.05$). In summary, both context-aware models (our proposed HDNet and existing CRTNet) outperformed specialized domain generalization approaches on all real-world transformations, and our proposed HDNet also outperformed the closest baseline (CRTNet) with statistical significance.

We also compared HDNet against a suite of additional baselines as reported in Table 10.1. This includes additional domain generalization benchmarks, and a modified FasterRCNN[308] designed to do visual recognition[38]. Findings are consistent across all architectures—HDNet, which utilizes scene context, outperformed all bench-

Real-World Transformation	AND Mask[326]	CAD[33]	CORAL[358]	MTL[34]	Self Reg[195]	VREx[207]	Faster RCNN[308]	HDNet (ours)
Light	0.82	0.80	0.81	0.81	0.75	0.83	0.95	0.98
Materials	0.75	0.75	0.75	0.74	0.74	0.75	0.78	0.94
Viewpoints	0.75	0.77	0.79	0.79	0.76	0.78	0.65	0.83

Table 10.1: Contextual information improves generalization across OOD transformations. We present additional baselines comparing HDNet to several domain generalization methods that do not use contextual cues, building on results presented in Fig. 10.3. All architectures were trained with transformational diversity of 80% and tested on the held-out remaining 20% of the corresponding semantic shift. HDNet beats all baselines by a large margin for all semantic shifts. HDNet also beat a version of FasterRCNN modified to do object recognition, which was provided with contextual cues. The best performing model (HDNet) has been shown in boldface for all real-world transformations.

marks across all real-world transformations including lighting, material, and viewpoint changes.

Given the success of HDNet, we asked whether implementing a two-stream separation of target and context would also improve performance for other architectures. We modified ResNet18[162] and ViT[99] to leverage scene context in the same way as HDNet. For ResNet, a two-stream version was made where each stream is a ResNet backbone. One stream operates on the target, and the other one on the scene context. Output features from each stream were concatenated, and passed through a fully connected layer for classification as shown in Fig. 10.1(c). The two-stream architecture for ViT was analogous. In contrast, the one-stream architecture did not use scene context and operated on the target object alone (see Sec. 10.4 for additional details).

The two-stream architectures consistently led to improved performance with statistical significance (two-sided t test, $p < 0.05$), as shown in Table 10.2. The increase in performance is due to the addition of contextual information and not to the two-stream architecture per se. Indeed, when both streams were trained with the target information, there was a decrease in performance (Table. Supp..3).

To further understand the role of contextual information on visual recognition,

Real-World Transformation	Architecture	1 Stream	2 Stream
Lighting	ResNet	0.85 ± 0.004	$0.95 \pm 0.009^{*}$
	ViT	0.91 ± 0.003	$0.97 \pm 0.007^{*}$
	HDNet (Ours)	-	0.98 ± 0.001
Materials	ResNet	0.64 ± 0.03	$0.83 \pm 0.008^{*}$
	ViT	0.78 ± 0.01	$0.92 \pm 0.003^{*}$
	HDNet (Ours)	-	0.94 ± 0.002
Viewpoint	ResNet	0.63 ± 0.02	$0.72 \pm 0.009^{*}$
	ViT	0.77 ± 0.01	$0.83 \pm 0.001^{*}$
	HDNet (Ours)	-	0.83 ± 0.006

Table 10.2: Adding scene context improves performance independent of architecture. Following the design of HDNet shown in Fig. 10.1(c), we modified standard architectures to have two streams—one operating on the target, and the other one on the contextual information. Representations for both streams are then concatenated and passed through a classification layer as shown in Fig. 10.1(c). We train the standard one-stream and these modified two-stream architectures on HVD, and report the average Top-1 accuracy for all models . We also report error bars, which measures the variance in accuracies over categories. Both the ResNet and the ViT architectures lead to a large improvement in generalization for all semantic shifts when modified to leverage scene context. To ensure we study impact of context independent of data diversity, all models were trained on 4 domains, i.e., 80% transformational diversity and tested on the held out domain. Best performing model (HDNet) has been shown in boldface for all real-world transformations. A $*$ refers to statistically significant improvement in performance when using a two-stream architecture as compared to a one-stream architecture (two-sided t-test, $p < 0.05$).

we evaluated the impact of blurring the scene context while keeping the target intact[437]. For each real-world transformation, we trained and tested models with increasing levels of Gaussian blurring applied to the scene context. Blurring was applied to the images in the form of a Gaussian kernel filter, with the kernel standard deviation (σ) set to 0, 25, or 125. The cropped image of the target object was passed to the second stream of the network without blurring (Supplementary Sec. S27). There was a drop in performance with context blurring for all three real-world transformations(Table 10.3).

Semantic Shift	Full Context ($\sigma =$ 0)	Less Context ($\sigma = 25$)	Least Context ($\sigma = 125$)
Lighting	0.98 ± 0.001	0.96 ± 0.001	0.94 ± 0.001
Material	0.94 ± 0.002	0.88 ± 0.01	0.83 ± 0.006
Viewpoint	0.83 ± 0.006	0.77 ± 0.01	0.76 ± 0.01

Table 10.3: Blurring scene context worsens generalization performance. We trained and tested HDNet with the scene context in HVD images blurred using a Gaussian blur. Here, σ is the standard deviation for the gaussian kernel applied to the image as a filter. Thus, blurring increases with σ. We applied three values for σ—0,25, and 125. For brevity, numbers less than 0.001 are reported as 0.001.

10.2.3 Alternatives to mimicking the human-like visual diet are not equally advantageous

We asked whether similar performance gains to the ones achieved by using a human-like visual diet could be obtained by alternate strategies that can be implemented on existing internet-scraped datasets. Achieving comparable performance using internet-scraped datasets would bypass the laborious efforts in collecting data and adequate controls. We investigated three such approaches which have been popularly used in the literature—data augmentation, using generative models to modify existing images, and increasing diversity on more easily controllable real-world transformations.

Traditional Data augmentation does not provide similar performance to true diversity with real-world transformations.

Here we investigated how real-world transformational diversity (RWTD) compares to traditional data augmentation strategies including 2D rotations, scaling, and changes in contrast. Models trained with a visual diet consisting of 80% RWTD were reported in Fig.3(e). We compared these with models trained with a visual diet consisting of 20% RWTD + traditional augmentation. As before, all models were tested on unseen lighting, material, and viewpoint changes.

Note that the number of training images was kept constant across all training scenarios to evaluate the quality of the training images rather than their quantity. Training set size equalization was achieved by sampling fewer images per domain in the 80% RTWD training set. For instance, for HVD experiments with unseen viewpoints we sampled $15,000$ training images per viewpoint domain to construct the training set with 20% RWTD + Data Augmentations. In comparison, we sampled only $3,750$ per viewpoint domain to construct the 80% RWTD training set. Thus, the initial sizes of the 80%RWTD and the 20%RWTD+Data Augmentation training sets was identical. However, due to data augmentations being stochastic the total number of unique images shown to models trained with data augmentations was much larger. Assuming a unique image was created by data augmentation in every epoch, over 50 epochs the dataset size would be 50 times larger with data augmentations. Additional details on dataset construction can be found in the methods in Sec. 10.4.

HDNet trained on HVD with 80% RWTD outperformed the same architecture trained with 20% RWTD+traditional data augmentation for lighting changes (two-sided t test, $p < 10^{-4}$), material changes (two-sided t test, $p < 10^{-5}$), and viewpoint changes (two-sided t test, $p < 10^{-6}$) (Fig. 10.4(a)). Similar conclusions were reached for the Semantic-iLab dataset. A ResNet model trained with 80% RWTD outperformed the same architecture trained with 20% RWTD+traditional data augmentation for lighting changes (two-sided t test, $p < 10^{-4}$), material changes (two-sided t test, $p < 10^{-7}$), and viewpoint changes (two-sided t test, $p < 10^{-5}$) (Fig. 10.4(b)).

One explanation for this finding could be that traditional data augmentation largely involves 2D affine operations (crops, rotations) or image-processing based methods (contrast, solarize) which are not necessarily representative of real-world transformations. In summary, the positive impact of a visual diet consisting of diverse lighting, material, and viewpoint changes (real-world transformational diversity) cannot be replicated by using traditional data augmentation applied to the dataset after data collection—diversity must be ensured at the data collection level.

Real-world transformations outperform modifying a low diversity sample with generative models.

Several existing works rely on increasing data diversity using AdaIn-based methods [172,447]. These style transfer methods change the colors in the image while retaining object boundaries, but do not modify materials explicitly as done in our HVD dataset. We evaluated how well models perform if diversity is increased using style transfer as opposed to material diversity. We started with one material domain, and created four additional domains using style transfer. Sample images of style transfer domains are shown in Fig. 10.4(c). Corresponding images from the HVD dataset with real-world transformation in materials can be seen in Fig. 10.2(a). The total number of domains (and images) created using style transfer was kept the same as the material domains in HVD. The only difference in the training data was that instead of four additional material domains, we have four additional style transfer domains. We compare models trained with these two different visual diets—one consisting of four material domains, and the other consisting of four style transfer domains. All models are then tested on the same held out OOD Materials domain.

Style transfer domains did not enable models to generalize to new materials as well as the material shift domains presented in HVD (Fig. 10.4(d)). These experiments support the notion that in order to build visual recognition models that can generalize to unseen materials, it is important to explicitly increase diversity using additional materials at the time of training data collection. The impact of diverse materials cannot be replicated by using style transfer to augment the dataset after data collection.

Importance of diversity on all real-world transformations

Some real-world transformations are easier to capture than others. For instance, capturing light changes during data collection might be significantly easier than collecting all possible room layouts, or object viewpoints. Thus, it would be beneficial if training with one transformation (e.g., light changes) can improve performance on a different transformation (e.g., viewpoint changes). We refer to such a regime as assymetric diversity—as models are trained with one kind of diversity, and tested on a different kind of diversity (Fig. 10.4(e),(f)).

In all cases, the best generalization performance was obtained when training and testing with the same real-world transformation for both HVD (Fig. 10.4(e)) and Semantic-iLab datasets (Fig. 10.4(f)). In most cases, there was a drop in performance of 10% or more when training in one transformation and testing in with a different (assymetric) transformation. These experiments imply that to build models that generalize well, it is important to collect training data with multiple real-world transformations.

10.2.4 Models trained with a human-like visual diet can generalize to real-world images

As a litmus test for our findings, we investigated if models trained with a human-like visual diet can generalize well to the natural images from the ScanNet dataset despite only being trained with synthetic images from HVD. Fig. 10.5(a) shows paired testing and training data across these two datasets. The test set is composed of natural images from the ScanNet dataset, while the training set consists of only synthetic images from HVD. HVD images were created by 3D reconstructing ScanNet scenes, and then rendering them under diverse lighting, material and viewpoint change as shown in Fig. 10.5(a).

We made three adaptations for these experiments. Firstly, as both ScanNet and ImageNet contain natural images and overlapping categories, we trained models from scratch to ensure pre-training does not interfere with our results. Thus, these models never saw any real-world images, not even ImageNet as they were not pretrained on those datasets. Secondly, we trained and tested models on overlapping classes between HVD and ShapeNet. Finally, we used the LabelMe[394] software to manually annotate a test set from ScanNet and training set for the HVD dataset using the same procedure to make sure biases from the annotation procedure do not impact experiments.

Thus, all models were trained purely on synthetic data from HVD and tested on only real-world natural image data from ScanNet as shown in Fig. 10.5(a). While all the models that we tested performed above chance levels when tested in the real-world ScanNet images, there were large performance differences among models (Fig. 10.5b).

Results on generalization to the real world are presented in Fig. 10.5. As can be seen, the best performing model (IRM) trained without the human visual diet obtained classification performance of 0.51, while HDNet trained with transformational diversity and scene context performs substantially better at 0.69. Despite being trained only on synthetic data and no pre-training on natural images, HDNet generalizes well to the real world by leveraging attributes of the human visual diet. Our approach with the HVD dataset and HDNet, which mimics and utilizes two hallmarks of human visual diet (real-world transformational diversity and scene context) beats all benchmarks with statistical significance ($p < 0.05$). This evidence confirms that our findings on Semantic-iLab and the HVD datasets also extend to real-world natural images from the ScanNet dataset.

10.3 Discussion

Alleviating the generalization gap between biological and computer vision remains an open and elusive problem. The past few years have seen unprecedented progress in improving generalization in machine vision driven by how algorithms process images through cutting-edge architectures and data augmentation techniques. Here, we present an alternative direction bringing together ideas from vision sciences, computer graphics, and computer vision—mimicking the human visual diet.

There is a long history of studies showing that changing the visual diet alters the visual cortex[204,189]. Such developmental neuroscience studies include evaluating the consequences of rearing animals in visual environments deprived of binocular vision[174], environmental directionality[85], temporal contiguity[408], surface features[409], or faces[18], among others. Some studies have also tried to quantitatively define the human visual diet[69]. In a similar vein, some computational works have also studied how the training data distribution (visual diet) impacts visual search assymetries[142], and object recognition[240,243]. Here, we combined ideas from biological and computer vision and evaluated the impact of a human-like visual diet on the generalization behavior of visual recognition models.

Through controlled analyses, we show that two hallmarks of human visual diet (transformational diversity and scene context) provide significant improvements in generalization across real-world transformations compared to existing popular approaches. These experiments are enabled by two computational contributions introduced here—a dataset which mimics the human visual diet, and an architecture which can leverage this visual diet.

As a first step in mimicking the human visual diet, we focus on two important aspects of this diet—scene context and transformational diversity. There is a long history of work studying the role scene context in human vision[348,135,134,376,165,253,437] and modeling contextual cues computationally[376,365,437,38]. Additinoally, recent work in machine learning has stressed the importance of data diversity for generalization[243,240]. These two aspects of the human visual diet are certainly not exhaustive and multiple other features warrant future investigation, including depth information, occlusions, and dynamic cues.

The concept of generalization is ill-defined and depends on the training and test distributions. Humans also struggle with generalization when image statistics change (e.g., consider humans trying to read bar codes in the supermarket, learning to diagnose clinical images, or deciphering the shapes of new galaxies). A laudable goal in computer vision is often to align humans and machines. Such alignment takes the form of avoiding adversarial attacks that are imperceptible for humans but alter machine-produced labels, enhancing computer vision function in the real world, or even recognition challenges that would benefit from human-machine collaboration. Ensuring that the visual diets are similar can help accelerate such alignment.

Advances in algorithms usually accompany progress in building better datasets. The introduction of a two-stream architecture constitutes a reasonable first-order approach to capitalize on transformational diversity and contextual cues. At the same time, we expect that more powerful datasets will incentivize the development of better algorithms that could also be in turn inspired and constrained by biological vision.

10.4 Methods

10.4.1 Datasets mimicking attributes of the human visual diet

Several benchmarks have been proposed to study generalization in computer vision. On one hand, we have datasets like ImageNet-P[166] and ImageNet-C[166], which introduce controlled synthetic noise that can be quantified, but the introduced noise does not reflect real-world transformations. On the other hand, there are domain generalization datasets like PACS[221], VLCS[373], Office-Home[391], DomainNet[287], and Terra Incognita[30], among others. While these datasets do include transformations that exist in the real world, the distribution shift between domains in these benchmarks cannot be quantified or disentangled into scene parameters. For instance in PACS[221], it is unclear how the Photo → Cartoon domain shift differs from a Photo → Art shift. Furthermore, all these datasets show objects with minimal context, as they are composed of crops centered around objects. The proposed HVD dataset was designed to address these issues—it shows objects in context, with controlled, disentangled real-world transformations. Below we introduce HVD, and also modify an existing natural image dataset to partly mimick the human-like visual diet.

Human Visual Diet (HVD) Dataset

We reconstructed 1,288 real-world scenes from the ScanNet dataset with the exact same 3D objects, scene layouts, class distributions and camera parameters using the OpenRooms framework[231,230]. With these scenes, we created 15 photo-realistic domains with 3 types of real-world transformations including—lighting, material, and viewpoint changes. Some sample images are provided in Fig. 10.2(a). We rendered $19,800$ images for each domain, which results in $70,000$ object instances from 13 indoor object categories. Across all domains, this amounts to roughly $300,000$ images and 1 million object instances. Below, we explain how different semantic shifts were created.

Material shift domains: We used 250 high quality, procedural materials from Adobe Substances including different types of wood, fabrics, floor and wall tiles, and metals, among others. These were split into sets of 50 materials each to create 5 different material domains (supplementary Fig. Supp.74). For each domain, its 50 materials were randomly assigned to scene objects. One domain was held out for testing (OOD

Materials), and never used for training any model.

Light shift domains: Outdoor lighting was controlled using 250 High Dynamic Range (HDR) environment maps from the Laval Outdoor HDR Dataset[169] and OpenRooms, which were split into 5 sets of 50 each (one set per domain). Disjoint sets of indoor lighting were created by splitting the HSV color space into chunks of disjoint hue values. Each domain sampled indoor light color and intensity from one chunk (supplementary Fig. Supp.73). One domain was held out for testing (OOD Light), and never used for training.

Viewpoint shift domains: Controlling object viewpoints presents a challenge as indoor objects are seen across a variety of azimuth angles (i.e., side vs front) across 3D scenes. Thus, to create disjoint viewpoint domains (supplementary Fig. Supp.75) we chose to control the zenith angle by changing the height at which the camera is focusing. Again, of the 5 domains, one was held out for testing (OOD Viewpoints).

Natural image test set from ScanNet

To create the real-world test dataset, we sampled images from the ScanNet dataset[82]. ScanNet contains scenes captured using a moving camera from which frames can be extracted. We manually compared ScanNet scenes with our reconstructions of these scenes using OpenRooms, and selected 3D scenes for which OpenRooms was successfully able to recreate all three—scene layout, camera parameters, and geometry. As the video is continuous, nearby frames are highly similar. Therefore, we subsampled one frame for every 100 frames in the video. These frames were then annotated using LabelMe. To ensure a high quality test set, results are reported in the main paper on a subset of 350 images which are not blurry and do not have significant clutter. However, here we report numbers on a larger subset of 700 test images where we allow clutter and minor blurring so as to achieve a bigger test set. These results are reported in supplemtary Sec. S27. The same procedure was also followed to hand-annotate $8,000$ training object instances from the HVD dataset to ensure there is no spurious impact of the annotation procedure on the performance of models when tested on ScanNet.

Semantic-iLab dataset

To ensure our findings extend to natural images, we modified the iLab[39] dataset to create a natural image dataset with controlled variations in Lighting, Material, and Viewpoint, as shown in Fig. 10.2 (b). We call this the Semantic-iLab dataset. The iLab dataset contains objects from 15 categories placed on a turntable and photographed from varied viewpoints. A foreground detector was used to extract a mask for the object in each image. Material variations were implemented using AdaIN[173] based style transfer on these images and overlaying the style transferred image onto the object mask in the original image. Lighting changes were implemented by modifying the white balance. Note that unlike HVD, this dataset does not contain scene context. Additional details can be found in supplementary Sec. S26.

10.4.2 Human Diet Network (HDNet)

A schematic of the proposed HDNet is shown in Fig 10.6. We start with CRTNet as the backbone[38] and introduce critical modification of contrastive learning described below to enable generalization across semantic shifts.

Feature Extraction in Context-aware Recognition using a Cross-attention Transformer

The context-aware recognition model in[38] achieved superior performance in in-context object recognition when the training and test data are from the same domain. Here, we used the same backbone and briefly introduce the network architecture below (see[38] for implementation details).

Given the training dataset $D = \{x_i, y_i\}_{i=1}^n$, HDNet is presented with an image x_i with multiple objects and the bounding box for a single target object location. $I_{i,t}$ is obtained by cropping the input image x_i to the bounding box whereas $I_{i,c}$ covers the entire contextual area of the image x_i. y_i is the ground truth class label for $I_{i,t}$. In this subsection, we focus on extracting context and target features in the embedding space and omit the index i for simplicity. Inspired by the eccentricity dependence of human vision, HDNet has one stream that processes only the target object (I_t, 224×224), and a second stream devoted to the periphery (I_c, 224×224) which processes the contextual area.

The context stream is a transformer decoder, taking I_c as the query input and I_t as the key and value inputs. The network integrates object and context information via hierarchical reasoning through a stack of cross-attention layers in the transformer, extracts context-integrated feature maps $F_{t,c}$ and predicts class label probabilities $y_{t,c}$ within C classes.

A model that always relies on context can make mistakes under distribution shifts. Thus, to increase robustness, HDNet makes a second prediction y_t, using only the target object information alone. A 2D CNN is used to extract feature maps F_t from I_t, and estimates the confidence p of this prediction y_t. Finally, HDNet computes a confidence-weighted average of y_t and $y_{t,c}$ to get the final prediction y_p. If the model makes a confident prediction with the object only, it overrules the context reasoning stage.

Supervised Contrastive Learning for Domain Generalization

Contrastive learning has benefited many applications in computer vision tasks (e.g.,[297,63,345,438,195]). However, all these approaches require sampling positive and negative pairs from real-world data. To curate positive and negative pairs, image and video augmentations operate in 2D image planes or spatial-temporal domains in videos. Here we introduce a contrastive learning method on 3D transformations.

Our contastive learning framework builds on top of the supervised contrastive learning loss[194]. Given the training dataset $D = \{x_i, y_i\}_{i=1}^{n}$, we randomly sample N data and label pairs $\{x_k, y_k\}_{k=1}^{N}$. The corresponding batch pairs used for constrative learning consist of $2N$ pairs $\{\tilde{x}_l, \tilde{y}_l\}_{l=1}^{2N}$, where $\tilde{x}_{2k}$ and $\tilde{x}_{2k-1}$ are two views created with random semantic domain shifts of $x_k (k = 1, ..., N)$ and $\tilde{y}_{2k} = \tilde{y}_{2k-1} = \tilde{y}_k$. Domain shifts are randomly selected from a set of HVD domains specified during training. For example, if x_k is from a material domain, $\tilde{x}_{2k}$ and $\tilde{x}_{2k-1}$ could be images from the same 3D scene but with different materials. For brevity, we refer to a set of N samples as a batch and the set of $2N$ domain-shifted samples as their multiviewed batch.

Within a multiviewed batch, let $m \in M := \{1, ..., 2N\}$ be the index of an arbitrary domain shifted sample. Let $j(m)$ be the index of the other domain shifted samples originating from the same source samples belonging to the same object category, also known as the positive. Then $A(m) := M \backslash \{m\}$ refers to the rest of indices in M except for m itself. Hence, we can also define $P(m) := \{p \in A(m) : \tilde{y}_p = \tilde{y}_m\}$ as the collection of indices of all positives in the multiviewed batch distinct from m. $|P(m)|$ is the cardinality. The supervised contrastive learning loss is:

$$L_{contrast} = \sum_{m \in M} L_m = \sum_{m \in M} \frac{-1}{|P(m)|} \sum_{p \in P(m)} \log \frac{\exp(z_m \cdot z_p / \tau)}{\sum_{a \in A(m)} \exp(z_m \cdot z_a / \tau)} \quad (10.1)$$

Here, z_m refers to the context-dependent object features $F_{m,t,c}$ on $\tilde{x}_m$ after L2 normalization. The design motivation is to encourage HDNet to attract the objects and their associated context from the same category and repel the objects and irrelevant context from different categories.

As previous works have demonstrated the essential role of context in object recognition[38,437], contrastive learning on the context-modulated object representations enforces HDNet to learn generic category-specific semantic representations across various domains. τ is a scalar temperature value which we empirically set to 0.1.

Overall, HDNet is jointly trained end-to-end with two types of loss functions: first, given any input x_m consisting of image pairs $I_{m,c}$ and $I_{m,t}$, HDNet learns to classify the target object using the cross-entropy loss with the ground truth label y_m; and second, contrastive learning is performed with features $F_{m,t,c}$ extracted from the context streams:

$$L = \alpha L_{contrast,c,t} + L_{classi,t} + L_{classi,p} + L_{classi,c,t} \tag{10.2}$$

Hyperparameter α is set to 0.5 to balance the supervision from constrastive learning and the classification loss. Supplementary Table Supp..2 shows that the contrastive loss introduced in HDNet results in improved performance across all real-world transformations.

10.4.3 Experimental Details

Baseline Architectures

HDNet was compared against several baselines presented below. All models were trained on NVIDIA Tesla V100 16G GPUs. Optimal hyper-parameters for benchmarks were identified using random search, and all hyper-parameters are available in the supplement in Sec. S29.

2D feed-forward object recognition networks: Previous works have tested popular object recognition models in generalization tests[119,41]. We include the same popular architectures ranging from 2D-ConvNets to transformers: DenseNet[175], ResNet[162], and ViT[99]. These models do not use context, and take the target object patch I_t as input.

Domain generalization methods: We also compare HDNet to an array of state-of-the-art domain generalization methods (Table 10.1). These methods also use only the target object, and do not use contextual information.

Context-aware recognition models: To compare against models which use scene context, we include CRTNet[38] and Faster R-CNN[308]. CRTNet fuses object and contextual information with a cross-attention transformer to reason about the class label of the target object. We also compare HDNet with a Faster R-CNN[308] model modified to perform recognition by replacing the region proposal network with the ground truth location of the target object.

Evaluation of computational models

Performance for all models is evaluated as the Top-1 classification accuracy. Error bars reported on all figures refer to the variance of per-class accuracies of different models. For statistical testing, p-values were calculated using a two-sample paired t-test on the per-category accuracies for different models. The t-test checks for the null hypothesis that these two independent samples have identical average (expected) values. For ScanNet, a t-test is not optimal due to the smaller number of samples, and thus a Wilcoxon rank-sum test was employed for hypothesis testing as suggested in past works[86,293]. All statistical testing was conducting using the python package scipy, and the threshold for statistical significance was set at 0.05.

Data and Code Availability Statement

Source code and data are available at https://github.com/Spandan-Madan/human_visual_diet.

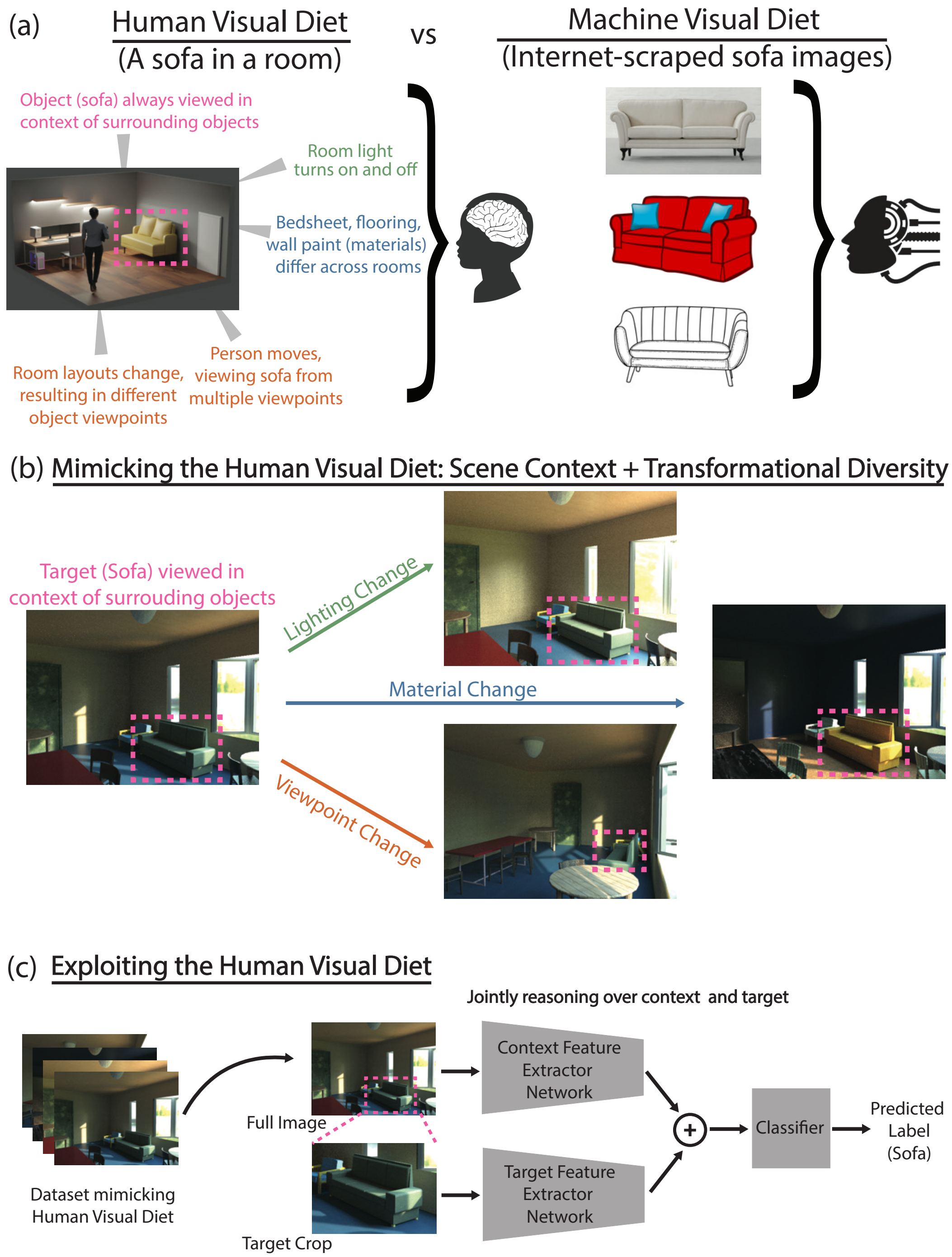


Figure 10.1: Mimicking and exploiting the human visual diet. (a) Comparing human and machine visual diets: The desk in the 3D room is viewed under a variety of real-world transformations which are essential components of the human visual diet. Furthermore, objects are always seen in context of their surroundings. In contrast, sample images of internet-scraped desks which constitute the machine visual diet do not contain these real-world transformations, or scene context. (b) Mimicking the human visual diet by introducing disentangled lighting, material, and viewpoint changes to a 3D scene where objects are shown in context. (c) Exploiting the human visual diet by using a two-stream architecture which reasons over both target object and its surrounding scene context.

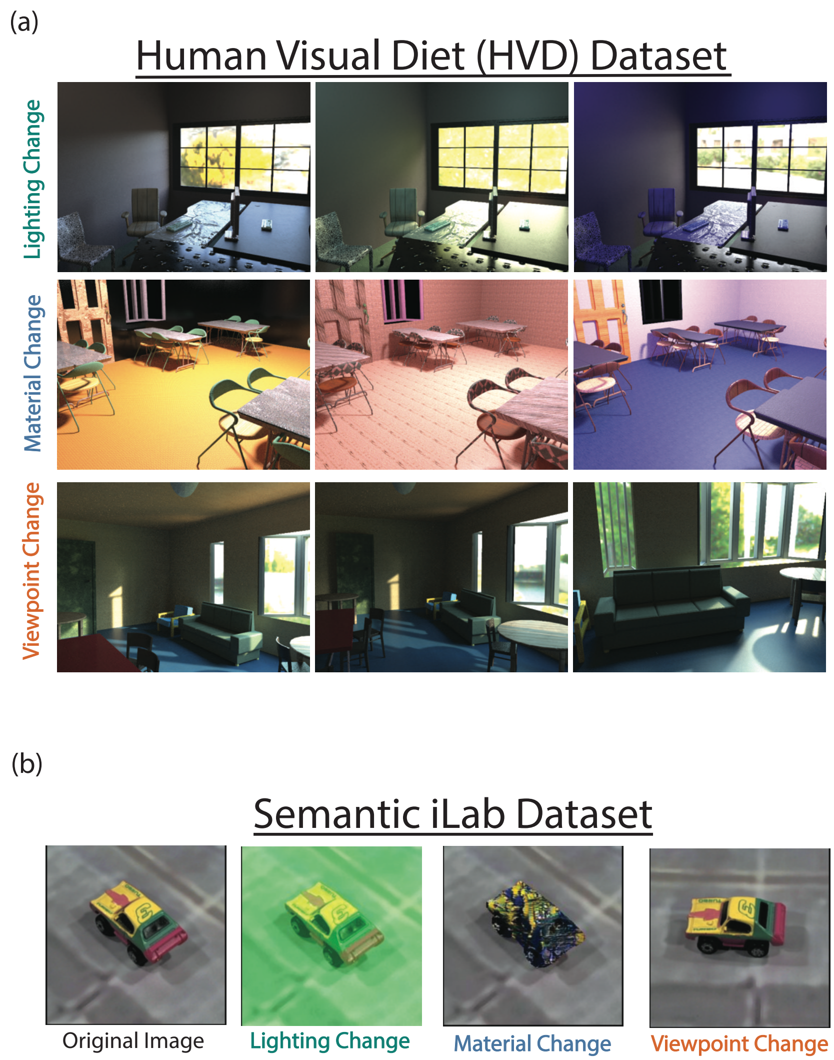


Figure 10.2: Datasets with real-world transformations. (a) Sample images from the Human Visual Diet dataset: We created 15 photo-realistic domains with three, disentangled real-world transformations—lighting, material, and viewpoint changes. Each 3D scene was created by reconstructing an existing ScanNet[82] scene using the OpenRooms framework[231], followed by introduction of controlled changes in scene parameters before rendering these images. (b) Sample images from the Semantic-iLab dataset: We modify the existing iLab dataset[39] by augmenting images with changes in lighting and material. These changes are achieved by modifying the white balance and using AdaIN[172] based style transfer, respectively.

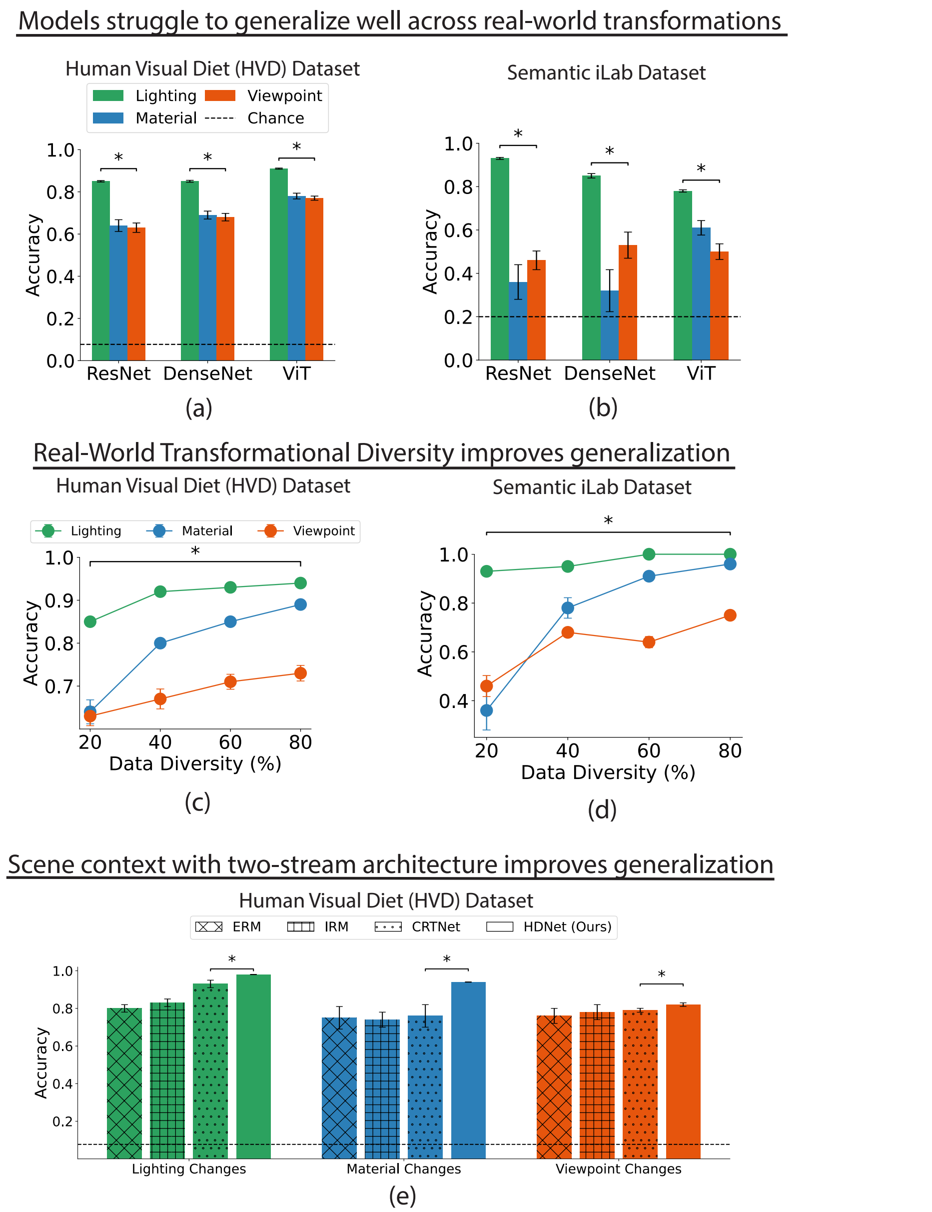


Figure 10.3: Human Visual Diet leads to significantly improved generalization across real-world transformations.((a) Existing models struggle to generalize across real-world transformations, especially material and viewpoint changes. This result holds for both HVD and Semantic-iLab datasets. (b) Increasing real-world transformational diversity leads to a significant increase in generalization performance for all transformations (lighting, viewpoint and materials) for both datasets. (c) HDNet leverages scene context resulting in substantially better generalization than seminal domain generalization architectures like ERM[34], IRM[19]. HDNet is designed to incorporate scene context into visual recognition, by using a two-stream architecture to reason over the target object and scene context simultaneously. In contrast, above mentioned state-of-the-art approaches for domain generalization are single stream architectures that do not leverage scene context. HDNet also beats a suite of additional domain generalization baselines presented in Table 10.1. The closest performing baseline is another context-aware [195] model (CRTNet[38]), and our proposed model beats theses baselines for all three transformations with statistical significance. For all plots, statistical significance is evaluated using a two-sample t-test, and an * indicates a p-value lower than the threshold of 0.05. See methods for additional details.

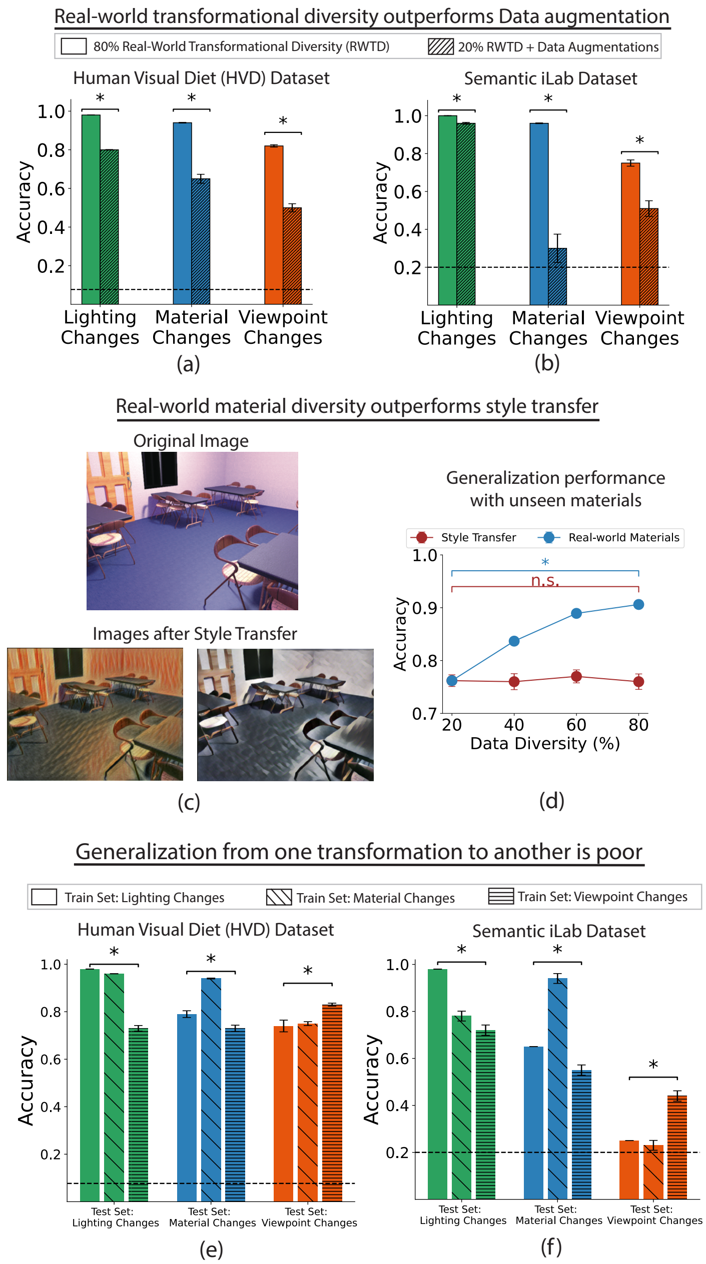


Figure 10.4: Data post-processing does not match gains from collecting data mimicking the human visual diet. (a) Models trained 80% real-world transformational diversity (RWTD) significantly outperform modesl trained with 20% along with traditional data augmentation. This is true for all transformations (lighting, material, and viewpoint) across both HVD and Semantic-iLab datasets. Number of images is held constant in these experiments. (b) Sample images from style transfer domains created using AdaIn[172], alongside accuracies of models trained with these domains. Models trained on style transfer domains generalize significantly worse than those trained with material diversity. (c) Generalization from one transformation to another (asymmetric diversity) does not help as much as training with the correct transformation—best generalization to unseen materials is achieved when material diversity is added to the training data. For generalizing to unseen light and viewpoint changes as well, training with the corresponding real-world diversity helps the most.

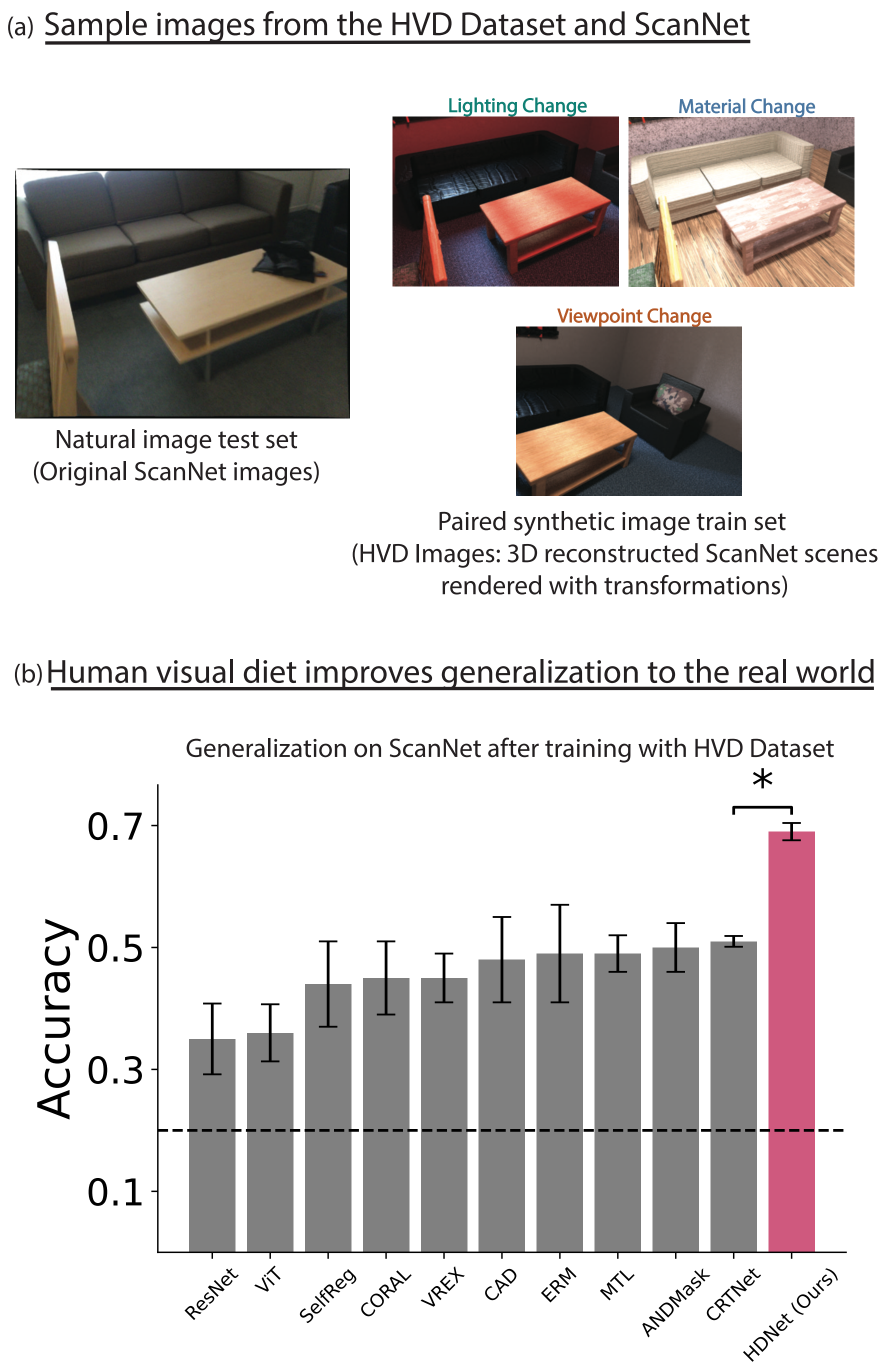
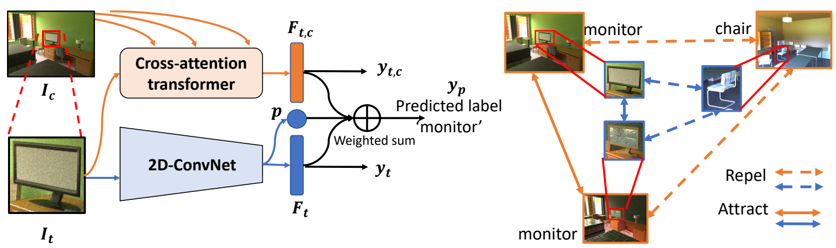


Figure 10.5: Utility of the human visual diet in generalizing from synthetic to real-world, natural image data. (a) Sample synthetic images from the HVD dataset used for training the model, and the corresponding real-world natural image from ScanNet used for testing. (b) Human Visual Diet enables substantially better generalization from synthetic to natural image data. Our approach, which mimics and effectively utilizes transformational diversity and scene context leads to better performance than all other baselines.



(a) Context-aware Feature Extraction (b) Contrastive Learning

Figure 10.6: Architecture overview for Human Diet Network(HDNet). (a) Modular steps carried out by HDNet in context-aware object recognition. HDNet consists of 3 modules: feature extraction, integration of context and target information, and confidence-modulated classification. HDNet takes the cropped target object I_t and the entire context image I_c as inputs and extracts their respective features. These feature maps are tokenized and information from the two streams is integrated over multiple cross-attention layers. HDNet also estimates a confidence score p for recognition using the target object features alone, which is used to modulate the contributions of F_t and $F_{t,c}$ in the final weighted prediction y_p. (b) To help HDNet learn generic representations across domains, we introduce contrastive learning on the context-modulated object representations $F_{t,c}$ in the embedding space. Target and context representations for objects of the same category are enforced to attract each other, while those from different categories are enforced to repel. Pairs for contrastive learning are generated using various material, lighting or viewpoint shifts (Sec. 10.4.1).

Part IV

OOD Generalization capabilities of the brain

11

OOD generalization capabilities of the models of the cortex

11.1 Introduction

Deep Neural Networks (DNNs) for vision have internal representations that share similarities with neural representations in the visual cortex, including the primate ventral visual stream[26,292]. This representational similarity allows for models that use image representations extracted from a pre-trained DNN (e.g., ResNet[162]) to predict neuronal firing rates[423] (Fig. 11.1(a)). However, DNNs are known to struggle

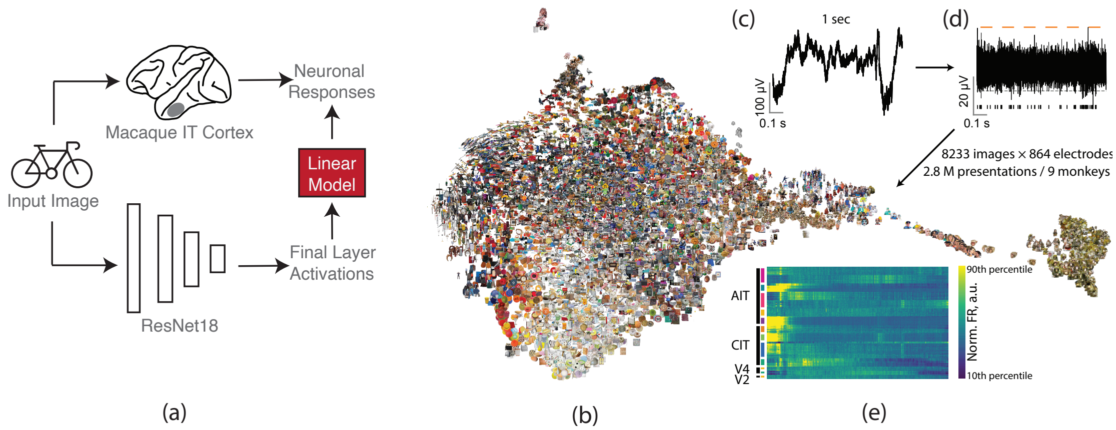


Figure 11.1: Modeling the visual cortex with MacaqueITBench. (a) DNN-Based models of the visual cortex employ a linear model to map image features extracted from pre-trained DNNs (e.g., ResNet18) to neuronal responses collected from the macaque IT cortex. (b) A UMAP[254] visualization of the representation by the neural pseudo-population. Nearby images have more similar population responses. (c) An example one-second segment of the raw wideband signals recorded on an electrode. (d), The wideband signals were highpass filtered, and threshold-crossing events below a voltage value (horizontal dashed line) were counted as multiunit spikes (lower vertical ticks). The top horizontal bars indicate image presentation periods. (e) The heatmap shows the neural response matrix. Each row indicates the responses from an electrode, pooled across sessions. The columns correspond to images, sorted by the reverse UMAP horizontal order. The vertical bars to the left of the heatmap denote the recorded areas (black lines) and monkeys (colored lines).

with generalization under distribution shifts such as Out-of-Distribution (OOD) viewpoints[244,240,78], materials and lighting [242,319], and noise[166,80]. This difficulty in generalization may also affect models of the visual cortex that rely on a DNN to extract image representations.

We posit that, even within an image set where DNN-based models predict neural responses well under random splits across images, specific train-test splits with distribution shifts will impair model performance, proportional to the size of distribution shift. To test this hypothesis, we collected MacaqueITBench, a large-scale dataset of responses to natural images by neurons in the macaque ventral visual pathway. The dataset represents neurons in V2, V4, Central IT (CIT), and Anterior IT (AIT) (primarily CIT and AIT) and responses to over 300,000 images (8,233 unique images presented to seven monkeys over 109 sessions), as illustrated in Fig. 11.1(b).

Using MacaqueITBench, we investigated the impact of distribution shifts on the neural predictivity of DNN-based models of the visual cortex. We constructed various OOD distribution shifts, some of which are schematized in Fig. 11.2. Foreshadowing, our main finding is that distribution shifts in even low-level image attributes break DNN-based models of the visual cortex. This observation highlights a problem in modern models of the visual cortex—good predictions are limited to images that belong to the training data distribution.

To explain the OOD model-performance drop, we built on theoretical work positing that generalization performance is closely correlated with the amount of distribution shift[49,285]. While theoretical studies have examined simplistic, simulated data, we show that a suitable metric of the size of distribution shifts can account for the OOD generalization performance of neural-encoding models.

In summary, our main contributions are threefold:

- We present MacaqueITBench, a large-scale dataset of neural population responses to over $300,000$ images spanning multiple areas of the primate ventral visual pathway.
- We show that modern models of the visual cortex do not generalize well—simple distribution shifts can reduce neural predictivity to as low as 20% of in-distribution performance.
- We show that a simple metric of distribution shift sizes can predict OOD neural predictivity.

11.2 Related Work

11.2.1 DNN-based models of the Visual Cortex

A touchstone for visual neuroscience is the ability to predict neuronal responses to arbitrary images. On this test, DNN-based models have emerged as state-of-the-art models, best explaining neural responese across species—e.g., mouse and macaque—and visual cortical areas—from the primary visual cortex (V1) to the high-level inferior temporal cortex (IT). DNN encoding models of the visual cotext are reviewed more generally in[205,422]. Most pertinently here, these DNN-based models have been evaluated using random cross-validation (e.g.,[322]), which tests IID generalization. OOD generalization in such models have been sparsely examined; we are only aware of one study[309] comparing model fit to neural responses on two image types. Here, we systematically vary the type and degree of OOD splits to investigate how different splits lead to different generalization gaps.

11.2.2 Out-of-distribution generalization capabilities of DNNs

DNNs for object recognition have been documented to fail at generalizing across a wide range of distribution shifts. Such shifts include 2D rotations and shifts[440,56], commonly occurring blur or noise patterns[166,258,416,417], and real-world changes in scene lighting[243,30,439], viewpoints[240,24,234,433,243,78,318], geometric modifications[31,414,424], color changes[185,330], and scene context[38,437].

There have been three broad approaches to address the lack of OOD generalization in DNNs: first, modifying the learning paradigm including modifying the architecture or loss function to enforce invariant representations[19,108,60,400,222], or using ensemble and meta-learning[220,21,447]; second, modifying the training data using data augmentation[434,167,420,172], or by increasing data diversity[416,331,298,343,393]; third, scaling data up to beyond billions of data points[421,301,279]. Despite these efforts, OOD generalization remains an unsolved problem for deep networks.

11.3 MacaqueITBench

We collected a large-scale dataset of neural population responses to over $300,000$ images across sessions, comprising $8,233$ unique natural images presented to seven monkeys over 109 sessions. In each session, a monkey maintained fixation while images were rapidly presented in random order. Each presentation was 83 milliseconds; with 83–150 milliseconds between presentations.

The images derived from published image sets[202] and photos taken in the lab and contained pictures of common objects, people, and other animals including monkeys (Fig. 11.1(b)). Image thumbnails are shown in Fig. 11.1(b)); sample images are provided in the supplement. Images belonged to over 300 semantic categories annotated by hand. A full list of categories can be found in the supplement. The large number and diversity of images allowed us to construct various OOD splits.

Neural responses were recorded on intracranial microelectrodes measuring extracellular electrical potentials (Fig. 11.1(c)) pre-processed to extract multi-unit spiking activity (Fig. 11.1(d))[45,360]. The analyses included 640 electrodes (12 multi-electrode arrays) recorded in nine hemispheres of seven monkeys, spanning four ventral-stream areas: V2, V4, central IT (CIT), and anterior IT (AIT), primarily sampling CIT and AIT (Fig. 11.1(e)). The electrodes were chronically implanted, and the responses showed stable selectivity when pooled across sessions. Nevertheless, our modeling focused on the more finely resolved within-session trial-averaged responses.

11.4 Constructing out-of-distribution data splits

We build on past work studying generalization under systematic distribution shifts[240,242,19,166], and define the training and test distributions parametrically using image attributes.

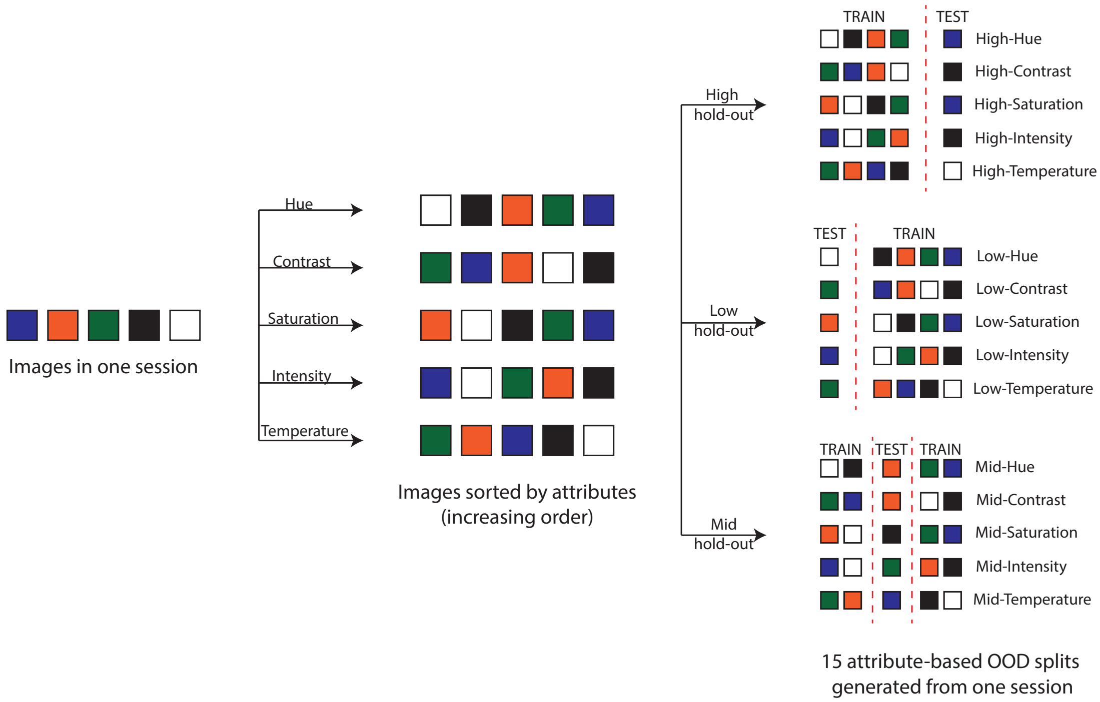


Figure 11.2: Constructing multiple attribute-based OOD splits. For each of our 109 sessions, we construct 15 different attribute-based OOD splits. These correspond to 3 hold-out strategies (high, low, mid) for each of 5 image-computable attributes (hue, contrast, saturation, intensity, temperature). For each attribute (e.g., hue), we compute the attribute value for each image in the session. For the high hold-out strategy, all images with the attribute value above a percentile cut-off serve as the OOD test set with the remaining serving as the train set. Analogously for the low hold-out splits, images below a percentile cut-off serve as the test set with the remaining serving as the train set. For mid hold-out splits, images within the middle percentiles serve as the test set.

Using these parametric data distributions, we construct three kinds of train-test splits:

InDistribution (InD) splits: For each session, we created one In-Distribution (InD) split to compare with OOD generalization performance. We sampled 25% of the images at random, and held these out as the InD test set, with the remaining serving as the training set.

Attribute-based OOD splits: For concreteness, we describe OOD splits based on image contrast; splits based on the other image attributes were constructed analogously. For each session, we computed the contrast value for each image. Then, one of three strategies were employed:

- High hold-out: The 75th percentile of contrast values served as the cut-off. Images with contrast above the cut-off formed the test set. Remaining images formed the training set.
- Low hold-out: The 25th percentile served as the cut-off. All images below this served as the held-out test set. The remaining served as the training set.
- Mid hold-out: Images with contrast values between the 42.5^{th} and 62.5^{th} percentile served as the held-out test set. The remaining formed the training set.

Cosine Distance-based splits: To investigate the relationship between the size of distribution shift and neural predictivity, we constructed 3 additional test splits. We first extracted the features for every image from the pre-final layer of a pre-trained ResNet18. A random image was picked to be the seed, and all images in the session were sorted in order of increasing cosine distance between the ResNet extracted features of the images and the seed. The sorted images were then divided into three chunks based on percentile cut-offs. The first chunk corresponds to the bottom 80 percentile which served as the Training + In-Distribution Test split. A random subset of this first chunk was held out to form the In-Distribution test split, with the remaining serving as the training set. The second chunk is images in the 90th to 95th percentile, which are held-out as the Near-OOD test split. Finally, the third chunk corresponds to images above the 95th percentile. These are held-out as the Far-OOD split. To ensure a gap between the train and test distributions, we discard images between the 80th and the 90th percentile. Note that the number of images in the In-Distribution test split was kept the same number of images as the Near-OOD split.

11.5 Quantifying distribution shifts

We present a unified framework for measuring distribution shifts over the parametric OOD train-test splits presented in Sec. 11.4.

11.5.1 Representations for training and testing data-splits

Let $D_T = \{i_1^T, i_2^T, ..., i_N^T\}$ denote a train split of N images, and let $D_t = \{i_1^t, i_2^t, ..., i_n^t\}$ denote the corresponding test split of n images. $\mathcal{R}(.)$ is a representation function that provides a vector representation for an image. The train and test images thus correspond to $\mathcal{R}(D_T) = \{\mathcal{R}(i_1^T), \mathcal{R}(i_2^T), \ldots, \mathcal{R}(i_N^T)\}$ and $\mathcal{R}(D_t) = \{\mathcal{R}(i_1^t), \mathcal{R}(i_2^t), \ldots, \mathcal{R}(i_n^t)\}$.

We analyzed representations $\mathcal{R}(i_j)$ formed by the features extracted for an image i_j by a pre-trained DNN. We explore 8 different DNN architectures, and multiple layers for every architecture. Equations below are agnostic to the architecture and the layer used. Other alternatives could include using HOG[83] or GIST[274] image features, or the vectorized pixel values of the image.

11.5.2 Defining distances over different datasets

To compute the shift between $\mathcal{R}(D_T)$ and $\mathcal{R}(D_t)$, we compared three distance metrics:

Maximum Mean Discrepancy (D_{MMD}): The MMD between the two datasets can be computed as

$$\begin{aligned} D^2_{MMD}(D_T, D_t) = \frac{1}{N^2}\sum_{j=1}^{N}\sum_{k=1}^{N} K(\mathcal{R}(i_j^T), \mathcal{R}(i_k^T)) + \frac{1}{n^2}\sum_{j=1}^{n}\sum_{k=1}^{n} K(\mathcal{R}(i_j^t), \mathcal{R}(i_k^t)) \\ - \frac{2}{Nn}\sum_{j=1}^{N}\sum_{k=1}^{n} K(\mathcal{R}(i_j^T), \mathcal{R}(i_k^t)) \end{aligned}$$

Here, $K(\mathcal{R}(i_j^T), \mathcal{R}(i_k^t))$ is a kernel distance between the representations of images i_j^T and i_k^t. A common choice for the kernel function $K(\cdot, \cdot)$ is the Gaussian RBF.

Covariate-Shift (D_{Cov}): Let $P_T(X)$ and $P_t(X)$ denote the distributions of the train and test input variables (i.e., image representations), and let $P(Y|X)$ denote the conditional distribution of the output (i.e., neural responses) given the input. Covariate shift exists if $P_T(X) \neq P_t(X)$ but $P_T(Y|X) = P_t(Y|X)$. D_{Cov} can be computed by training a binary classifier to classify if data comes from the training or the testing dataset. We denote the accuracy of this classifier as $a_{T,t}$ and measure the covariate shift as:

$$D_{Cov}(D_T, D_t) = 2 \times (0.5 - a_{T,t})).$$

Closest Cosine Distance (D_{CCD}): For every image in the test set, we find its distance to the closest training image, and compute the mean of this distance over all test images. For brevity, we will refer to this as Closest Cosine Distance. Let $i_k^T \in D_T$ denote the closest training image to test image $i_j^t \in D_t$ as measured by the cosine distance $D_{\cos}(\mathcal{R}(i_j^t), \mathcal{R}(i_k^T))$. The distance $D_{\cos}$ between two vectors u and v is given by

$$D_{\cos}(u, v) = 1 - \frac{u \cdot v}{||u|| \, ||v||}$$

The average distance to the closest training image is

$$D_{CCD} = \frac{1}{n} \sum_{j=1}^{n} \min_{k \in \{1,2,\ldots,N\}} D_{\cos}(\mathcal{R}(i_j^t), \mathcal{R}(i_k^T))$$

11.6 Model training and evaluation

As depicted in Fig. 11.1(a), we employ a linear model to map pre-trained model activations to neuronal firing rates from the IT cortex (Fig. 11.1(a)). The linear model was learned using ridge regression. We used only pre-trained DNNs, not DNNs fine-tuned for our analysis.

For feature extraction, we investigated 8 DNN architectures and 2 layers for each architecture. The DNNs include supervised models trained on ImageNet (ResNet-18[162], ViT[99]), self-supervised models trained on billion-scale data with self-supervised and weakly supervised learning (ResNet18_swsl[421], ResNext101_32x16d_swsl[421], ResNet-50_ssl[421]), Noisy student with EfficientNet[418], self-supervised learning over billions of tokens (DinoV2[279]), and the multi-modal vision-language model CLIP[301].

A linear encoding model was fit for the trial-averaged responses of each neuron in a session. The results are presented as the mean and S.E.M. across 109 sessions (7 monkeys); each session's results is the median across neurons. The model fit per neuron was quantified as the ceiling-normalized, squared Pearson's correlation, $r^2_{\text{pred}}/r^2_{\text{cons}}$ following convention[322,415] and related to the explained variance, R^2. The ceiling r_{cons} of a neuron was calculated as its response correlation between split-half trials, across images, with Spearman-Brown correction (because models fitting used all trials per image). The model fit r_{pred} was the correlation across test images between neuronal responses and model predictions. All experiments were conducted on a compute cluster with 300 nodes, 48 cores per node. CPU machines running Rocky Linux release 8.9 (Green Obsidian) were used.

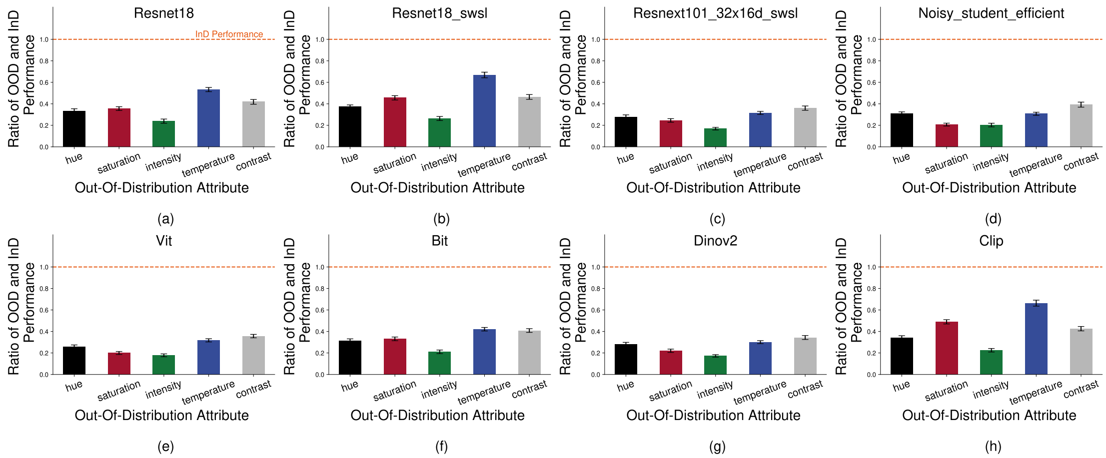


Figure 11.3: Neural predictivity drops under distribution shifts. The y-axis shows the ratio of the neural predictivity for out-of-distribution (OOD) images to in-distribution (InD) test images. A ratio of 1 would indicate no drop in performance. Each panel (a-h) shows a different architecture used for extracting image features. Each bar in a panels corresponds to a different OOD split constructed by using the high hold-out strategy across 5 different attributes (hue, saturation, saturation, intensity, temperature, and contrast). For all architectures and OOD splits, models fail to generalize well to OOD samples and are significantly and substantially below the 1.0 horizontal line. Image features were extracted from the pre-final layer for all architectures.

11.7 Results

11.7.1 Neural prediticivity drops under distribution shifts

DNN-based encoding models become worse at predicting neuronal responses under simple shifts in the image distribution. To demonstrate this, we report the ratio of neural predictivity between OOD and In-Distribution test splits (r^2_{ood}/r^2_{ind}). A ratio of 1 would indicate that models generalize equally well to InD and OOD test images (horizontal dashed line; Fig. 11.3a). In contrast, the OOD/InD performance ratios are substantially lower than 1. For instance, the black bar in Fig. 11.3a shows that the model's neural predictivity was 0.33 on high-hue OOD images (constructed using the high hold-out strategy in Sec. 11.4) compared to images with InD hue. Models show a similar lack of OOD generalization to OOD images with regard to saturation (red

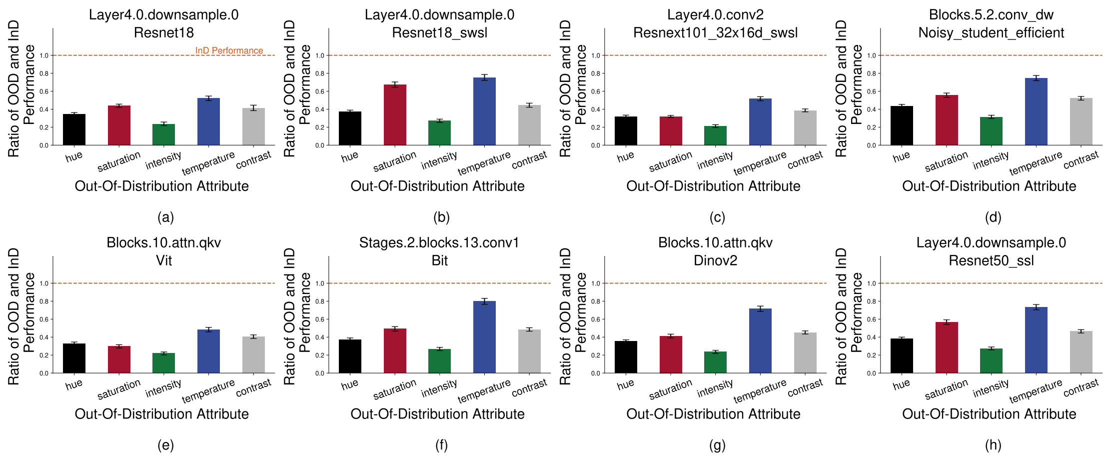


Figure 11.4: Neural predictivity drops for different model layers as well. Neural predictivity on OOD samples is reported for multiple DNN architectures across multiple different layers. Layer name is mentioned alongside architecture in all panels (a-h). All OOD splits reported here were constructed using the high hold-out strategy. For all architectures, layers, and OOD splits, models fail to generalize well to OOD samples and are significantly below the 1.0 horizontal line.

bar), intensity (green bar), temperature (blue bar), and contrast (gray bar). This performance drop was observed for all eight DNNs tested (Fig. 11.3b-h) and ranged from a best-case ratio of 0.66 for the CLIP model generalizing to high-temperature OOD images to a worst-case ratio of 0.2 for the ViT model generalizing to high-saturation OOD images.

The lack of OOD generalization by neuron encoding models extended to models based on intermediate DNN layers, not just the penultimate layer. Fig. 11.4 reports OOD/InD generalization performance ratios of models based on activations extracted from intermediate DNN layers (layer names shown in Fig. 11.4). For all architectures, OOD performance was substantially lower than InD performance.

The lack of model OOD generalization extended to different hold-out strategies. Fig. 11.5 shows the OOD/InD model performance ratio for OOD splits constructed using the low hold-out strategy described in Sec. 11.4. OOD performance was lower than InD (ratios below 1) for all architectures and image attributes. Additional results

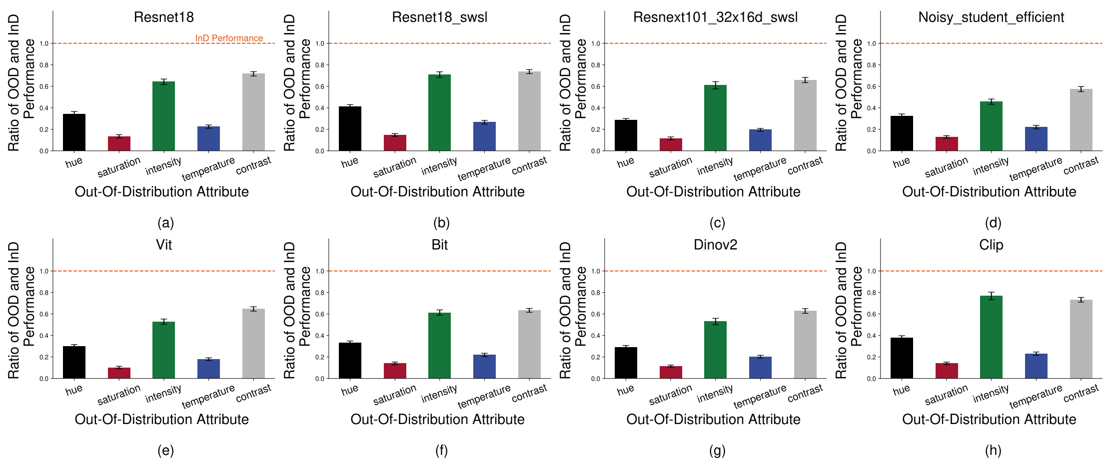


Figure 11.5: Neural predictivity drops for the low hold-out strategy as well. Neural predictivity is reported on OOD test splits constructed using the low hold-out strategy. Across all DNN architectures and image-computable attributes, performance is below 1.0 for all panels (a-h). Thus, models do not generalize well to OOD splits constructed with the low hold-out strategy as well.

with the mid hold-out strategy are provided in the supplement.

Combined, these results showcase a problem for current DNN-based models of the visual cortex—despite their ability to predict neural responses to in-distribution test images, the models generalize poorly under distribution shifts even in low-level image attributes.

11.7.2 The distance between train and test distributions explains generalization performance

The results above raise a natural question—when and how do models of the ventral visual cortex fail to generalize under distribution shifts? Theoretical work has related OOD generalization to the amount of distribution shift [49,285]. Here we apply this theoretical framework to characterize generalization in DNN models of the brain.

Intuitively, model generalization should be worse for train-test splits under larger distribution shifts. We tested this intuition by constructing splits with different lev-

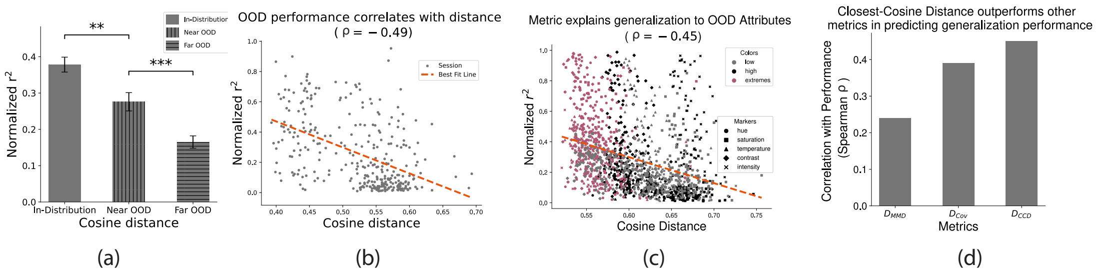


Figure 11.6: Closest-Cosine Distance metric well-explains performance across all attribute-based OOD splits. (a) Neural predictivity on distance-based splits. Models performed best on In-Distribution (InD) the split, with a dip in performance from InD to Near OOD test set (two-sided t-test, $p < 0.01$), and from Near OOD to Far-OOD (two-sided t-test, $p < 0.01$). This suggests a relationship between the extent of distribution shift and generalization performance. (b) OOD performance can be well-explained by the distribution shift. For all 109 sessions, the plot shows performance on the InD, Near-OOD, and Far-OOD with the corresponding distribution shift measured using the Closest-Cosine Distance metric (D_{CCD}). Performance and D_{CCD} have a Spearman correlation of $-0.49 (p < 0.001)$. (c) Scatter plot of neural predictivity and the corresponding distribution shift (D_{CCD}) across all 15 attribute-based OOD splits for all 109 sessions. Generalization performance and the proposed distance metric have a Spearman correlation of $-0.45 (p < 0.001)$ (d) Comparing different distance metrics w.r.t. their correlation with OOD performance. The proposed Closest-Cosine Distance has the highest correlation with neural predictivity, outperforming both MMD (D_{MMD}) and Covariate-Shift (D_{Cov}).

els of distribution shifts—InD, Near OOD, and Far OOD. As described in Sec. 11.4, images in every session were sorted based on cosine distance, and split into three chunks. The first chunk comprises the training and the In-Distribution test set, while the second and third chunks form the Near OOD and Far OOD test sets. As hypothesized, the model performance decreased progressively and significantly from In-Distribution to Near OOD, then Far OOD test distributions (Fig. 11.6(a); two-sided t-test, $p < 0.01$).

Beyond category-level differences, the size of the distribution shift predicted the OOD model performance drop across individual data splits (Fig. 11.6(b)). The distribution shift between each pair of train and OOD test distributions was quantified with the Closest Cosine Distance (D_{CCD}; described in Sec. 11.5). The D_{CCD} strongly correlated with the OOD model performance drop (Spearman correlation $\rho = -0.49$).

The distribution shift (D_{CCD}) calculated from ResNet features also explained OOD performance for attribute based splits (Fig. 11.6(c). Across all image attributes (hue, saturation, temperature, contrast, intensity) and hold-out strategies (low, high, mid) used to create OOD splits, D_{CCD} correlated with OOD model performance drop with a Spearman correlation coefficient $\rho = -0.45$. Compared to two other popular measures of the sizes of distribution shifts (MMD, D_{MMD}[136] and Covariate-Shift, D_{Cov}[357]; Sec. 11.5), our proposed Closest Cosine Distance (D_{CCD}) metric best predicted OOD model performance (Fig. 11.6 (d)).

11.8 Conclusions

These results reveal a deep problem in modern models of the visual cortex: good prediction is limited to the training image distribution. Simple distribution shifts break DNN models of the visual cortex, consistent with broader findings that the underlying DNNs are brittle to OOD shifts. Going one step further, we introduce an image-computable metric that significantly predicts the generalization performance of models under distribution shifts. This metric can help investigators gauge how well a neural model fit on one dataset may generalize to novel images.

Our findings underline an important limitation of AI models for Neuroscience. Fields like Computer Vision have responded to the issue of distribution shifts by collecting progressive larger datasets, hoping models will learn to generalize to most images[61,46,294,268] at the billion-image scale. However, it is infeasible to achieve the same scale in neuroscience—the time needed to present a billion images is already a formidable challenge, not to mention the resource intensiveness of data collection. We hope our characterization of when and how modern models of the visual cortex fail out-of-domain will motivate the development of data-efficient ways to improve DNN generalization.

11.9 Limitations

In this work, we have explored the impact of OOD samples on DNN-based models of the visual cortex. Our analyses have two main limitations that we hope future research can address. First, we did not fine-tune the DNNs on neural data. It is possible that training these models on the specific images and/or neural data can help improve generalization. Second, we did not explore the contributions of the images being OOD for the underlying pre-trained DNNs, as we only fit the linear encoding models on train set images and neural data. Because our images were naturalistic, it is plausible that they belonged to the training distribution of the pre-trained models we used, some of which (e.g., CLIP) having hundreds of millions of images. An interesting future direction will be to examine how the model performance is affected by using out-of-distribution images for the pre-trained DNNs. These images could include those from ImageNet-P, ImageNet-C[166], and evolved images[292].

The more you know, the more you know you don't know.

Aristotle

12 Conclusion

Over the past six years, I have sought to understand and unravel the complexities of how intelligent systems can go past the data, and learn knowledge that can truly adapt and generalize to the unseen. This journey has led me through the cutting-edge of several disciplines and methodologies across the sciences. Not just machine learning and artificial intelligence, this thesis has delved deep into fundamental concepts in computer graphics, cognitive science, neuroscience, and even philosophy. By examining the problem through these diverse lenses, I aimed to uncover the principles that underlie the adaptive intelligence observed in both biological systems. This multidisciplinary approach was not just a methodical choice but a reflection of my evolving understanding of the problem, influenced by my advisors, collaborators, and the broader scientific community. The outline below aims to capture the different perspectives discussed in this thesis, and highlight the primary contributions.

Our journey began in Chapter 1, where the context and importance of OOD generalization were established. We began with the compelling example of Captain Sully Sullenberger, who performed an emergency landing on the Hudson River—a scenario he had never explicitly trained for. This real-life event highlights the human capacity to adapt to novel situations, a capability that current AI systems lack. Juxtaposing the adaptability of humans with the struggle of modern AI set the stage for the thesis.

In Chapter 2, we delved into the philosophical roots of generalization, tracing the evolution of ideas that have shaped our understanding of knowledge and learning. Philosophers such as John Locke, David Hume, Immanuel Kant, and Charles Darwin laid the groundwork for concepts that are central to modern AI research. Locke's empiricism, which posits that knowledge is derived from sensory experience, and Hume's skepticism about the certainty of inductive reasoning, both resonated with the challenges faced by AI in generalizing from training data to new, unseen situations. Kant's idea of a priori knowledge, suggesting that some inherent structures allow humans to organize and interpret sensory data, paralleled current attempts in AI to incorporate inductive biases for better generalization. Darwin's insights into natural selection and adaptation provided a biological perspective on how organisms handle variability and change, offering valuable lessons for developing adaptable AI systems. This historical and philosophical context was crucial for understanding the theoretical underpinnings of OOD generalization, bridging the gap between abstract philosophical inquiry and modern AI. This included laying out the mathematical formalization of key concepts such as data distributions, joint distributions, and distribution shifts, which are essential for systematically analyzing how AI models learn and generalize. The concept of Empirical Risk Minimization (ERM) was also introduced, highlighting its role in guiding model selection and its limitations in OOD scenarios—ERM focuses on minimizing loss over the training data, but it does not guarantee good performance on data that deviates from the training distribution.

Building on this foundation, Chapter 3 introduced our general approach—studying OOD generalization was presented, which involved the use of controlled environments and computer-generated datasets. This method, akin to Darwin's strategy of studying domesticated species to understand natural selection, allowed for precise control and manipulation of variables, providing deeper insights into the factors that influence generalization and revealing the limitations of current AI models. By creating datasets with well-defined distribution shifts, the thesis enabled a systematic study of generalization across various real-world factors such as lighting, viewpoints, materials, shapes, scene contexts, and many more. Chapter 3 showed that a huge factor in improving OOD generalization is data diversity. A smaller but more diverse dataset significantly beat a larger but less diverse dataset when tested under OOD settings. In explaining what drives this phenomenon, we discovered that a diverse dataset leads to building of increased invariances in the learned representations, which results in better generalization.

Chapter 4 dug deeper into the phenomenon, trying to understand how generalization happens at mechanistic level. We employed the analytical tools often utilized in Neurosciences to understand DNNs. Foreshadowing, Chapter 11 will do the same, but with actual brains. We zoomed in to the ability of these machines to generalize to unseen transformations (e.g., rotation) as they are shown more diverse data. We found that the network disseminates orientation-invariance from fully-seen instances (objects for which all possible transformations are seen) to partially-seen instances using brain-like mechanism like those reported in Neuroscience. Thus, machines truly do learn like the brain in some sense.

Building on our findings in Chapters 3 and 4, Chapter 5 sought to make an engineering application utilizing these findings. To this end, we collected a large scale dataset of everyday objects photographed under varying lighting and viewpoints with a robotic arm, and designed methodologies to improve generalization across these transformations be enforcing invariance (as defined in Chapters 3 and 4). We found, that doing so let to significant improvements in performance. Watching our scientific findings come full circle to drive engineering gains was both inspiring and impactful.

Chapter 6 continued similar ideas, but expanded beyond supervised learning—to understand and improve generalization in Reinforcement Learning. There, such controlled analysis of generalization led us to another striking similarity between biological and artificial intelligence. Conventional wisdom suggests that RL agents perform best when tested on the same environment as they were trained on. But, we found that just like humans, RL agents learn best when the noise in the system is removed. We called this the Indoor Training Effect. Similar to how training in a calm, noise-free indoor environment helps athletes focus on mastering the fundamentals of a sport, we explored whether training on certain environments is more conducive to learning than training on the same testing environment

Given the several benchmarks discussed in the previous chapter showcasing the inability of AI models at generalizing, Chapter 7 went back to the drawing board with one fundamental question—Are these models even safe inside the data distribution? To utter dismay, we found that they are not. Our findings showed that while data augmentation, unbiased datasets, and specialized shift-invariant architectures would certainly be helpful, the real problem runs far deeper. Despite high test accuracies, networks are plagued by adversarial examples that lie within the training distribution. These errors present a grave challenge for AI, as this shows that a malicious agent no longer needs to add engineered noise to induce an error.

Chapter 8 shifted the focus to biological systems, examining how they achieve OOD generalization. We introduced the Out-of-Context Dataset (OCD), and it to systematically and quantitatively study the role of context in object recognition. Most interestingly, this dataset put human vision under the microscope, and we found that removing real-world attributes like gravity can hugely impact human vision. We showed consistent results for humans and computational models—contextual cues can enhance visual recognition, but also that the "wrong" context can impair visual recognition capabilities both for humans and models.

Continuing further with evaluating the gap between human and artificial intelligence, Chapter 9 presented a thorough analysis of this gap as perceived by other humans. One way to think of it is, instead of the generalization gap being measured with a numeric metric of accuracy, in this chapter we conducted Turing Tests which measured how this gap is perceived by humans themselves. The datasets and evaluations introduced here are extensive (25,650 Turing test trials, 549 humans contributing to the dataset, and 1,126 human judges). Across 6 common language and vision tasks, we showed that current AI algorithms are not far from being able to imitate humans in these tasks. That is, humans couldn't easily tell if the task was completed by a human or by an AI agent.

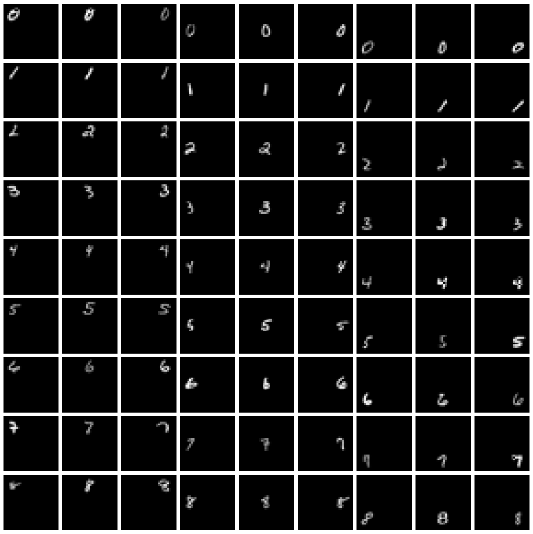
Insights from these past chapters paved the way for Chapter 10. An important thread raised in Chapters 3-5, was that invariances are built by looking at diverse data. Chapter 3 showed models can generalize without seeing all possible combinations, chapter 4 explained how this happens, and chapter 5 harnessed it mathematically. However, an important question this raises is what do we mean by diversity. For instance, a dataset containing diversity in 3D objects is very different from a dataset containing one object viewed densely from all viewpoints. Chapter 10 takes inspiration from human vision, and defines a diverse visual diet as one which is most human-like. To this end, we presented the Human Visual Diet (HVD) dataset, which focussed on two key aspects of the typical human visual diet which are currently absent in all large-scale internet-scraped datasets. Firstly, children learn from sampling few 3D scenes under diverse real-world transformations. Secondly, they always view objects in context of one-another. We already proved the importance of data diversity and scene context in previous chapters. Here, we provided a working definition for how these could be modeled. We showed that on extensive benchmarks, generalization improved significantly when the training diet consisted of these attributes. In fact, with even 1000-fold less data, models trained with such a diet beat models trained on large-scale non-human-like diets.

Finally, Chapter 11 went one step deeper into biological systems. So far, all chapters pertaining to biological intelligence were only testing the system at a behavioural level. However, the results so far make it clear that taking inspiration from biological intelligence can help make AI much stronger. So, in order to better understand how the gold standard i.e., brain achieves generalization, Chapter 11 looked at the generalization capabilities of models trained on electrophysiological recordings from Macaque Brains. We explored the impact of OOD samples on DNN-based models of the visual cortex, and found that modern models of the visual cortex which rely on DNNs achieve good prediction only with test data limited to the training image distribution. Simple distribution shifts break DNN models of the visual cortex, consistent with broader findings in the previous chapters.

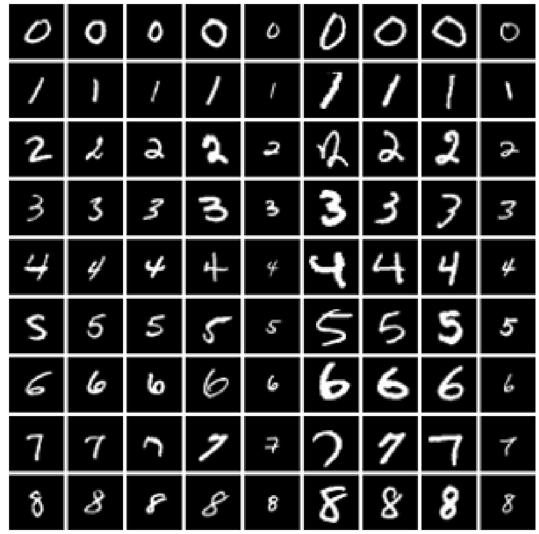
In conclusion, this thesis explored OOD generalization by bridging the gap between biological and artificial intelligence. By drawing on insights from philosophy, neuroscience, and machine learning, this thesis hopes to have offered a multifaceted perspective on one of the most pressing challenges in the field of AI. We hope that the strategies and findings presented will pave the way for future research and innovation aimed at building intelligent systems that are not only capable of learning from data but also of adapting to the ever-changing complexities of the real world.

Part V

Appendix



(a) MNIST-Position



(b) MNIST-Scale

Figure Supp.1: Combinations grids for MNIST-Position and MNIST-Scale. Each row represents images from a category and each column from a viewpoint. (a) MNIST-Position was created by adding viewpoint in the form of position to images. For this, MNIST images were placed into one of nine positions in an empty three-by-three grid with equal probability. (b) MNIST-Scale was created by resizing images from MNIST to one of nine possible sizes, and then zero-padding.

S1 Additional details on Datasets

S1.1 Samples from MNIST-Position and MNIST-Scale datasets

Fig. Supp.1 presents one representative example for each category-viewpoint combination through the combinations grid for the MNIST-Position and MNIST-Scale datasets.

S1.2 Rendering Pipeline for Biased-Cars Dataset

To generate photo-realistic data with systematic, controlled biases we implemented our computer graphics pipeline which offered us fine grained control over scene attributes including but not limited to - backgrounds, textures, lighting and geometry. Below we present the details of our rendering pipeline, along with some sample images.

Pipeline Details: We used Esri CityEngine[267] to model the city layout and geometry, to which we add 3D assets - car models, pedestrians, trees, street furniture like bus stops, textures for buildings, roads and car paints. Blender Python API[77] is used to modify the 3D city file. This includes placing vehicles and other assets at user defined locations, modifying their material properties including vehicle paint, adding specified textures to roads, buildings and pedestrians, and defining camera attributes (lens, field of view, motion blur etc) and camera locations. For randomization, a distribution over each parameters was defined. For instance, a discrete uniform distribution over possible car color paints. Similarly, we defined distributions over object positions in the city, camera viewpoint and distance, among other factors.

Sample images are shown in Fig. Supp.2 below, rendered at 1024×1024 pixels. As network input was 224×224, training images were rendered at 256×256 and then resized to 224×224 (as side length of the form 2^k lead to computational gains in physically based rendering). Physically based rendering accurately models the flow of light in the scene resulting in highly photo-realistic images. As can be seen, our pipeline reproduces lighting artefacts like color bleeding and specular highlights very gracefully. As shown, images include cars seen from different distances and viewpoints, under different lighting conditions, scene clutter and even occlusions.

Figure S.2: Sample images from the Biased Cars dataset

S2 Selectivity and Invariance

In the paper we defined the selectivity score of a neuron with respect to category and its invariance score with respect to viewpoint. Following the same notation as the paper: a_{ij}^k denotes the activations grid for neuron k, where each row represents one category and each column represents a viewpoint.

S2.1 Normalization of activations grid

For every neuron, we first normalize its activations for every image by dividing them by its maximum activation across all images. This ensures that that the activation for every image lies between 0 and 1 for all neurons. The entries of the activations grid for a neuron are then computed by averaging these normalized activation for images belonging to each category-viewpoint combination.

The activations grid is then normalized to be between 0 and 1. To do so, we subtract the minimum of the activations grid and then divide it by the maximum.

S2.2 Selectivity and Invariance with respect to viewpoint

In the paper, we used $i^{\star k}$, S_c^k, I_v^k to denote the preferred category, selectivity score with respect to category and invariance score with respect to viewpoint respectively. We also presented these equations to compute these quantities:

$$i^{\star k} = \arg\max_i \sum_j a_{ij}^k. \tag{Supp.1}$$

$$S_c^k = \frac{\hat{a}^k - \bar{a}^k}{\hat{a}^k + \bar{a}^k}, \quad \text{where } \hat{a}^k = \frac{1}{N}\sum_j a_{i^{\star k}j}^k, \;\; \bar{a}^k = \frac{\sum_{i \neq i^{\star k}} \sum_j a_{ij}^k}{N(N-1)}. \tag{Supp.2}$$

$$I_v^k = 1 - \left(\max_j \;\; a_{i^{\star k}j}^k - \min_j \;\; a_{i^{\star k}j}^k\right) \tag{Supp.3}$$

We now present how to compute the selectivity with respect to viewpoint, and invariance with respect to category, denoted as S_v^k and I_c^k respectively. These can be obtained by first finding the preferred viewpoint, denoted as $j^{\star k}$, and proceeding as in the above equations:

$$j^{\star k} = \arg\max_j \sum_i a_{ij}^k. \tag{Supp.4}$$

$$S_v^k = \frac{\hat{a}^k - \bar{a}^k}{\hat{a}^k + \bar{a}^k}, \quad \text{where } \hat{a}^k = \frac{1}{N}\sum_i a_{ij^{\star k}}^k, \;\; \bar{a}^k = \frac{\sum_{j \neq j^{\star k}} \sum_i a_{ij}^k}{N(N-1)}. \tag{Supp.5}$$

$$I_c^k = 1 - \left(\max_i \ a_{ij^{\star}k}^k - \min_i \ a_{ij^{\star}k}^k \right) \tag{Supp.6}$$

Observe that like S_c^k, S_v^k is a value between 0 and 1, and higher value indicates that the neuron is more active for the preferred viewpoint as compared to the rest of the viewpoints. I_c^k too is a value between 0 and 1, with higher values indicating higher invariance to the category for images containing the preferred viewpoint.

S3 Experimental Details and Hyper-Parameters

Each of our four datasets contains both category and viewpoint labels for all images. We define the location and the scale as the viewpoint for MNIST-Position and MNIST-Scale datasets respectively. For both iLab and Biased-Cars dataset, the viewpoint refers to the azimuth viewpoint. Networks are trained to classify both category and viewpoint labels simultaneously, and all models are trained from scratch, without any pre-training to ensure controlled testing. This ensures that any existing biases in common pre-training datasets like ImageNet[373] do not impact our results.
Number of Images: The number of training images is kept fixed for every dataset, and was decided by training networks on these datasets while gradually increasing size, till the performance on OOD combinations saturated. For the Biased-Cars dataset, performance plateaud at 3,400 train, 445 validation, and 800 OOD test images. For iLab, we used 70,000 train, 8,000 validation images, and 8,000 OOD test images. As the iLab dataset is a natural image dataset, it required much more images to saturate. For MNIST, 54,000 train, 8,000 validation and 8,000 test images were used.
Hyper-parameters: We used the Adam[196] optimizer with 0.001 as learning rate, and ReLU activations. For the Biased-Cars datasets, all models were trained for 200 epochs, while we trained for 50 epochs for the iLab dataset. MNIST-Position and MNIST-Scale were trained for 5 epochs. These stopping criterion were picked to ensure convergence on generalization to OOD combinations. All experiments were repeated multiple times and confidence intervals (95%) are shown in the plots in the main paper. iLab and Biased-Cars experiments were repeated 3 times each, and MNIST experiments were repeated 10 times. Loss for training Shared architectures was simply the sum of CrossEntropy Loss for both category and viewpoint classification. We compared how different weighted sums perform, and found this to be performing best as measured by the geometric mean of category and viewpoint classification.

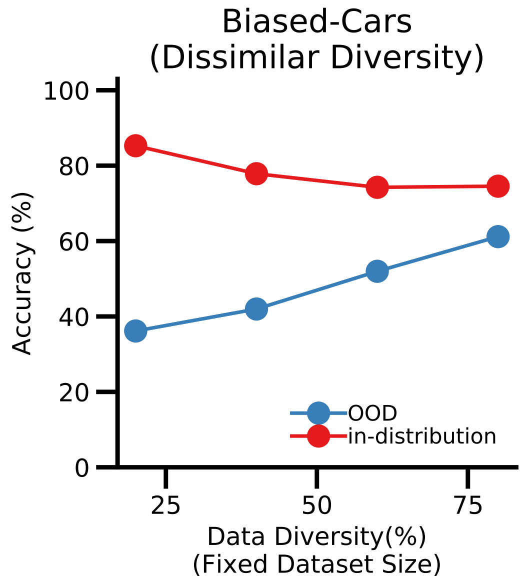


Figure Supp.3: Generalization of Separate architectures to OOD combinations as number of in-distribution combinations are increased while ensuring that viewpoints increasingly distant from the OOD set are added (Dissimilar Diversity). To control that the increasing generalization performance as the data diversity increases is not due to closer viewpoint angles between In-distribution and OOD combinations, we created a dataset split where the In-distribution combinations start closest to the OOD combinations (in terms of viewpoint angle), and become increasingly distant. Thus, as In-distribution combinations are increased, the training data becomes increasingly dissimilar to the OOD combinations (test set). The results show that the performance on the OOD combinations still improves as data diversity increases. This experiment discards the hypothesis the increase in generalization performance is due to having closer viewpoint angles between the In-distribution and OOD combinations

.

S4 Additional Experiments:“When Do CNNs generalize to OOD combinations?”

Below we present additional results that re-inforce our findings presented in the results sections of the main paper.

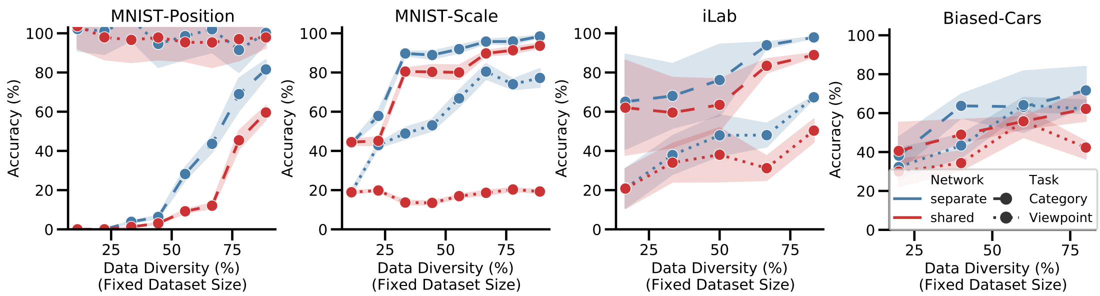
S4.1 Similarity between In-Distribution and OOD Combinations

To discard that the increasing generalization performance as the data diversity is increased is not due to having closer viewpoint angles between the in-distribution and OOD combinations, we provide the following control experiment. For the smallest number of in-distribution combinations, we use the combinations that are the closest to the OOD combinations (i.e., consecutive bins). As we increase the number of in-distribution combinations, we keep adding the rest of in-distribution combinations in the order of closeness to the OOD combinations. Fig. Supp.3 show a clear increase of the accuracy in OOD combinations. The increase of accuracy in this experiment can not be explained by the fact that the in-distribution combinations tend to be more similar to the OOD combinations when increasing the data diversity, because in this experiment, the combinations tend to be more dissimilar as increasing the data diversity. Thus, this experiment discards that the increase in generalization performance is due to having closer viewpoint angles between the in-distribution and OOD combinations

S4.2 Separate performance of category and viewpoint classification

In Fig. Supp.4, we show the individual accuracy for category and viewpoint classification in OOD category-viewpoint combinations. The results show that Separate also obtains better accuracy than Shared for each individual task accuracy. Note that the relative accuracy of the two tasks varies depending on the dataset, and no task is consistently harder than the other across all datasets. For instance, viewpoint classification is easier for MNIST-Position, while it is significantly hard for MNIST-Scale. MNIST digits are centered by default, and when placed in different positions to create MNIST-Position images, the viewpoint is easily distinguishable. For MNIST-Scale however, there is little visual variation between adjacent scales, which leads to a poor Top-1 classification accuracy for viewpoint (scale) classification.

Furthermore, we have found that for MNIST-Position, the pooling operation at the end of ResNet-18 is critical to obtain good generalization accuracy to OOD category-viewpoint combinations. We evaluated ResNet-18 without the pooling operation and the category recognition accuracy of OOD category-viewpoint combinations dropped to baseline. Pooling facilitates an increase of position invariance and it does not harm the viewpoint classification accuracy (as shown by[20], pooling does not remove the position information).



(a) (b) (c) (d)

Figure Supp.4: Generalization performance for category recognition and viewpoint estimation for Shared and Separate ResNet-18 as in-distribution combinations are increased for all datasets. The category recognition accuracy and viewpoint estimation accuracy are reported along with confidence intervals (95%) (a) MNIST-Position dataset. (b) MNIST-Scale dataset. (c) iLab dataset. (d) Biased-Cars dataset.

S4.3 Number of neurons in shared vs. separate networks

To control for the number of neurons in Shared and Separate architectures, we present additional results with the Biased-Cars dataset in Fig. Supp.5. In the paper, we presented the Shared-Wide architecture for the ResNet-18 backbone, which is the Shared architecture with double the number of neurons per layer, i.e., double the width. Here we go one step further and test a number of similar scenarios with the ResNet-18 backbone. The Separate Half and Separete One Fourth architectures are made by reducing the number of neurons in every layer to one half, and one fourth of the original number respectively. It is to be noted, that the Separate architectures has double the number of neurons as the Shared architecture, as there is no weight sharing between branches in the Separate case. Thus, the Separate Half architecture has the same number of neurons as the Shared architecture, and the Separate architecture has the same number as the Shared-Wide architecture. In a similar vein, the Shared Four Times was created by multiplying the neurons in each layer of the Shared architecture four times. Thus, the Shared Four Times has double the number of neurons as compared to the Shared Wide architecture, and 4 times the Shared architecture.

As can be seen in Fig. Supp.5, even at one-eighth number of neurons, the Separate One Fourth architecture substantially outperforms the Shared Four Times architecture at generalizing to OOD category-viewpoint combinations. This confirms that our findings are not a function of the number of neurons in the Shared and Separate architectures.

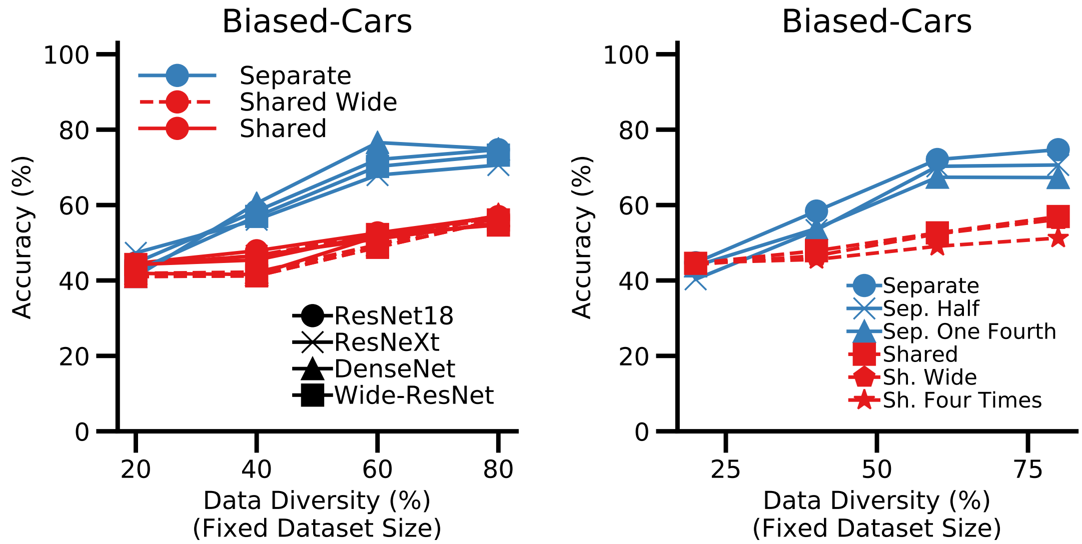


Figure Supp.5: Generalization to OOD combinations as number of neurons per layer are varied for the ResNet-18 backbone. Separate architectures substantially outperform Shared architectures across a range of widths, i.e., number of neurons per layer. The Separate architecture contains double the parameters as the Shared architecture, as there is no weight sharing in the Separate case. Variants of these architectures are created by increasing or decreasing the neurons in each layer by a factor of 2 at a time. Even at one-eighth the number of neurons, the Separate One Fourth architecture generalizes much better to OOD combinations as compared to the Shared Four Times architecture.

S4.4 Number of Training examples

To ensure that our findings are not a function of the amount of training data, we present the results for different number of images for the Biased-Cars and the iLab dataset in Fig. Supp.6. As can be seen in both these datasets, across a different number of images the Separate architecture substantially outperforms the Shared one at generalizing to OOD category-viewpoint combinations.

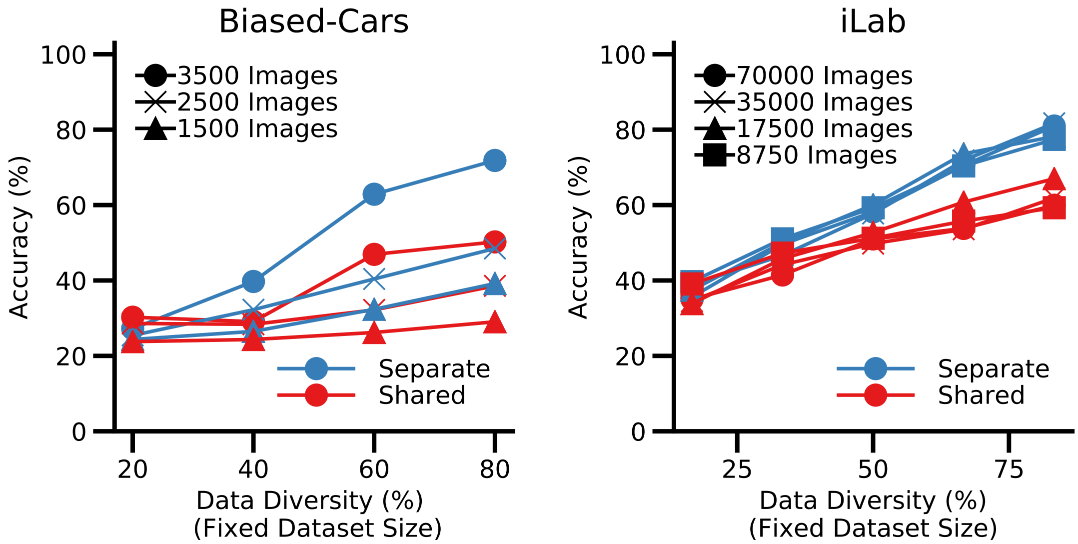
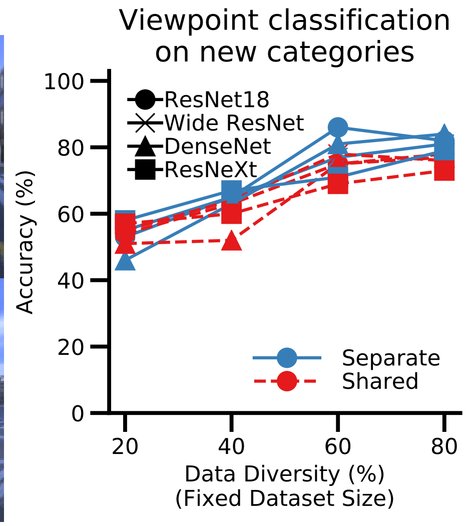


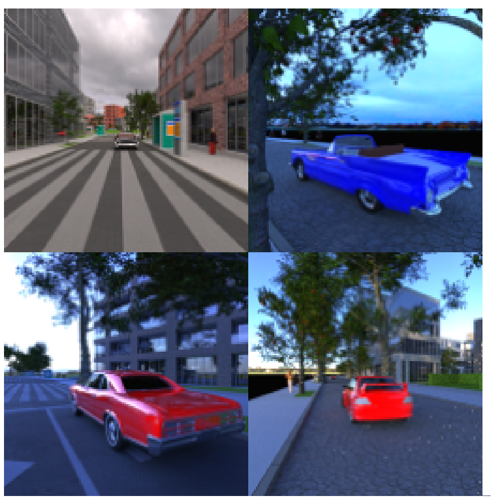
Figure Supp.6: Generalization to OOD combinations as number of training images is varied. For both iLab and Biased-Cars dataset, Separate architecture outperforms the Shared architecture trained with the same number of images.

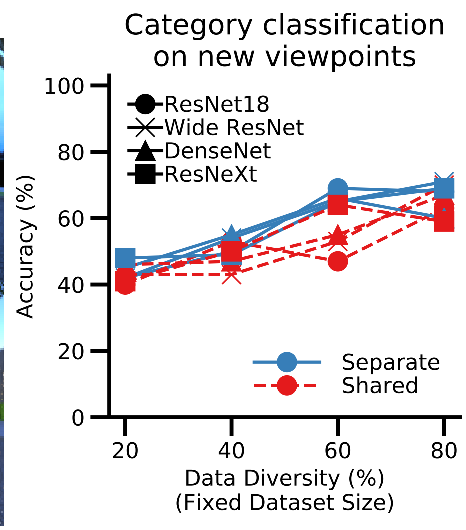
S4.5 Results on new categories and viewpoints

We also evaluated our trained CNNs on classifying the viewpoint of 4 new car categories (Fig. Supp.7a). Analogously, we also evaluated category classification in new viewpoints (side-to-back of car as in Fig. Supp.7c, instead of the front-to-side shown in training). As shown in Fig. Supp.7b and d, these results confirm that our conclusions also apply to new car categories and new viewpoints: generalization increases with more data diversity and Separate architecture.



(a) (b)





(c) (d)

Figure Supp.7: Controlling category and viewpoint separately. (a) Images of 4 new car categories, (b) Viewpoint classification accuracy for the 4 new car categories, (c) Images of new viewpoints, (d) Car category recognition accuracy for the new viewpoints.

S4.6 Task training order

We present results on the impact of the order in which networks are trained on category and viewpoint classification. Our networks contain three components: (i) shared layers, (ii) category branch and (iii) viewpoint branch. Here we start by training on

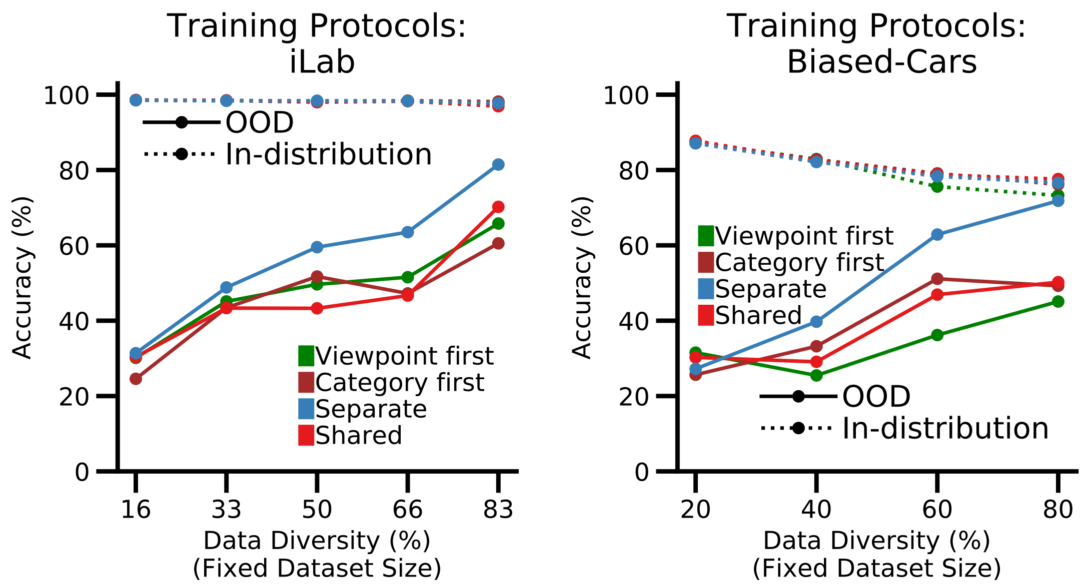


Figure Supp.8: Generalization performance for additional training protocols, besides Separate and Shared protocols, for the ResNet-18 backbone. The geometric mean of category recognition accuracy and viewpoint estimation accuracy is reported for OOD combinations as the number of in-distribution combinations is increased. For these, we start by training the shared network on only one task first, i.e., Viewpoint first or Category first. We then train the second task starting from these learned features from the first task. We present their comparison with our Shared and Separate training protocols presented in the main paper. (a) Accuracy for the iLab dataset, (b) Accuracy for the Biased-Cars dataset.

one task first, say Category recognition. We then train the other task, i.e., Viewpoint classification starting from these features learned from the first task. We call this the Category first protocol. The Viewpoint first protocol is defined analogously by starting with viewpoint classification first, and then training for category recognition. Results for these are provided in Fig. Supp.8.

As can be seen, our findings are consistent with these new protocols as well. The Separate architecture outperforms the Shared architectures independent of the training protocol. Furthermore, all architectures get better with OOD combinations as in-distribution combinations are increased.

S4.7 Results on additional datasets: UIUC 3D and MNIST-Rotation

Going beyond the four datasets presented in the main paper, we replicate our analysis on two additional datasets as a confirmatory experiment: (1) the UIUC 3D Dataset, and (2) the MNIST-Rotation dataset. As can be seen from Figs. Supp.9 (a) and (b), our findings are consistent across these additional datasets as well - Separate outperforms the Shared, and all architectures get better at OOD combinations as in-distribution combinations are increased.

Small size of UIUC 3D dataset: It is important to note that the small size of the UIUC 3D dataset makes it difficult to adapt it for training with biased in-distribution combinations. We picked 8 of the total 10 object categories (to ensure symmetry between tasks as explained in the paper), which amounts to 5,400 images in total across 64 category-viewpoint combinations. Thus, there are only 1700 training images for the 24 in-distribution combinations case, which is kept constant as in-distribution combinations are increased. In contrast, the other natural image dataset used in this paper, the iLab dataset contains 70,000 training images for 6 categories and viewpoints each. Due to this the generalization performance is slightly low, however the findings are still consistent as reported above. As an additional control, we also tried using all available 4500 images for the 87.5% seen case (i.e., all images other than the OOD test set) - generalization numbers were still low overall, but trends were preserved.

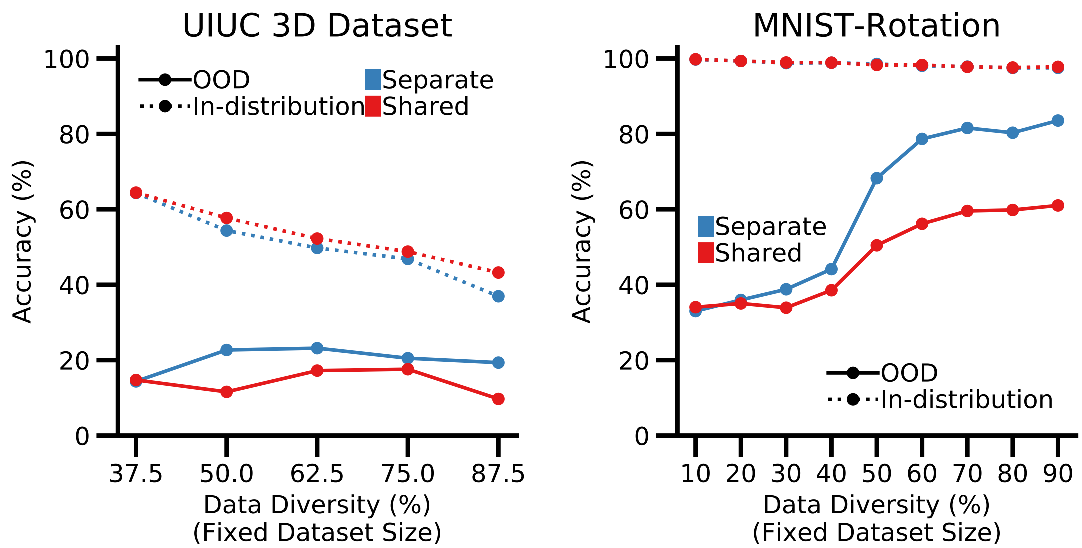


Figure Supp.9: Generalization performance for additional datasets for ResNet-18 backbone. The geometric mean of category recognition accuracy and viewpoint estimation accuracy is reported for OOD combinations as the number of in-distribution combinations is increased. (a) Accuracy for the UIUC 3D dataset, (b) Accuracy for the MNIST-Rotation dataset. Due to the small size of the UIUC dataset there is poor generalization - there are only 1700 train set for 37.5% in-distribution combinations (which is kept constant as the number of in-distribution combinations increases). This leads to lesser generalization, but our findings still hold true - (1) Increasing in-distribution combinations improves performance on OOD data, and (2) Separate architectures outperform Shared ones on OOD combinations.

S4.8 Results on group equivariant architectures

Group and gauge equivariant CNNs have recently emerged as an alternative to standard CNNs which theoretically offer better viewpoint invariance. While these architectures[72,73] are yet to be adapted to more complex datasets like ImageNet, they have shown great results on simpler image datasets like MNIST-Rotation. Here, we present results with two such architectures in Fig. Supp.10. As can be seen, our findings also extend to these architectures - Separate outperforms the Shared independent of the training protocol, and all architectures get better at OOD combinations as in-distribution combinations are increased. This suggests our findings extend beyond standard CNNs. We believe that a detailed comparison between GCNNs and standard CNNs with respect to generalization to OOD combinations would be an interesting starting point for future work.

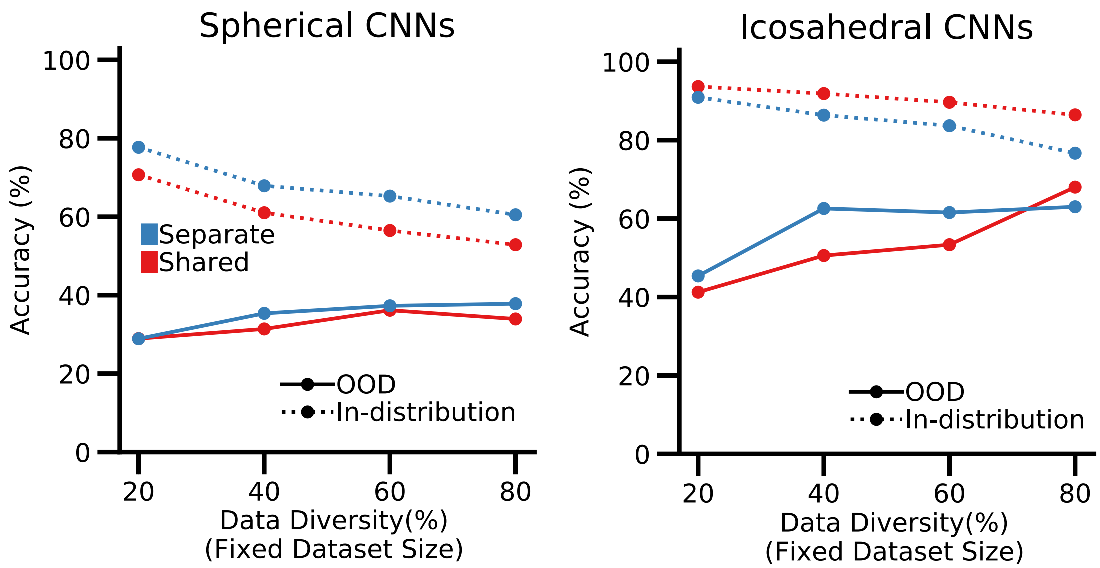


Figure Supp.10: Generalization performance for different group equivariant architectures as in-distribution combinations are increased for MNIST-Rotation dataset. The geometric mean of category recognition accuracy and viewpoint estimation accuracy is reported for OOD combinations as the number of in-distribution combinations is increased. (a) Accuracy of Separate and Shared architectures using a Spherical CNN[72] as backbone, (b) Accuracy using an Icosahedral CNN[73] as backbone.

S5 Additional Experiments for "How Do CNNs Generalize to OOD Combinations?"

S5.1 Ratio of Specialized Neurons

In the main paper, we have presented the ratio of specialized neurons for the iLab and Biased-Cars dataset. Here, we also provide these for the MNIST-Position and MNIST-Scale datasets. As can be seen, our findings are consistent across these datasets as well. Figs. Supp.11a and b show that neurons in the final convolutional layer specialize to become either category or viewpoint neurons as more category-viewpoint combinations are shown. Category and viewpoint branches of the Separate architecture become completely specialized to category and viewpoint, respectively. In the Shared architecture, both kinds of neurons emerge in roughly equal numbers.

We also observe that for a small number of in-distribution combinations, the ratio of neurons specialized for category or viewpoint classification may be impacted by the relative difficulty of these two tasks. We observe that when the accuracy is higher for category classification (shown in Fig. Supp.4), a higher fraction of neurons becomes specialized for category, as observed for the iLab and MNIST-scale datasets.

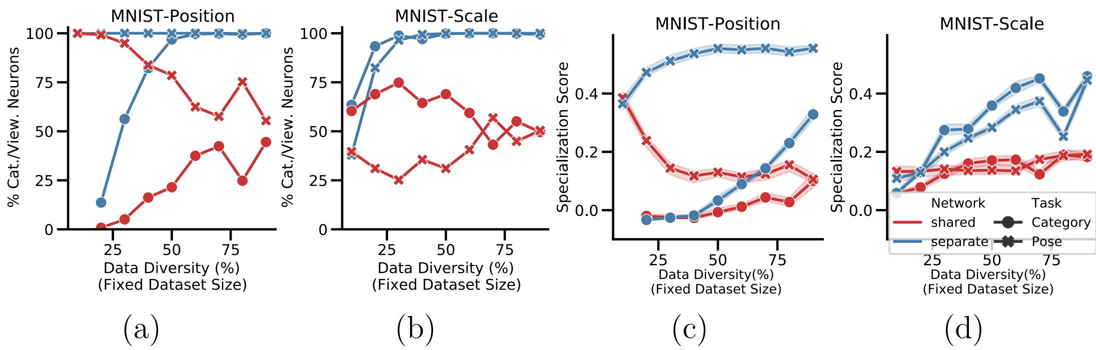


Figure Supp.11: Neuron specialization in MNIST-Position and MNIST-Scale datasets. (a) and (b) Percentage of neurons in the final convolutional layer of ResNet-18 that are specialized to category and viewpoint, for MNIST-Position and MNIST-Scale datasets, respectively. (c) and (d) Median of the specialization scores of neurons in the final convolutional layer of ResNet-18 Separate and Shared architectures, for category and viewpoint classification tasks, respectively.

Similarly, when accuracy for viewpoint classification is higher, a greater fraction of neurons becomes specialized for viewpoint, as observed in MNIST-Position.

S5.2 Specialization score for additional datasets

Figs. Supp.11c and d show that as the number of in-distribution combinations are increased, there is a steady increase in the specialization score for both MNIST-Position and MNIST-Scale. In Fig. Supp.12, we show that the selectivity score results are also consistent in iLab for different backbones and split architectures.

S5.3 Invariance and Selectivity Scores

In Fig. Supp.13 and Supp.14, we show the invariance and selectivity scores separately for the Biased-Cars dataset. In both cases, the trends follow what we observed for the specialization score, though the differences are much more pronounced in terms of invariance rather than selectivity.

S5.4 Specialization Score per Layer

In Fig. Supp.15, we show the specialization score in each layer. We can see that it builds up across layers, and this is more pronounced for Separate architectures than for Shared.

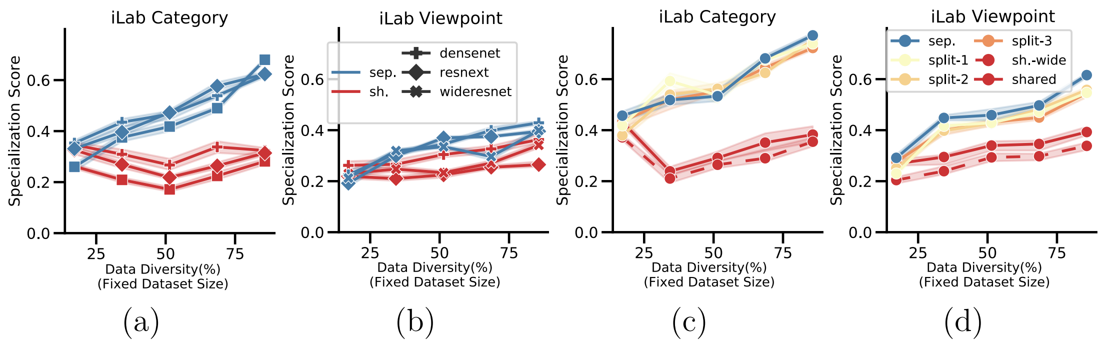


Figure Supp.12: Neuron specialization (selectivity to category and invariance to viewpoint, and vice versa) in the iLab dataset. (a) and (b) Median of the specialization score among neurons (Γ^k) in network architectures, other than ResNet-18, separate and shared, for category and viewpoint classification tasks, respectively. Confidence intervals (95%) are displayed in low opacity. (c) and (d) Median of the specialization score among neurons in ResNet-18 Separate and Shared with splits made at different blocks of the network, for category and viewpoint classification tasks, respectively. Similar results for the Biased-Cars dataset are provided in the main paper.

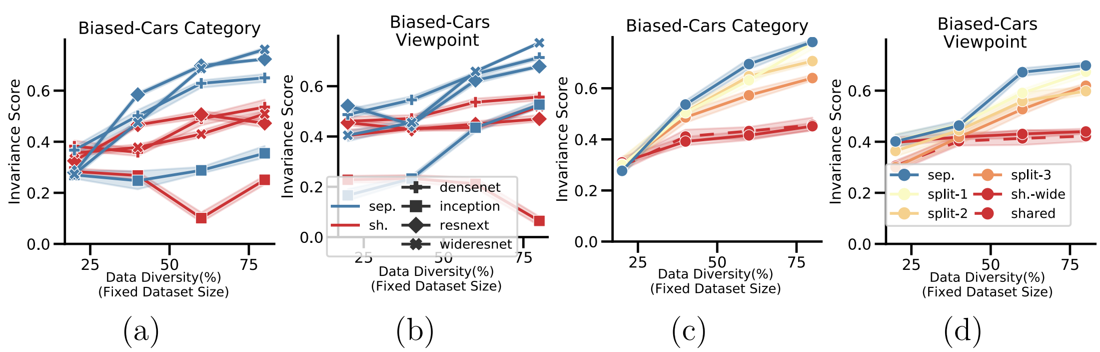
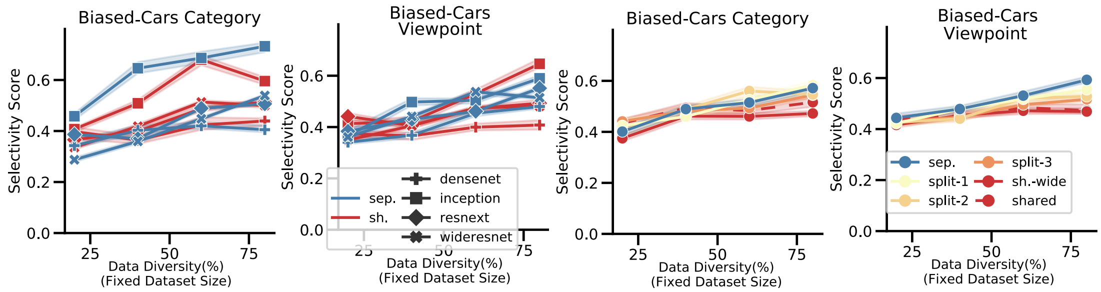


Figure Supp.13: Invariance scores in the Biased-Cars dataset. (a) and (b) Median of the invariance score among neurons in network architectures, other than ResNet-18, separate and shared, for category and viewpoint recognition tasks, respectively. Confidence intervals (95%) are displayed in low opacity. (c) and (d) Median of the invariance score among neurons in ResNet-18 Separate and Shared with splits made at different blocks of the network, for category and viewpoint recognition tasks, respectively.

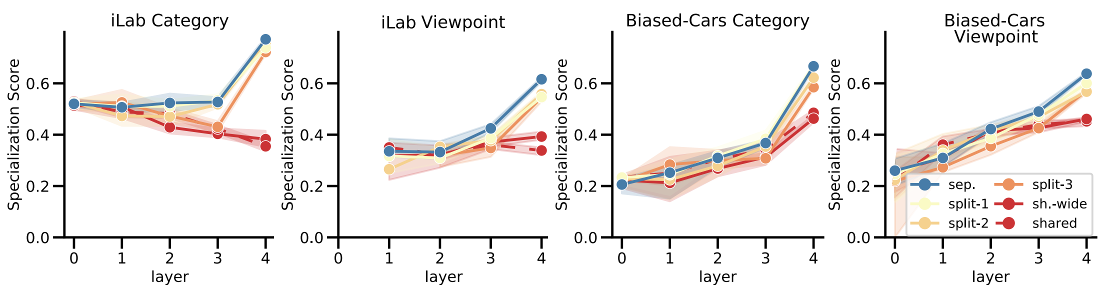
S6 Limitations

In this paper we have only considered rigid objects, while general object recognition often involves deformable and articulated object categories including humans and other animals.For such objects, parts may appear in various configurations for the same viewpoint. One way to analyze this more complex scenario would be to extend



(a) (b) (c) (d)

Figure Supp.14: Selectivity scores in the Biased-Cars dataset. (a) and (b) Median of the selectivity score among neurons in network architectures, other than ResNet-18, separate and shared, for category and viewpoint recognition tasks, respectively. Confidence intervals (95%) are displayed in low opacity. (c) and (d) Median of the selectivity score among neurons in ResNet-18 Separate and Shared with splits made at different blocks of the network, for category and viewpoint recognition tasks, respectively.



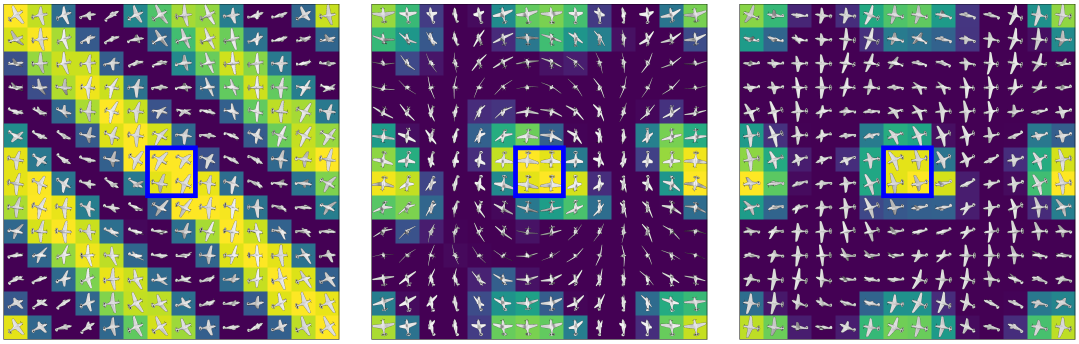
(a) (b) (c) (d)

Figure Supp.15: Specialization Score Per Layer for 30 seen category-viewpoint Combinations for iLab, and 20 seen category-viewpoint Combinations for the Biased-Cars dataset. (a) and (b) Median of the specialization score among neurons in ResNet-18 Separate and Shared with splits made at different blocks of the network, for category and viewpoint classification tasks, respectively. (c) and (d) Same as (a) and (b) for Biased-Cars dataset.

our experiments to study combinations of configurations, viewpoints and categories. Furthermore, this analysis may also be extended to study the impact of object symmetries, which would alter the effective number of visually distinct object viewpoints.

Also, we have considered selectivity and invariance of individual neurons as a model for understanding generalization to OOD combinations. This model is limited in several ways as it only considers the properties of individual neurons, and assumes that selectivity to one single category (or viewpoint) is needed alongside invariance to viewpoint (or category) to achieve generalization. There could be other ways to achieve generalization not taken into account by the model. Also, the evidence presented here is correlational and based on the average neural activity for a set of images. Nonetheless, the model has been shown to be useful to explain in simple and intuitive terms why the Separate architecture outperforms the Shared one, and how these generalize as more category-viewpoint combinations are seen.

a



b

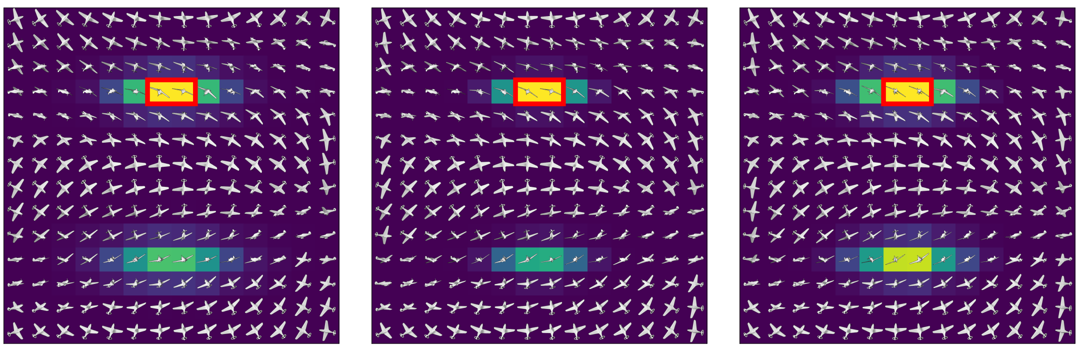


Figure Supp.16: Accuracy heatmaps: alternative 'seed' orientations. (a) 'Seed' orientations include $-0.25 \leq \alpha \leq 0.25, -0.25 \leq \gamma \leq 0.25, -1/2\pi \leq \beta < 1/2\pi$. (b) 'Seed' orientations include $-0.1 \leq \beta \leq 0.1, -0.25 \leq \gamma \leq 0.25, -1.8\pi \leq \alpha < -1.3/\pi$.

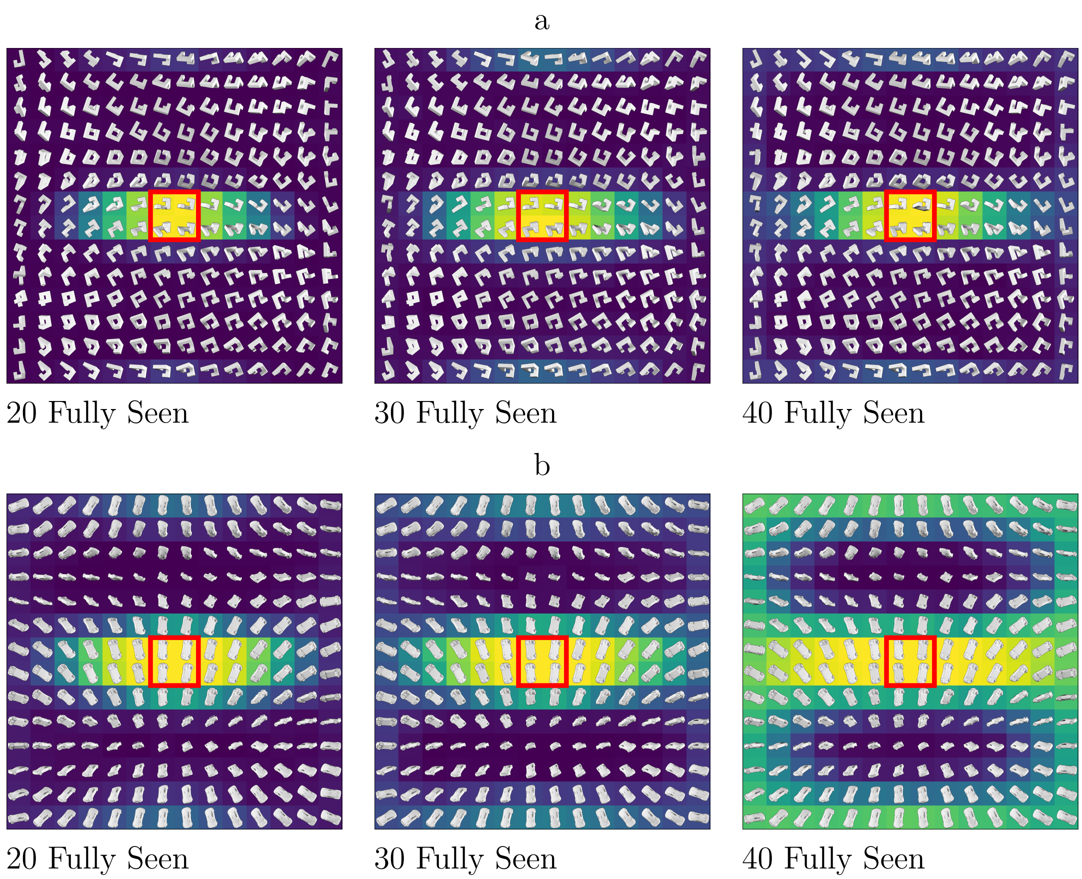
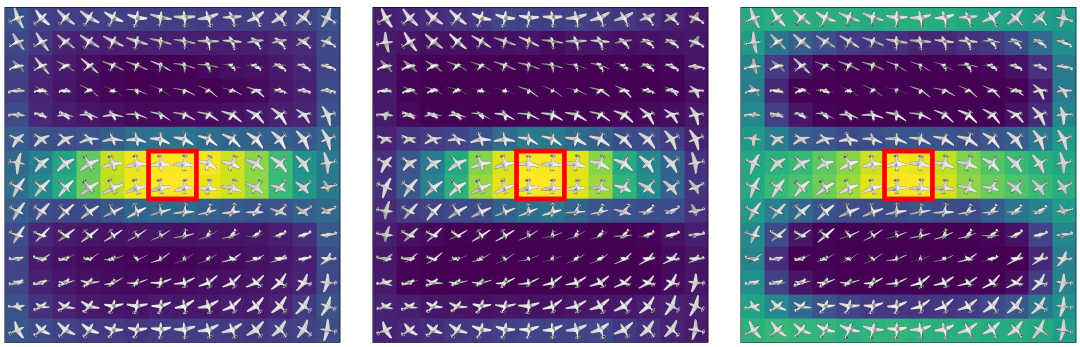


Figure Supp.17: Accuracy heatmaps: effect of data diversity - alternative object categories. Increasing number of fully-seen instances, with different object classes. (a) Shepard-Metzler Objects. (b) Cars.

a



b

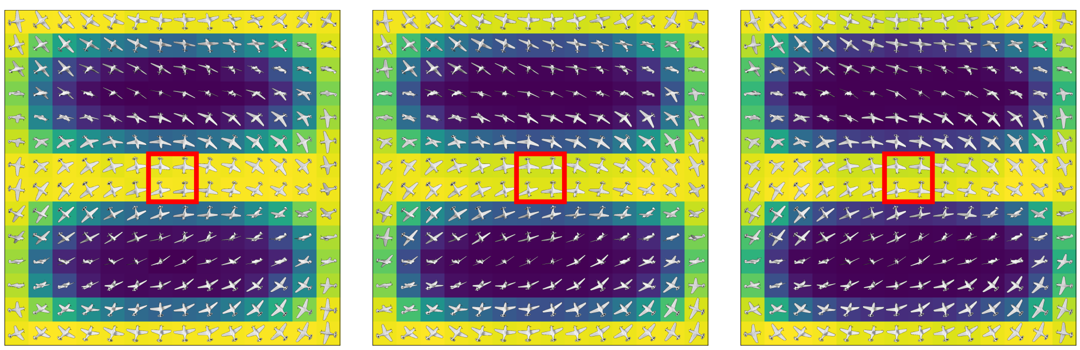


Figure Supp.18: Accuracy heatmaps: alternative training conditions - pretraining and augmentation. (a) ResNet-18 pretrained on ImageNet[314], finetuned on our learning paradigm with airplanes. Network behavior isn't meaningfully altered. (b) All data (both from fully-seen and partially-seen instances) were augmented with random 2D image rotations. This effectively expands the in-distribution set to include all generalizable orientations. This results in generalizable orientations with high accuracy.

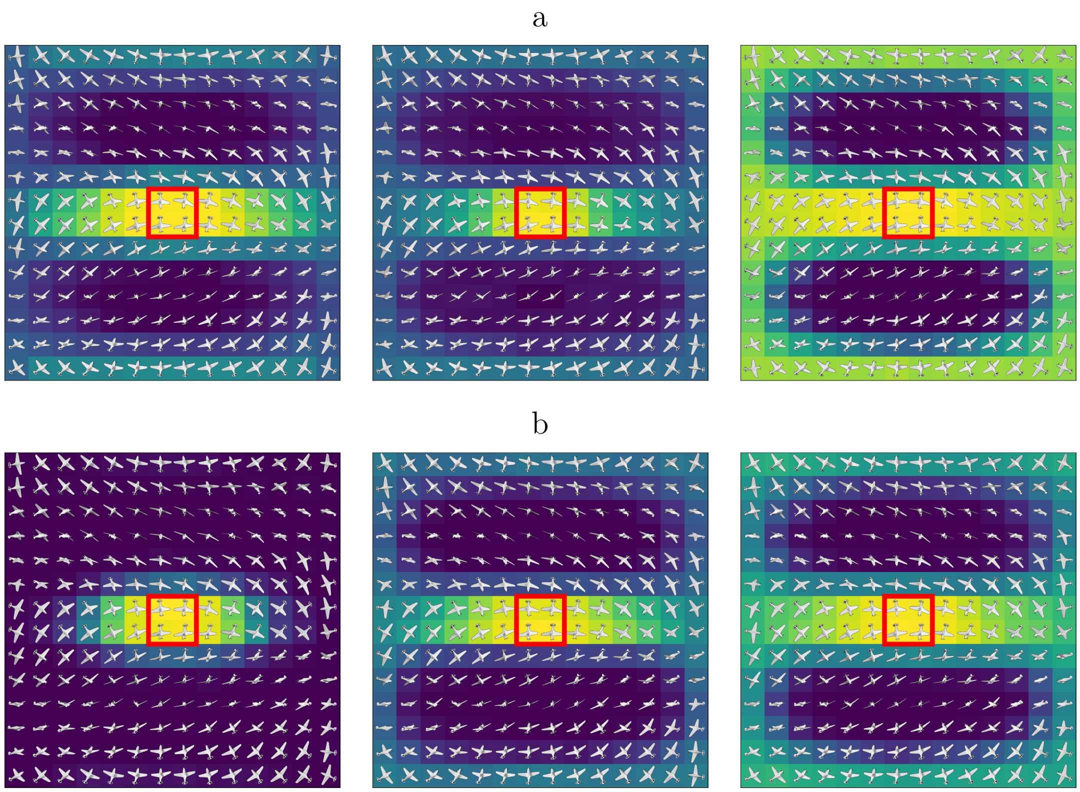


Figure Supp.19: Accuracy heatmaps: alternative backbone architectures. Network's backbone used (in place of ResNet-18): (a) DenseNet. (b) CORnet.

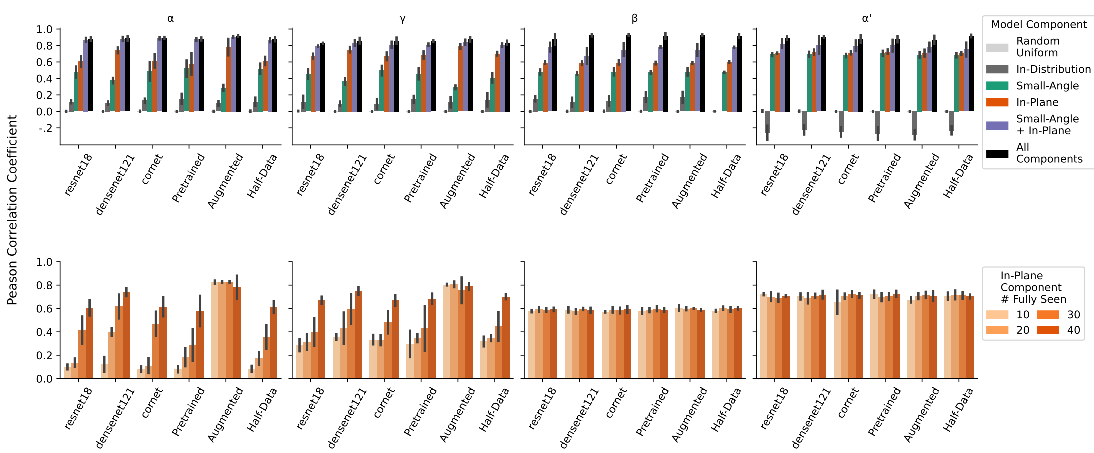


Figure Supp.20: Modeling generalization patterns for OoD orientations, continued. The same analysis as Figs. 4.3b is applied to the controls introduced in Figs. Supp.18, Supp.19.

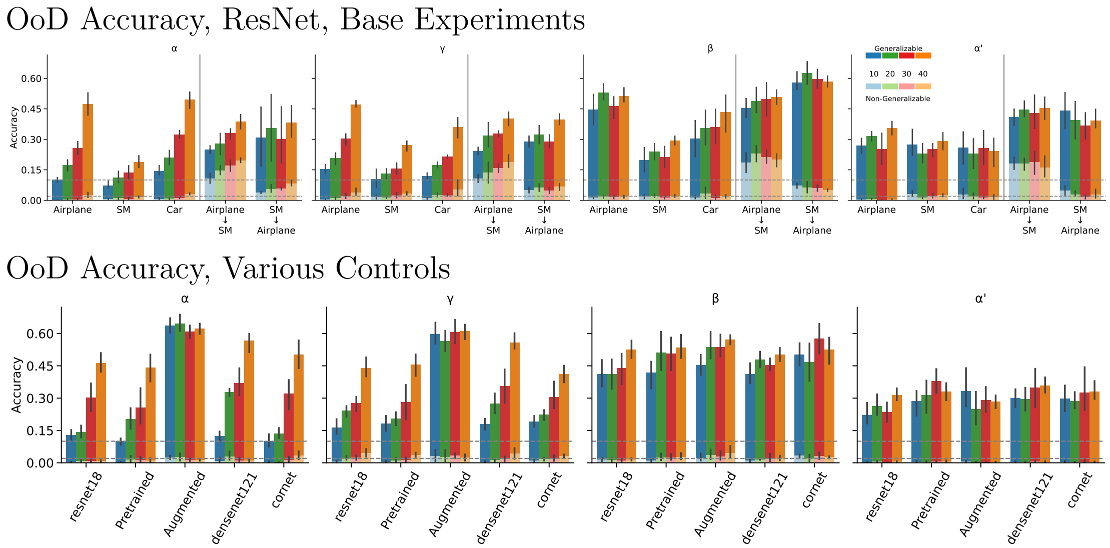


Figure Supp.21: OoD accuracy, split between generalizable and non-generalizable orientations In Fig. 4.3a we report the average accuracy across all OoD orientations. As we note, however, accuracy behavior is differentiated between generalizable and non-generalizable orientations. Here we report the average accuracy for these two orientation groups. Gray horizontal lines indicate chance performance of 2% and 10% (the latter relevant in the case where fully-seen and partially-seen instances are of two different classes.) Generalizable accuracy is always greater than non-generalizable accuracy. The former is always well above chance, while the latter is below or at chance level. (a) The generalizable and non-generalizable average accuracy for the same set of experiments presented in Fig. 4.3a. (b) The average accuracies for several other conditions. These other conditions are explained in Figs. Supp.18, Supp.19.

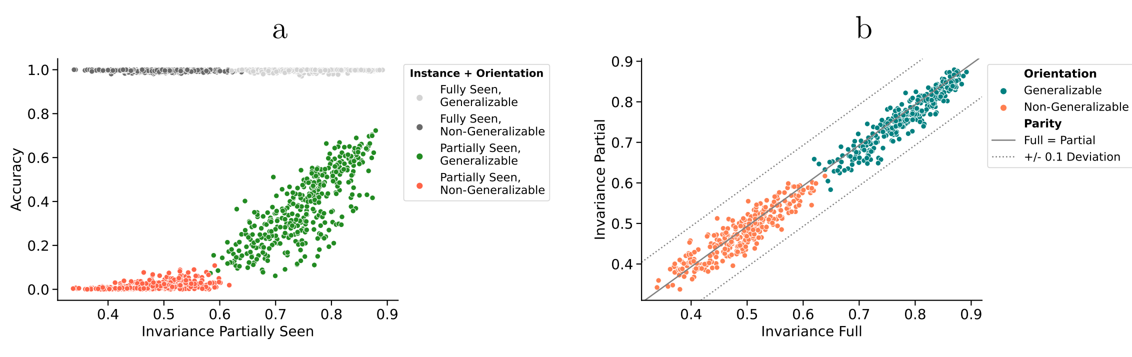


Figure Supp.22: Invariance and Dissemination: controls. The same analysis as Figs. 4.4b, c is applied to the controls introduced in Figs. Supp.18, Supp.19.

S7 Variation-base decomposition, invariance and selectivity scores

S7.1 Variation-base decomposition

We need α_{ij}^{nm}, the averaged activity of a neuron over an object category i and an nuisance attribute category j, to obtain the invariance score and selectivity score introduced by Madan et al.[241]. The set $\{\alpha_{ij}^{nm}\}_{i\in\mathcal{I},j\in\mathcal{J}}$ is called the activations grid. In this appendix, we first overview how to calculate α_{ij}^{nm} from $a^{nm}(\mathbf{x}) \geq 0$, the activity of a neuron corresponding to an image $\mathbf{x}$. Fig. Supp.1 shows overview of calculation of activations grid. Then we introduce the variation-base decomposition that enables us to describe the activity $a^{nm}(\mathbf{x})$ by the variation activity $\tilde{a}^{nm}(\mathbf{x})$, scale factor ρ^{nm}, and base activity b^{nm}. This decomposition simplifies the formula regarding α_{ij}^{nm}, and consequently, the ones regarding the invariance and selectivity scores.

In the calculation process of the activations grid $\{\alpha_{ij}^{nm}\}_{i\in\mathcal{I},j\in\mathcal{J}}$, we first normalize the activity $a^{nm}(\mathbf{x})$ by the maximum activity over the population of images $\mathcal{X}$ as

$$\frac{a^{nm}(\mathbf{x})}{\max_{\mathbf{x}\in\mathcal{X}} a^{nm}(\mathbf{x})}. \tag{Supp.7}$$

This value is then averaged over $\mathcal{X}_{ij} = \{\mathbf{x} \in \mathcal{X} \mid \mathbf{y} = (i,j)\}$, the set of all images belonging to the object category i and nuisance attribute category j, as follows:

$$\operatorname*{average}_{\mathbf{x}\in\mathcal{X}_{ij}} \left(\frac{a^{nm}(\mathbf{x})}{\max_{\mathbf{x}\in\mathcal{X}} a^{nm}(\mathbf{x})} \right) = \frac{\operatorname*{average}_{\mathbf{x}\in\mathcal{X}_{ij}} a^{nm}(\mathbf{x})}{\max_{\mathbf{x}\in\mathcal{X}} a^{nm}(\mathbf{x})}. \tag{Supp.8}$$

Then we regard its minimum value as the baseline of activity and subtract it from the right hand side of Eq. Supp.8:

$$\frac{\operatorname*{average}_{\mathbf{x}\in\mathcal{X}_{ij}} a^{nm}(\mathbf{x})}{\max_{\mathbf{x}\in\mathcal{X}} a^{nm}(\mathbf{x})} - \min_{\mu,\nu} \frac{\operatorname*{average}_{\mathbf{x}\in\mathcal{X}_{\mu\nu}} a^{nm}(\mathbf{x})}{\max_{\mathbf{x}\in\mathcal{X}} a^{nm}(\mathbf{x})}. \tag{Supp.9}$$

Consider the situation that $\max_{\mu,\nu} \frac{\operatorname*{average}_{\mathbf{x}\in\mathcal{X}_{\mu\nu}} a^{nm}(\mathbf{x})}{\max_{\mathbf{x}\in\mathcal{X}} a^{nm}(\mathbf{x})} - \min_{\mu,\nu} \frac{\operatorname*{average}_{\mathbf{x}\in\mathcal{X}_{\mu\nu}} a^{nm}(\mathbf{x})}{\max_{\mathbf{x}\in\mathcal{X}} a^{nm}(\mathbf{x})} = 0$ holds. This means $\operatorname*{average}_{\mathbf{x}\in\mathcal{X}_{\mu\nu}} a^{nm}(\mathbf{x})$ takes the same value for all combinations of object and nuisance attribute categories. We call such a neuron degenerated.

For non-degenerated neurons, we normalize Eq. Supp.9 so that it ranges from 0 to 1 and define the result as α_{ij}^{nm}:

$$\alpha_{ij}^{nm} = \frac{\frac{\underset{\mathbf{x}\in\mathcal{X}_{ij}}{\text{average}}\ a^{nm}(\mathbf{x})}{\max_{\mathbf{x}\in\mathcal{X}} a^{nm}(\mathbf{x})} - \min_{\mu,\nu} \frac{\underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x})}{\max_{\mathbf{x}\in\mathcal{X}} a^{nm}(\mathbf{x})}}{\max_{\mu,\nu} \frac{\underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x})}{\max_{\mathbf{x}\in\mathcal{X}} a^{nm}(\mathbf{x})} - \min_{\mu,\nu} \frac{\underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x})}{\max_{\mathbf{x}\in\mathcal{X}} a^{nm}(\mathbf{x})}}. \tag{Supp.10}$$

We reduce Eq. Supp.10 to

$$\alpha_{ij}^{nm} = \frac{\underset{\mathbf{x}\in\mathcal{X}_{ij}}{\text{average}}\ a^{nm}(\mathbf{x}) - \min_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x})}{\max_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x}) - \min_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x})}. \tag{Supp.11}$$

This is the original definition of entries of activation grid α_{ij}^{nm}.

Next, we define a decomposition of activity to the variation component and base component to express α_{ij}^{nm} in simple formula. Firstly, we divide Eq. Supp.11 into two parts as follows:

$$\alpha_{ij}^{nm} = \frac{\underset{\mathbf{x}\in\mathcal{X}_{ij}}{\text{average}}\ a^{nm}(\mathbf{x}) - \min_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x})}{\max_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x}) - \min_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x})} \tag{Supp.12}$$

$$= \frac{\underset{\mathbf{x}\in\mathcal{X}_{ij}}{\text{average}}\ a^{nm}(\mathbf{x})}{\max_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x}) - \min_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x})} - \frac{\min_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x})}{\max_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x}) - \min_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x})}. \tag{Supp.13}$$

We define the scale factor ρ^{nm} as

$$\rho^{nm} = \max_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x}) - \min_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x}). \tag{Supp.14}$$

Then we obtain

$$\alpha_{ij}^{nm} = \frac{\underset{\mathbf{x}\in\mathcal{X}_{ij}}{\text{average}}\ a^{nm}(\mathbf{x})}{\rho^{nm}} - \frac{\min_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x})}{\rho^{nm}}. \tag{Supp.15}$$

We define b^{nm} as

$$b^{nm} = \min_{\mu,\nu} \underset{\mathbf{x}\in\mathcal{X}_{\mu\nu}}{\text{average}}\ a^{nm}(\mathbf{x}), \tag{Supp.16}$$

and call it the base activation. Using ρ^{nm} and b^{nm}, α_{ij}^{nm} is expressed as

$$\alpha_{ij}^{nm} = \underset{\mathbf{x}\in\mathcal{X}_{ij}}{\text{average}} \, (a^{nm}(\mathbf{x})/\rho^{nm} - b^{nm}/\rho^{nm}). \tag{Supp.17}$$

Let us define $\widetilde{a}^{nm}(\mathbf{x})$ by the following equation:

$$\widetilde{a}^{nm}(\mathbf{x}) = a^{nm}(\mathbf{x})/\rho^{nm} - b^{nm}/\rho^{nm}. \tag{Supp.18}$$

Now $a^{nm}(\mathbf{x})$ can be expressed as follows:

$$a^{nm}(\mathbf{x}) = \rho^{nm}\widetilde{a}^{nm}(\mathbf{x}) + b^{nm}. \tag{Supp.19}$$

We call Eq. Supp.19 as the variation-base decomposition of $a^{nm}(\mathbf{x})$. Using these definitions, we simplify Eq. Supp.11 as follows:

$$\alpha_{ij}^{nm} = \underset{\mathbf{x}\in\mathcal{X}_{ij}}{\text{average}} \, \widetilde{a}^{nm}(\mathbf{x}). \tag{Supp.20}$$

Note that we need information on population of images to calculate specific values of $\widetilde{a}^{nm}(\mathbf{x})$ and b^{nm}. In practice, we use all images in the dataset to calculate actual values.

Regarding a layer, we denote the activity of neurons in layer n as $\boldsymbol{a}^n(\mathbf{x}) = [a^{n1}(\mathbf{x}), \ldots, a^{nM_n}(\mathbf{x})]^\top$, where M_n denotes the number of neurons in a layer n. If there are no degenerated neurons in this layer, we obtain the variation-base decomposition of $\boldsymbol{a}^n(\mathbf{x})$ based on Eq. Supp.19 as follows:

$$\boldsymbol{a}^n(\mathbf{x}) = \rho^n \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}) + \boldsymbol{b}^{nm}, \tag{Supp.21}$$

where $\rho^n = [\rho^{n1}, \ldots, \rho^{nM_n}]^\top$, $\widetilde{\boldsymbol{a}}^n(\mathbf{x}) = [\widetilde{a}^{n1}(\mathbf{x}), \ldots, \widetilde{a}^{nM_n}(\mathbf{x})]^\top$, and $\boldsymbol{b}^n = [b^{n1}, \ldots, b^{nM_n}]^\top$.

S7.2 Invariance score and selectivity score

The invariance score and selectivity score introduced in[241] have been shown effective at analysing generalization beyond data bias. Because we need $\{\alpha_{ij}^{nm}\}_{i\in\mathcal{I},j\in\mathcal{J}}$ to calculate the invariance and selectivity scores, we continue focusing on non-degenerated neurons. To calculate these scores, we first identify the object category that a neuron is most active on average, i.e., $i^{*nm} = \text{argmax}_i \sum_j \alpha_{ij}^{nm}$. This is called the preferred object category.

The invariance score I^{nm} with respect to nuisance attribute category for the m-th neuron in the n-th layer is defined as

$$I^{nm} = 1 - (\max_j \alpha_{i^{*nm}j}^{nm} - \min_j \alpha_{i^{*nm}j}^{nm}). \tag{Supp.22}$$

It ranges from 0 to 1 and takes the maximum value in the case that the neuron outputs the same value for the preferred object category on average regardless of the nuisance attribute category.

The selectivity score S^{nm} with respect to the object category for the m-th neuron in the n-th layer is defined as

$$S^{nm} = \frac{\hat{\alpha}^{nm} - \bar{\alpha}^{nm}}{\hat{\alpha}^{nm} + \bar{\alpha}^{nm}}, \tag{Supp.23}$$

where $\hat{\alpha}^{nm}$ and $\bar{\alpha}^{nm}$ are the average of α_{ij}^{nm} over the preferred object category and the remaining object categories defined as follows:

$$\hat{\alpha}^{nm} = \frac{1}{\#(\mathcal{J})} \sum_j \alpha_{i^{*nm}j}^{nm}, \bar{\alpha}^{nm} = \frac{\sum_{i \neq i^{*nm}} \sum_j \alpha_{ij}^{nm}}{\#(\mathcal{J})(\#(\mathcal{I}) - 1)}. \tag{Supp.24}$$

In the experiments, we only treated the cases where $\#(\mathcal{J}) = \#(\mathcal{I})$. N is used to denote this number in the main body of our paper. This score also ranges from 0 to 1 and takes its maximum value in the case that the neuron only outputs a positive value for the preferred object category.

S7.3 Summary of definitions

Here we summarize the definitions and results introduced in Sec. S7.1 and Sec. S7.2. To simplify the descriptions, we refer the m-th neuron in the n-the layer as neuron (n, m).

Definition 1 (Degenerated neuron / non-degenerated neuron). Let $\mathbf{x}$ denote an image and $a^{nm}(\mathbf{x}) \geq 0$ denote the activity of a neuron (n, m) corresponding to $\mathbf{x}$. Let $\mathcal{X}_{ij}$ denote the set of all images belonging to the object category $i \in \mathcal{I}$ and nuisance attribute category $j \in \mathcal{J}$.
We call the neuron (n, m) degenerated if it satisfies

$$\max_{i,j} \operatorname*{average}_{\mathbf{x} \in \mathcal{X}_{ij}} a^{nm}(\mathbf{x}) = \min_{i,j} \operatorname*{average}_{\mathbf{x} \in \mathcal{X}_{ij}} a^{nm}(\mathbf{x}), \tag{Supp.25}$$

or equivalently

$$\operatorname*{average}_{\mathbf{x} \in \mathcal{X}_{ij}} a^{nm}(\mathbf{x}) = \text{ a common constant across all } (i, j) \in \mathcal{I} \times \mathcal{J}. \tag{Supp.26}$$

Otherwise the neuron is called non-degenerated. ◁

Definition 2 (Original definition of entries of activations grid). Let $\mathbf{x}$ denote an image and $a^{nm}(\mathbf{x}) \geq 0$ denote the activity of a non-degenerated neuron (n, m) corresponding

to $\mathbf{x}$. Let $\mathcal{X}_{ij}$ denote the set of all images belonging to the object category $i \in \mathcal{I}$ and nuisance attribute category $j \in \mathcal{J}$.
Each entry of activativations grid $\{\alpha_{ij}^{nm}\}_{i\in\mathcal{I},j\in\mathcal{J}}$ is defined as

$$\alpha_{ij}^{nm} = \frac{\operatorname*{average}_{\mathbf{x}\in\mathcal{X}_{ij}} a^{nm}(\mathbf{x}) - \min_{\mu,\nu} \operatorname*{average}_{\mathbf{x}\in\mathcal{X}_{\mu\nu}} a^{nm}(\mathbf{x})}{\max_{\mu,\nu} \operatorname*{average}_{\mathbf{x}\in\mathcal{X}_{\mu\nu}} a^{nm}(\mathbf{x}) - \min_{\mu,\nu} \operatorname*{average}_{\mathbf{x}\in\mathcal{X}_{\mu\nu}} a^{nm}(\mathbf{x})}. \tag{Supp.27}$$

$\triangleleft$

Remark. Note that α_{ij}^{nm} ranges from 0 to 1.

Definition 3 (Variation-base decomposition). Let $\mathbf{x}$ denote an image and $a^{nm}(\mathbf{x}) \geq 0$ denote the activity of a non-degenerated neuron (n, m) corresponding to $\mathbf{x}$. Let $\mathcal{X}_{ij}$ denote the set of all images belonging to the object category $i \in \mathcal{I}$ and nuisance attribute category $j \in \mathcal{J}$.
The base activation b^{nm} and the scale factor ρ^{nm} are defined as follows:

$$b^{nm} = \min_{i,j} \operatorname*{average}_{\mathbf{x}\in\mathcal{X}_{ij}} a^{nm}(\mathbf{x}), \tag{Supp.28}$$

$$\rho^{nm} = \max_{i,j} \operatorname*{average}_{\mathbf{x}\in\mathcal{X}_{ij}} a^{nm}(\mathbf{x}) - \min_{i,j} \operatorname*{average}_{\mathbf{x}\in\mathcal{X}_{ij}} a^{nm}(\mathbf{x}) > 0. \tag{Supp.29}$$

The variation of activity $\widetilde{a}^{nm}(\mathbf{x})$ is then defined as follows:

$$\widetilde{a}^{nm}(\mathbf{x}) = a^{nm}(\mathbf{x})/\rho^{nm} - b^{nm}/\rho^{nm}. \tag{Supp.30}$$

As a consequence, the activity $a^{nm}(\mathbf{x})$ is expressed by ρ^{nm}, $\widetilde{a}^{nm}(\mathbf{x})$, and $b^{nm}(\mathbf{x})$ as follows:

$$a^{nm}(\mathbf{x}) = \rho^{nm}\widetilde{a}^{nm}(\mathbf{x}) + b^{nm}. \tag{Supp.31}$$

We call Eq. Supp.31 the variation-base decomposition of $a^{nm}(\mathbf{x})$. $\triangleleft$

Remark. Let us denote the activity of neurons in a layer n as $\boldsymbol{a}^n(\mathbf{x}) = [a^{n1}(\mathbf{x}), \ldots, a^{nM_n}(\mathbf{x})]^\top$, where M_n denotes the number of neurons in this layer. If there are no degenerated neurons in this layer, $\boldsymbol{a}^n(\mathbf{x})$ is expressed as

$$\boldsymbol{a}^n(\mathbf{x}) = \rho^n \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}) + \boldsymbol{b}^{nm}, \tag{Supp.32}$$

where $\rho^n = [\rho^{n1}, \ldots, \rho^{nM_n}]^\top$, $\widetilde{\boldsymbol{a}}^n(\mathbf{x}) = [\widetilde{a}^{n1}(\mathbf{x}), \ldots, \widetilde{a}^{nM_n}(\mathbf{x})]^\top$, and $\boldsymbol{b}^n = [b^{n1}, \ldots, b^{nM_n}]^\top$.
We call Eq. Supp.32 the variation-base decomposition of $\boldsymbol{a}^n(\mathbf{x})$.

Corollary 4 (Entries of activations grid). Let $\mathbf{x}$ denote an image, $a^{nm}(\mathbf{x}) \geq 0$ denote the activity of a non-degenerated neuron (n, m) corresponding to $\mathbf{x}$, and $a^{nm}(\mathbf{x}) = \rho^{nm}\widetilde{a}^{nm}(\mathbf{x}) + b^{nm}$ be the variation-base decomposition of $a^{nm}(\mathbf{x})$. Let $\mathcal{X}_{ij}$ denote the set of all images belonging to object category $i \in \mathcal{I}$ and nuisance attribute category $j \in \mathcal{J}$. Each entry of activations grid $\{\alpha_{ij}^{nm}\}_{i\in\mathcal{I},j\in\mathcal{J}}$ is equal to the averaged variation of activity over $\mathcal{X}_{ij}$, that is, the following relationship holds:

$$\alpha_{ij}^{nm} = \underset{\mathbf{x}\in\mathcal{X}_{ij}}{\text{average}}\ \widetilde{a}^{nm}(\mathbf{x}). \tag{Supp.33}$$

Definition 5 (Preferred object category). Consider a non-degenerated neuron (n, m). Let $\{\alpha_{ij}^{nm}\}_{i\in\mathcal{I},j\in\mathcal{J}}$ be its activations grid. A preferred object category i^{*nm} is defined as

$$i^{*nm} = \text{argmax}_i \sum_j \alpha_{ij}^{nm}. \tag{Supp.34}$$

◁

Definition 6 (Invariance score). Consider a non-degenerated neuron (n, m). Let $\{\alpha_{ij}^{nm}\}_{i\in\mathcal{I},j\in\mathcal{J}}$ be its activations grid, and i^{*nm} be its preferred object category. The invariance score with respect to nuisance attribute category for the neuron (n, m) is defined as

$$I^{nm} = 1 - (\text{max}_j \alpha_{i^{*nm}j}^{nm} - \text{min}_j \alpha_{i^{*nm}j}^{nm}). \tag{Supp.35}$$

◁

Remark. Note that I^{nm} ranges from 0 to 1.

Definition 7 (Selectivity score). Consider a non-degenerated neuron (n, m). Let $\{\alpha_{ij}^{nm}\}_{i\in\mathcal{I},j\in\mathcal{J}}$ be its activations grid, and i^{*nm} be its preferred object category. The selectivity score S^{nm} with respect to object category for the neuron (n, m) is defined as

$$S^{nm} = \frac{\hat{\alpha}^{nm} - \bar{\alpha}^{nm}}{\hat{\alpha}^{nm} + \bar{\alpha}^{nm}}, \tag{Supp.36}$$

where $\hat{\alpha}^{nm}$ and $\bar{\alpha}^{nm}$ are the average of α_{ij}^{nm} over the preferred object category and the remaining object categories defined as follows:

$$\hat{\alpha}^{nm} = \frac{1}{\#(\mathcal{J})} \sum_j \alpha_{i^{*nm}j}^{nm}, \bar{\alpha}^{nm} = \frac{\sum_{i\neq i^{*nm}} \sum_j \alpha_{ij}^{nm}}{\#(\mathcal{J})(\#(\mathcal{I}) - 1)}. \tag{Supp.37}$$

When $\#(\mathcal{I}) = \#(\mathcal{J}) = N$, $\hat{\alpha}^{nm}$ and $\bar{\alpha}^{nm}$ become

$$\hat{\alpha}^{nm} = \frac{1}{N}\sum_j \alpha^{nm}_{i^{*nm}j}, \bar{\alpha}^{nm} = \frac{\sum_{i \neq i^{*nm}} \sum_j \alpha^{nm}_{ij}}{N(N-1)}. \tag{Supp.38}$$

◁

Remark. Note that S^{nm} ranges from 0 to 1.

S7.4 Properties of non-degenerated neurons

Here we show two properties of non-degenerated neurons as lemmas. We use them in the proof of theorems in Section S8.

Lemma 8. If the m-th neuron in the n-th layer is non-degenerated, there exist at least one image $\mathbf{x}$ belonging to its preferred object category i^{*nm} satisfying

$$\widetilde{a}^{nm}(\mathbf{x}) \neq 0. \tag{Supp.39}$$

Proof. We use the method of proof by contradiction. Assume that $\widetilde{a}^{nm}(\mathbf{x}) = 0$ holds for all $\mathbf{x} \in \mathcal{X}_{i^{*nm}}$. From the definition of i^{*nm} (Definition 5), we obtain

$$\alpha^{nm}_{ij} = 0 \text{ for all } i \text{ and } j. \tag{Supp.40}$$

Based on Definition 2, this leads to

$$\operatorname*{average}_{\mathbf{x} \in \mathcal{X}_{ij}} a^{nm}(\mathbf{x}) = \min_{\mu,\nu} \operatorname*{average}_{\mathbf{x} \in \mathcal{X}_{\mu\nu}} a^{nm}(\mathbf{x}) \text{ for all } i \text{ and } j. \tag{Supp.41}$$

Then

$$\max_{i,j} \operatorname*{average}_{\mathbf{x} \in \mathcal{X}_{ij}} a^{nm}(\mathbf{x}) = \min_{i,j} \operatorname*{average}_{\mathbf{x} \in \mathcal{X}_{ij}} a^{nm}(\mathbf{x}) \tag{Supp.42}$$

is satisfied, which means the neuron is degenerated. This is contradictory to non-degeneracy of the neuron. Therefore there exists at least one $\mathbf{x} \in \mathcal{X}_{i^{*nm}}$ satisfying $\widetilde{a}^{nm}(\mathbf{x}) \neq 0$. □

Lemma 9. If the m-th neuron in the n-th layer is non-degenerated, there exists at least one $j \in \mathcal{J}$ satisfying

$$\alpha^{nm}_{i^{*nm}j} > 0, \tag{Supp.43}$$

where i^{*nm} is the preferred object category of the neuron.

Proof. We use the method of proof by contradiction. From Corollary 4, the following equation holds:

$$\underset{\mathbf{x}\in\mathcal{X}_{i^*nm_j}}{\text{average}}\widetilde{\alpha}^{nm}_{i^*nm_j}(\mathbf{x}) = \alpha^{nm}_{i^*nm_j}. \tag{Supp.44}$$

If $\alpha^{nm}_{i^*nm_j} = 0$ holds for all j, we have following equation from the definition of the preferred object category in Definition 5:

$$\max_i \sum_j \alpha^{nm}_{ij} = 0. \tag{Supp.45}$$

From Definition 2, entries of the activation grid α^{nm}_{ij} ranges 0 to 1. Thus have

$$\sum_j \alpha^{nm}_{ij} = 0 \text{ for all } i \in \mathcal{I}. \tag{Supp.46}$$

As a consequence, we obtain

$$\alpha^{nm}_{ij} = 0 \text{ for all } i \text{ and } j. \tag{Supp.47}$$

This leads a contradiction in the same way as in the proof of Lemma 8 (see from Eq. Supp.40 to the end of proof). Therefore there exists at least one $j \in \mathcal{J}$ satisfying $\alpha^{nm}_{i^*nm_j} > 0$.

□

S8 Identical activity achieves the maximum value of invariance score and complementary activity achieves the maximum value of selectivity score

In this section we introduce identical activity and complementary activity, which are two types of neural activity of a layer. Then we provide the theorems showing that the identical activity achieves the maximum value of invariance score and the complementary activity achieves the maximum value of selectivity score in all neurons in the layer.

Definition 10 (Identical activity). Consider a layer n which consists of non-degenerated neurons, and the variation-base decomposition of the activity of neurons in this layer, $\boldsymbol{a}^n(\mathbf{x}) = \rho^n \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}) + \boldsymbol{b}^n$, given by Eq. Supp.32. Let $\mathcal{X}_i$ denote the set of all images belonging to an object category i and $\mathbf{x}, \mathbf{x}'$ denote images. We call the neural activity of layer n identical if the following condition holds for all $i \in \mathcal{I}$:

$$\widetilde{\boldsymbol{a}}^n(\mathbf{x}) = \widetilde{\boldsymbol{a}}^n(\mathbf{x}') \text{ for all } \mathbf{x} \in \mathcal{X}_i \text{ and } \mathbf{x}' \in \mathcal{X}_i. \tag{Supp.1}$$

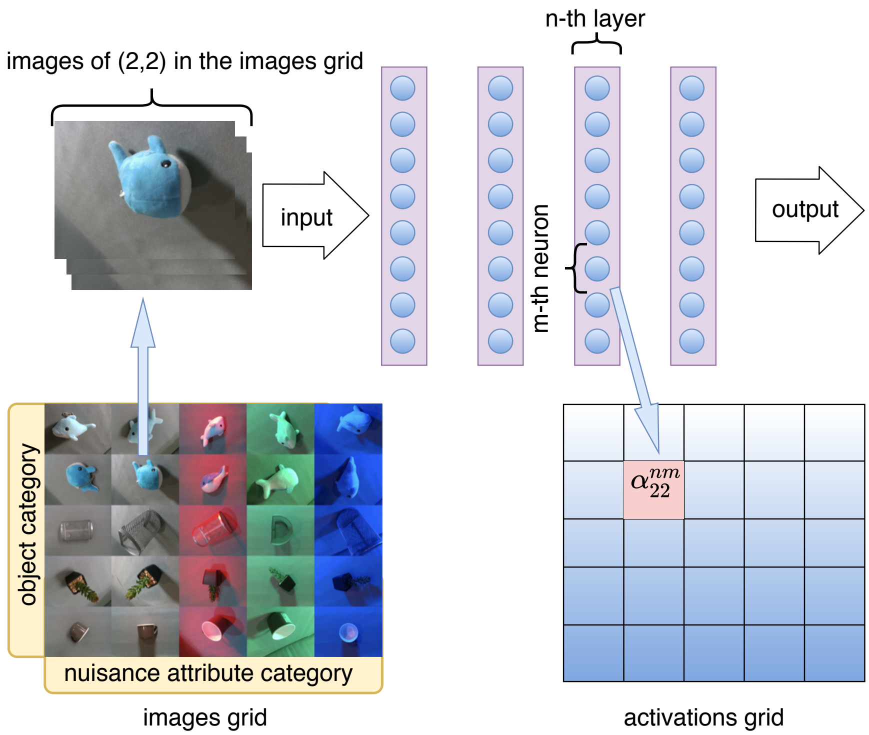


Figure Supp.1: Overview of calculation of activations grid. The images grid is a dataset consisting of all combinations of the object category and the nuisance attribute category. We extract all images belonging to one combination and calculate the α_{ij}^{nm} using their activity. We do this for all combinations and calculate the activations grid. Figure shows one example of calculation of the entry of activations grid α_{22}^{nm} using all images belonging to the combination of object category $i = 2$ and nuisance attribute category $j = 2$.

$\triangleleft$

Definition 11 (Complementary activity). Consider a layer n which consists of non-degenerated neurons, and the variation-base decomposition of the activity of neurons in this layer, $\boldsymbol{a}^n(\mathbf{x}) = \rho^n \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}) + \boldsymbol{b}^n$, given by Eq. Supp.32. Let $\mathcal{X}_i$ denote the set of all images belonging to an object category i and $\mathbf{x}, \mathbf{x}'$ denote images. We call the neural activity of layer n complementary if the following condition holds for all $i \in \mathcal{I}$:

$$\widetilde{\boldsymbol{a}}^n(\mathbf{x}) \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}') = \mathbf{0} \text{ for all } \mathbf{x} \in \mathcal{X}_i \text{ and } \mathbf{x}' \notin \mathcal{X}_i, \tag{Supp.2}$$

where $\odot$ denotes the element-wise product of vectors. $\triangleleft$

Theorem 12. Identical activity achieves the maximum value of the invariance score in all neurons.

Proof. Eq. Supp.1 holds for the preferred object category i^{*nm} in particular:

$$\widetilde{a}^{nm}(\mathbf{x}) = \widetilde{a}^{nm}(\mathbf{x}') \text{ for all } \mathbf{x} \in \mathcal{X}_{i^{*nm}} \text{ and } \mathbf{x}' \in \mathcal{X}_{i^{*nm}}. \tag{Supp.3}$$

We calculate entries of activations grid for every combination of (i^{*nm}, j) and (i^{*nm}, j'). Because $\mathcal{X}_{i^{*nm}j} \subset \mathcal{X}_{i^{*nm}}$ and $\mathcal{X}_{i^{*nm}j'} \subset \mathcal{X}_{i^{*nm}}$, the following equation also holds:

$$\widetilde{a}^{nm}(\mathbf{x}) = \widetilde{a}^{nm}(\mathbf{x}') \text{ for all } \mathbf{x} \in \mathcal{X}_{i^{*nm}j} \text{ and } \mathbf{x}' \in \mathcal{X}_{i^{*nm}j'}. \tag{Supp.4}$$

Therefore we have

$$\underset{\mathbf{x} \in \mathcal{X}_{i^{*nm}j}}{\text{average}} \widetilde{a}^{nm}(\mathbf{x}) = \underset{\mathbf{x}' \in \mathcal{X}_{i^{*nm}j'}}{\text{average}} \widetilde{a}^{nm}(\mathbf{x}'). \tag{Supp.5}$$

Due to Corollary 4, this is equivalent to

$$\alpha^{nm}_{i^{*nm}j} = \alpha^{nm}_{i^{*nm}j'}. \tag{Supp.6}$$

Eq. Supp.6 is satisfied for every combination of (i^{*nm}, j) and (i^{*nm}, j'). Hence,

$$\max_j \alpha^{nm}_{i^{*nm}j} = \min_j \alpha^{nm}_{i^{*nm}j}. \tag{Supp.7}$$

The above argument holds for all m. Thus the invariance score $I^{nm} = 1 - (\max_j \alpha^{nm}_{i^{*nm}j} - \min_j \alpha^{nm}_{i^{*nm}j})$ becomes 1 in all neurons in the layer n. $\square$

Theorem 13. Complementary activity achieves the maximum value of the selectivity score in all neurons.

Proof. Think about the complementarity condition Eq. Supp.2,

$$\widetilde{\boldsymbol{a}}^n(\mathbf{x}) \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}') = 0 \text{ for all } \mathbf{x} \in \mathcal{X}_i \text{ and } \mathbf{x}' \notin \mathcal{X}_i. \tag{Supp.8}$$

This means

$$\widetilde{a}^{nm}(\mathbf{x})\widetilde{a}^{nm}(\mathbf{x}') = 0 \text{ for all } \mathbf{x} \in \mathcal{X}_i \text{ and } \mathbf{x}' \notin \mathcal{X}_i \tag{Supp.9}$$

holds for every m in the layer n.
Firstly, we calculate $\bar{\alpha}^{nm}$ in Definition 7 (the definition of selectivity score). Due to Lemma 8, there exists at least one $\mathbf{x} \in \mathcal{X}_{i^{*nm}}$ satisfying $\widetilde{a}^{nm}(\mathbf{x}) \neq 0$. Therefore, we obtain from Eq. Supp.9 the following equation:

$$\widetilde{a}^{nm}(\mathbf{x}') = 0 \text{ for all } \mathbf{x}' \in \mathcal{X}_{i \neq i^{*nm}} (= \mathcal{X} \setminus \mathcal{X}_{i^{*nm}}). \tag{Supp.10}$$

From Corollary 4, entries of activations grid are calculated as

$$\alpha_{ij}^{nm} = 0 \text{ for all } i \neq i^{*nm}. \tag{Supp.11}$$

By taking the sum over all combinations of (i,j) with $i \neq i^{*nm}$, we have

$$\sum_{i \neq i^{*nm}} \sum_{j} \alpha_{ij}^{nm} = 0. \tag{Supp.12}$$

Dividing this by $\#(\mathcal{J})(\#(\mathcal{I}) - 1) = N(N-1)$, we obtain

$$\bar{\alpha}^{nm} = \frac{\sum_{i \neq i^{*nm}} \sum_{j} \alpha_{ij}^{nm}}{N(N-1)} = 0. \tag{Supp.13}$$

Next, we calculate $\hat{\alpha}^{nm}$ in Definition 7. Due to Lemma 9, there exists at least one $j \in \mathcal{J}$ satisfying $\alpha_{i^{*nm}j}^{nm} > 0$. Therefore we have

$$\hat{\alpha}^{nm} = \frac{1}{N} \sum_{j} \alpha_{i^{*nm}j}^{nm} > 0. \tag{Supp.14}$$

The selectivity score S^{mn} is calculated as

$$S^{nm} = \frac{\hat{\alpha}^{nm} - \bar{\alpha}^{nm}}{\hat{\alpha}^{nm} + \bar{\alpha}^{nm}} = \frac{\hat{\alpha}^{nm}}{\hat{\alpha}^{nm}} = 1. \tag{Supp.15}$$

The above argument holds for all m. In conclusion, complementary activity achieves $S^{nm} = 1$ for all neurons in the layer n.

□

Given the importance of complementary activity with regard to the selectivity score, we introduce the following score to measure how close a neural activity of a layer is to the complementary activity.

Definition 14 (Mean overlap of Activation (moA)). Consider a layer n which consists of non-degenerated neurons, and the variation-base decomposition of the activity of neurons in this layer, $\boldsymbol{a}^n(\mathbf{x}) = \rho^n \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}) + \boldsymbol{b}^n$, given by Eq. Supp.32. Let $\mathcal{X}_\mathrm{r}$ be a set of randomly sampled pair of images $(\mathbf{x}, \mathbf{x}')$ belonging to different object categories. The mean overlap of Activation (moA) is defined as

$$1 - \frac{1}{\#(\mathcal{X}_\mathrm{r})} \sum_{(\mathbf{x},\mathbf{x}') \in \mathcal{X}_\mathrm{r}} \frac{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}') \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0}}{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))}. \tag{Supp.16}$$

◁

Remark. This score takes a sufficiently large number of combinations of samples with alternating object categories i, and uses the overlap of the activation as the indicator of the complementarity. It ranges from 0 to 1, and the closer to 1, the better. In this study, we set $\#(\mathcal{X}_\mathrm{r}) = 500$, which we experimentally confirmed to be large enough for our purpose.

Remark. We replaced orthogonal activity defined by Eq. 5 in the main body of our paper by the complementary activity in Definition 11, because after the submission of the main body we found that the orthogonality is not sufficient to achieve the maximum value of the selectivity score but the complementarity is. Yet, the mean overlap of Activation (moA) works as a measure of the complementarity as it is, and the analysis in Sec. 4 in the main body is still valid.

S9 Invariance Enforcement Regularization (IER)

S9.1 Construction of the regularization term

In this section, we discuss how to construct a regularization term based on the identical activity and complementary activity. First, we think about the identical activity given by Definition 10. From Eq. Supp.30, we have

$$\widetilde{a}^{nm}(\mathbf{x}) - \widetilde{a}^{nm}(\mathbf{x}') = \frac{a^{nm}(\mathbf{x}) - b^{nm}}{\rho^{nm}} - \frac{a^{nm}(\mathbf{x}') - b^{nm}}{\rho^{nm}} \tag{Supp.1}$$

$$= \frac{a^{nm}(\mathbf{x}) - a^{nm}(\mathbf{x}')}{\rho^{nm}}. \tag{Supp.2}$$

If $a^{nm}(\mathbf{x}) - a^{nm}(\mathbf{x}') = 0$ is achieved, then Eq. Supp.2 becomes 0. If all neurons in the layer n achieve this, the neural activity of this layer becomes identical, which is a sufficient condition for the neurons to attain the maximum value of invariance score

(Theorem 12). From this analysis, we introduce the following regularization term:

$$
\begin{aligned}
R &= \|\boldsymbol{a}^n(\mathbf{x}) - \boldsymbol{a}^n(\mathbf{x}')\|_p &\text{(Supp.3)}\\
&= \|\dot{\boldsymbol{f}}^{\text{in}}(\mathbf{x}; \theta^{\text{in}}) - \dot{\boldsymbol{f}}^{\text{in}}(\hat{\mathbf{x}}; \theta^{\text{in}})\|_p, &\text{(Supp.4)}
\end{aligned}
$$

where $\|\cdot\|_p$ denotes the Lp norm. Based on the discussion above, adding this term to the original loss function (with an appropriate weight) is expected to enforce the invariance. We call the regularization technique using this term as Invariance Enforcement Regularization (IER).

Next we think about the complementary activity given by Definition 11, which is a sufficient condition for the neurons to attain the maximum value of selectivity score (Theorem 13). From Eq. Supp.30, we have

$$
\begin{aligned}
\widetilde{a}^{nm}(\mathbf{x})\widetilde{a}^{nm}(\mathbf{x}') &= \frac{a^{nm}(\mathbf{x}) - b^{nm}}{\rho^{nm}}\frac{a^{nm}(\mathbf{x}') - b^{nm}}{\rho^{nm}} &\text{(Supp.5)}\\
&= \frac{a^{nm}(\mathbf{x})a^{nm}(\mathbf{x}') - b^{nm}a^{nm}(\mathbf{x}) - b^{nm}a^{nm}(\mathbf{x}') + (b^{nm})^2}{\rho^{nm}}. &\text{(Supp.6)}
\end{aligned}
$$

Unfortunately we cannot calculate actual values of b^{nm} in the training phase, and thus it is difficult to derive a regularization term like Eq. Supp.4 directly from Eq. Supp.6.

Given that the complementary activity is difficult to use for constructing a regularization term, we consider another type of neural activity.

Definition 15 (Sparse activity). Consider a layer n which consists of non-degenerated neurons, and the variation-base decomposition of the activity of neurons in this layer, $\boldsymbol{a}^n(\mathbf{x}) = \rho^n \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}) + \boldsymbol{b}^n$, given by Eq. Supp.32. We call the neural activity in layer n sparse if the following condition holds:

$$
\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0} \ll \#(\widetilde{\boldsymbol{a}}^n(\mathbf{x})), \qquad \text{(Supp.7)}
$$

where $\#(\cdot)$ denotes the number of elements and $\#(\cdot)_{\neq 0}$ denotes the number of non-zero elements. $\triangleleft$

We show in the next section that if a neural activity is sparse, it is highly probable that the complementarity condition holds. While any norm in Eq. Supp.6 can work to enforce invariant neural activity in principle, different norms may work differently on sparseness, and thus differently on complementarity and selectivity. In this study, we investigate the effects of different norms on sparseness, complementarity and selectivity experimentally. For this purpose, we introduce another score to measure the sparseness.

Definition 16 (Mean rate of Activation (mrA)). Consider a layer n which consists of non-degenerated neurons, and the variation-base decomposition of the activity of neurons in this layer, $\boldsymbol{a}^n(\mathbf{x}) = \rho^n \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}) + \boldsymbol{b}^n$, given by Eq. Supp.32. Let $\mathcal{X}_i$ denotes the set of all images belonging to an object category i and $\mathbf{x}$ denote an image. The mean rate of Activation (mrA) is defined as

$$1 - \frac{1}{\#(\mathcal{I})} \frac{1}{\#(\mathcal{X}_i)} \sum_{i \in \mathcal{I}} \sum_{\mathbf{x} \in \mathcal{X}_i} \frac{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0}}{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))}. \tag{Supp.8}$$

◁

Remark. This score uses the ratio of activated neurons to all neurons as an indicator of the sparseness. It ranges from 0 to 1, and the closer to 1, the better.

S9.2 Relationship between sparse activity and complementary activity

If neural activity of a layer n is sparse, i.e., follows Eq. Supp.7, then $\#(\widetilde{\boldsymbol{a}}^{nm}(\mathbf{x}))_{\neq 0}$ is small and it is highly probable that Eq. Supp.9 (element-wise expression of complementary activity) is satisfied. To obtain a more concrete and quantitative result, we introduce the following definitions.

Definition 17 (Activation pattern). Consider a layer n which consists of non-degenerated neurons, and the variation-base decomposition of the activity of neurons in this layer, $\boldsymbol{a}^n(\mathbf{x}) = \rho^n \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}) + \boldsymbol{b}^n$, given by Eq. Supp.32. We define the activation pattern $\mathcal{P}^n_{\mathbf{x}}$ as follows:

$$\mathcal{P}^n_{\mathbf{x}} = \{m \mid \widetilde{a}^{nm}(\mathbf{x}) \neq 0\}. \tag{Supp.9}$$

That is, $\mathcal{P}^n_{\mathbf{x}}$ is the set of indices of neurons activated by the image $\mathbf{x}$. ◁

Remark. By definition, following relationship holds:

$$\#(\mathcal{P}^n_{\mathbf{x}} \cap \mathcal{P}^n_{\mathbf{x}'}) = \#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}') \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0}. \tag{Supp.10}$$

Definition 18 (Set of activation patterns). Consider a layer n which consists of non-degenerated neurons. Let $\mathcal{X}_i$ denote the set of all images belonging to an object category $i \in \mathcal{I}$. We define $\mathfrak{P}^n_i$, the set of activation patterns corresponding to the object category i, as follows:

$$\mathfrak{P}^n_i = \{\mathcal{P}^n_{\mathbf{x}} \mid \mathbf{x} \in \mathcal{X}_i\}. \tag{Supp.11}$$

◁

Definition 19 (Simple sparse activity). Consider a layer n which consists of non-degenerated neurons. We call the activity of the layer n simple sparse if it satisfies the following conditions.

1. Each activation pattern consists of exactly the same number of indices m. That is,

$$\#(\mathcal{P}^n_{\mathbf{x}}) = \#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0} = \text{ a common constant for all } \mathbf{x} \in \mathcal{X}. \tag{Supp.12}$$

2. The number of activation patterns corresponding to each object category is one. That is,

$$\#(\mathfrak{P}^n_i) = 1 \text{ for all } i \in \mathcal{I}. \tag{Supp.13}$$

3. Each activation pattern is determined randomly and independently in training phase.

◁

We have introduced the mean overlap of Activation (moA, Definition 14) as a quantitative score to measure the complementarity, and the mean rate of Activation (mrA, Definition 16) as a quantitative score to measure the sparseness. In the case of simple sparse activity, we can derive an explicit relationship of these scores.

Theorem 20 (Relationship between mrA and moA for simple sparse activity). In the case of simple sparse activity, the relationship between mrA and moA is given by the following equation:

$$\mathrm{E}[(\mathrm{moA})] = 2(\mathrm{mrA}) - (\mathrm{mrA})^2, \tag{Supp.14}$$

where $\mathrm{E}[\cdot]$ denotes the expected value.

Remark. Eq. Supp.14 means that the expected value of moA increases monotonically according to mrA because mrA ranges from 0 to 1.

Proof. Let $\mathcal{X}_i$ denote the set of all images belonging to an object category $i \in \mathcal{I}$. The condition 1 in Definition 19 holds for $\mathbf{x} \in \mathcal{X}_i$ in particular. Note that we can choose an arbitrary $i \in \mathcal{I}$. Note also that $\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x})) = M_n$ (the number of all neurons in layer n) regardless of $\mathbf{x}$. Therefore the following relation holds:

$$\frac{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0}}{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))} = \text{ a common constant for all } \mathbf{x} \in \mathcal{X}_i \text{ and } i \in \mathcal{I}. \tag{Supp.15}$$

We set this constant as τ. Then we can express mrA using τ as follows:

$$(\text{mrA}) = 1 - \frac{1}{\#(\mathcal{I})}\frac{1}{\#(\mathcal{X}_i)}\sum_{i\in\mathcal{I}}\sum_{\mathbf{x}\in\mathcal{X}_i}\frac{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0}}{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))} \tag{Supp.16}$$

$$= 1 - \frac{1}{\#(\mathcal{I})}\frac{1}{\#(\mathcal{X}_i)}\sum_{i\in\mathcal{I}}\sum_{\mathbf{x}\in\mathcal{X}_i}\tau \tag{Supp.17}$$

$$= 1 - \tau. \tag{Supp.18}$$

Conversely, τ can be expressed as

$$\tau = 1 - (\text{mrA}). \tag{Supp.19}$$

Think about pairs of object categories different to each other: $(i, i'), i \neq i'$. From the conditions 2 and 3 in Definition 19, each category has only one activation pattern and each pattern is determined randomly and independently in training phase. The expected value of overlap rate of activated neurons of the patterns of $\mathcal{P}_{\mathbf{x}}$ for all $\mathbf{x} \in \mathcal{X}_i$ and $\mathcal{P}_{\mathbf{x}'}$ for all $\mathbf{x}' \in \mathcal{X}_{i'}$ is τ^2. Therefore expected value of the number of overlapped activated neurons is $M_n\tau^2$, that is,

$$\mathrm{E}[\#(\mathcal{P}^n_{\mathbf{x}} \cap \mathcal{P}^n_{\mathbf{x}'})] = M_n\tau^2. \tag{Supp.20}$$

Due to Eq. Supp.10, this is equivalent to the following equation:

$$\mathrm{E}[\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}') \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0}] = M_n\tau^2. \tag{Supp.21}$$

Recall the definition of moA (Definition 14):

$$(\text{moA}) = 1 - \frac{1}{\#(\mathcal{X}_\mathrm{r})}\sum_{(\mathbf{x},\mathbf{x}')\in\mathcal{X}_\mathrm{r}}\frac{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}') \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0}}{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))}. \tag{Supp.22}$$

By taking the expected value of both side, we have

$$\mathrm{E}[(\text{moA})] = \mathrm{E}\left[1 - \frac{1}{\#(\mathcal{X}_\mathrm{r})}\sum_{(\mathbf{x},\mathbf{x}')\in\mathcal{X}_\mathrm{r}}\frac{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}') \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0}}{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))}\right] \tag{Supp.23}$$

$$= 1 - \frac{1}{\#(\mathcal{X}_\mathrm{r})}\sum_{(\mathbf{x},\mathbf{x}')\in\mathcal{X}_\mathrm{r}}\frac{\mathrm{E}[\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}') \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0}]}{\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}))} \tag{Supp.24}$$

$$= 1 - \frac{1}{\#(\mathcal{X}_\mathrm{r})}\sum_{(\mathbf{x},\mathbf{x}')\in\mathcal{X}_\mathrm{r}}\frac{\mathrm{E}[\#(\widetilde{\boldsymbol{a}}^n(\mathbf{x}') \odot \widetilde{\boldsymbol{a}}^n(\mathbf{x}))_{\neq 0}]}{M_n} \tag{Supp.25}$$

From Eq. Supp.21, we obtain expected value of moA as follows:

$$\mathrm{E}[(\mathrm{moA})] = 1 - \tau^2. \tag{Supp.26}$$

By substituting Eq. Supp.19 into Eq. Supp.26, we obtain the conclusion:

$$\mathrm{E}[(\mathrm{moA})] = 1 - (1 - (\mathrm{mrA}))^2 \tag{Supp.27}$$
$$= 2(\mathrm{mrA}) - (\mathrm{mrA})^2. \tag{Supp.28}$$

$\square$

S10 Details of implementation

S10.1 Learning algorithm with Invariance Enforcement Regularization (IER)

Algorithm 3 represents a learning algorithm with IER, where $\mathrm{y}^{(k)}$ be the category label corresponding to an image $\mathbf{x}^{(k)}$. In function $\mathrm{T}(\mathcal{D})$, we create a new mini-batch $\{(\hat{\mathbf{x}}^{(k)}, \hat{\mathrm{y}}^{(k)})\}_{k=1}^{B}$ for the regularization term apart from the original training mini-batch $\{(\mathbf{x}^{(k)}, \mathrm{y}^{(k)})\}_{k=1}^{B}$. In this process, we sample $(\mathbf{x}^{(k)}, \mathrm{y}^{(k)})$ randomly, and find corresponding $(\hat{\mathbf{x}}^{(k)}, \hat{\mathrm{y}}^{(k)})$ with $\hat{\mathrm{y}}^{(k)} = \mathrm{y}^{(k)}$. Note that the number of samples belonging to each set $\mathcal{X}_i$ are randomly determined. Look at row 7 in Algorithm 3. We calculate two forward paths, but the process corresponding to $\{(\hat{\mathbf{x}}^{(k)}, \hat{\mathrm{y}}^{(k)})\}_{k=1}^{B}$ is just used for calculation of R and we used gradients calculated only with $\{(\mathbf{x}^{(k)}, \mathrm{y}^{(k)})\}_{k=1}^{B}$, which is corresponding to row 6. In our implementation we employed Cross-Entropy loss for Loss function in row 6.

Algorithm 3 Learning algorithm with IER

Require:Training data $\mathcal{D}^{(\text{train})}$, Learning rate η, Regularization weight γ, Epochsize, Batchsize B, Norm $\|\cdot\|_p$
Ensure:Network parameters $\theta = (\theta^{\text{in}}, \theta^{\text{out}})$
while Epochsize do
    $\{(\mathbf{x}^{(k)}, \mathrm{y}^{(k)})\}_{k=1}^{B}, \{(\hat{\mathbf{x}}^{(k)}, \hat{\mathrm{y}}^{(k)})\}_{k=1}^{B} \leftarrow \mathrm{T}(\mathcal{D}^{(\text{train})})$
    $R \leftarrow \frac{1}{B}\sum_k \|\boldsymbol{f}^{\text{in}}(\mathbf{x}^{(k)};\theta^{\text{in}}) - \boldsymbol{f}^{\text{in}}(\hat{\mathbf{x}}^{(k)};\theta^{\text{in}})\|_p$
    $L \leftarrow \frac{1}{B}\sum_k \mathrm{Loss}(\mathrm{y}^{(k)}, \boldsymbol{f}^{\text{out}}(\boldsymbol{f}^{\text{in}}(\mathbf{x}^{(k)};\theta^{\text{in}});\theta^{\text{out}})))$
    $\theta \leftarrow \theta - \eta\nabla_\theta(L + \gamma R)$
end while
function $\mathrm{T}(\mathcal{D})$
    for $k = 1, \ldots, B$ do
        $(\mathbf{x}^{(k)}, \mathrm{y}^{(k)}) \leftarrow$ sample from $\mathcal{D}$
        $(\hat{\mathbf{x}}^{(k)}, \hat{\mathrm{y}}^{(k)}) \leftarrow$ sample from $\mathcal{D}$ with $\hat{\mathrm{y}}^{(k)} = \mathrm{y}^{(k)}$
    end for
return $\{(\mathbf{x}^{(k)}, \mathrm{y}^{(k)})\}_{k=1}^{B}, \{(\hat{\mathbf{x}}^{(k)}, \hat{\mathrm{y}}^{(k)})\}_{k=1}^{B}$
end function

Remark. The source codes are included in the zip file (/source_code).

S10.2 Calculation of invariance and selectivity scores

For the evaluation of invariance and selectivity scores, we calculate the entries of activation grid α_{ij}^{nm}, which is the averaged activation over the same category i and attribute j using all images; namely, we do not technically conduct variation-base decomposition. Then actual scores of invariance and selectivity are calculated.

S10.3 Calculation of mrA and moA

First of all, we calculate ρ^{nm} and a_{ij}^{nm} for all neurons using all images. To calculate mrA and moA scores, we regard elements of $\widetilde{a}^{nm} > 0$ are active and set them to be $\widetilde{a}^{nm} = 1$ and set others to be $\widetilde{a}^{nm} = 0$. Then calculate mrA by taking average of $\widetilde{a}^{nm}(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}_i$ and $i \in \mathcal{I}$ over all neurons and also calculate moA by taking average of $\widetilde{a}^{nm} \odot \widetilde{a}^{nm}$ over all neurons over all sampled pairs of images $\mathcal{X}_r$ (see definition of mrA 16 and moA 14).

S10.4 Preparation of seen data and unseen data in bias-controlled experiments

In bias-controlled experiment, we prepare the seen data $\mathcal{D}^{(\mathrm{seen})}$ that consists of images with biased object and nuisance attribute combinations; we denote the complement set of $\mathcal{D}^{(\mathrm{seen})}$ as $\mathcal{D}^{(\mathrm{unseen})}$.

Let $\mathbf{x}^{(k)}$ be an image and $\mathbf{y}^{(k)} = (i^{(k)}, j^{(k)})$ be the corresponding label. We divide our dataset as follows.

First, we select certain combinations of object and nuisance attribute categories $\mathcal{S} \subset \mathcal{I} \times \mathcal{J}$. Seen data is sampled as

$$\mathcal{D}^{(\mathrm{seen})} = \{(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}) | \mathbf{y}^{(k)} \in \mathcal{S}\}, \tag{Supp.1}$$

while the sampling process is force to satisfy the conditions described below. Let us define $\mathcal{S}|_{\mathcal{I}} \subset \mathcal{I}$ and $\mathcal{S}|_{\mathcal{J}} \subset \mathcal{J}$ as follows:

$$\mathcal{S}|_{\mathcal{I}} = \{i \in \mathcal{I} \mid (i,j) \in \mathcal{S} \text{ for at least one } j \in \mathcal{J}\}, \tag{Supp.2}$$
$$\mathcal{S}|_{\mathcal{J}} = \{j \in \mathcal{J} \mid (i,j) \in \mathcal{S} \text{ for at least one } i \in \mathcal{I}\}. \tag{Supp.3}$$

We impose conditions $\mathcal{S}|_{\mathcal{I}} = \mathcal{I}$ and $\mathcal{S}|_{\mathcal{J}} = \mathcal{J}$ to ensure that $\mathcal{D}^{(\mathrm{seen})}$ contains all object categories and all nuisance attribute categories (but not all combinations of them). Training data is sampled from $\mathcal{D}^{(\mathrm{seen})}$ so that each combination of object categories and nuisance attribute categories has the same number of images. Validation data (seen test data) is sampled from the rest of $\mathcal{D}^{(\mathrm{seen})}$ in the same way.

Unseen data is defined as

$$\mathcal{D}^{(\mathrm{unseen})} = \{(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}) | \mathbf{y}^{(k)} \notin \mathcal{S}\}. \tag{Supp.4}$$

Unseen test data is sampled from $\mathcal{D}^{(\mathrm{seen})}$ so that each combination of object categories and nuisance attribute categories has the same number of images. The term unseen test accuracy refers to the accuracy on the unseen test data constructed in this way. We also define seen rate as $s = \#(\mathcal{S}) / \#(\mathcal{I} \times \mathcal{J})$. To examine the dependency of the generalization on seen rate, we vary seen rate and attribute combinations as

$$\mathcal{S} \subset \mathcal{S}' \text{ if } s < s', \tag{Supp.5}$$

while keeping

$$\#(\mathcal{D}^{(\mathrm{seen})}(\mathcal{S})) = \#(\mathcal{D}^{(\mathrm{seen})}(\mathcal{S}')). \tag{Supp.6}$$

For each $j \in \mathcal{J}$, we define $\mathcal{S}_{*j} \subset \mathcal{I}$ as follows:

$$\mathcal{S}_{*j} = \{i \in \mathcal{I} \mid (i,j) \in \mathcal{S}\}. \tag{Supp.7}$$

We also define $\mathcal{S}_{i*} \subset \mathcal{J}$ for each $i \in \mathcal{I}$:

$$\mathcal{S}_{i*} = \{j \in \mathcal{J} \mid (i,j) \in \mathcal{S}\}. \tag{Supp.8}$$

This is not necessarily required but in our experiments, we only treat the cases where

$$\#(\mathcal{I}) = \#(\mathcal{J}) \tag{Supp.9}$$

and keep the following additional condition:

$$\#(\mathcal{S}_{*1}) = \#(\mathcal{S}_{*2}) = \cdots = \#(\mathcal{S}_{*\#(\mathcal{J})}) = \#(\mathcal{S}_{1*}) = \#(\mathcal{S}_{2*}) = \cdots = \#(\mathcal{S}_{\#(\mathcal{I})*}). \tag{Supp.10}$$

Then, seen rates are limited in the following form:

$$s = \frac{\#(\mathcal{S}_{*1})}{\#(\mathcal{I})} = \frac{\#(\mathcal{S}_{*2})}{\#(\mathcal{I})} = \cdots = \frac{\#(\mathcal{S}_{*\#(\mathcal{J})})}{\#(\mathcal{I})} = \frac{\#(\mathcal{S}_{1*})}{\#(\mathcal{J})} = \frac{\#(\mathcal{S}_{2*})}{\#(\mathcal{J})} = \cdots = \frac{\#(\mathcal{S}_{\#(\mathcal{I})*})}{\#(\mathcal{J})}. \tag{Supp.11}$$

For instance, if $\#(\mathcal{I}) = \#(\mathcal{J}) = 5$, seen rate takes $1/5$, $2/5$, $3/5$, $4/5$. Fig. 1 of the main body of the paper shows an example of $\#(\mathcal{I}) = \#(\mathcal{J}) = 5$ and $s = 2/5$.

In each trial of the experiments, we first determined the seen rate s and then created the set $\mathcal{S}$ randomly as long as it satisfied the constraints explained above. Then $\mathcal{D}^{(\text{seen})}$, training data, validation data (seen test data), $\mathcal{D}^{(\text{unseen})}$, and unseen test data were created accordingly. The actual numbers for each dataset are given in Sec. S12.

S11 Details of datasets

S11.1 MNIST-Positions

Starting with the MNIST dataset[215], we created a dataset of 42×42 pixels with nine numbers (0 to 8; 567378 images) by resizing images to 14×14 and placing them in one of 9 possible positions in a 3×3 empty grid. We call it MNIST-Positions. Fig. Supp.1 shows the all categories and positions of MNIST-Positions. In our experiments, the numbers are considered to be the object category set $\mathcal{I}$ and the positions where the numbers are placed is considered as the nuisance attribute set $\mathcal{J}$. Thus, it is written as $\#(\mathcal{I}) = \#(\mathcal{J}) = 9$.

S11.2 iLab-Orientations

iLab-2M is a dataset created from iLab-20M dataset[?] . The dataset consists of images of 15 categories of physical toy vehicles photographed in various orientations, eleva-

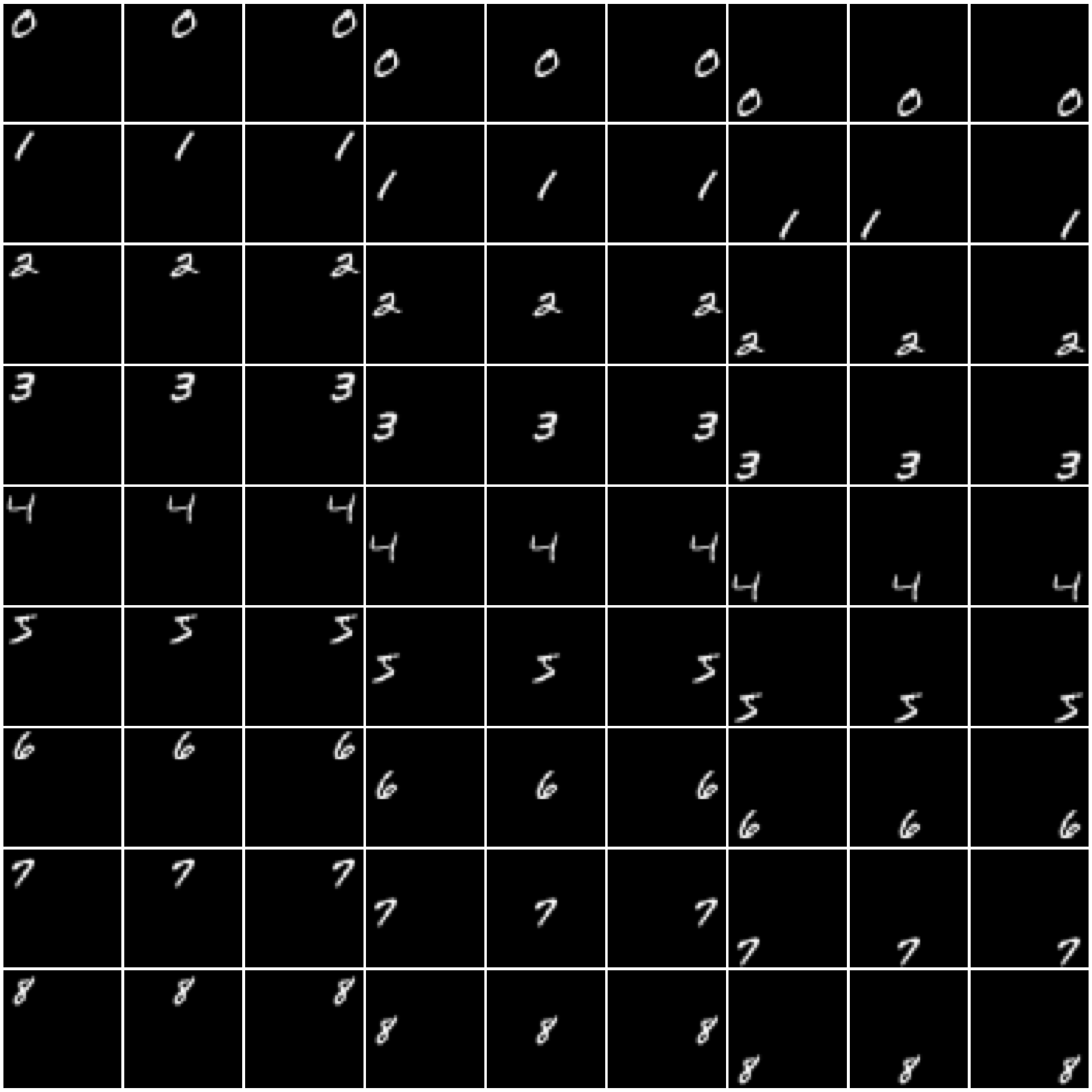


Figure Supp.1: Sample images of MNIST-Positions dataset arranged in a grid pattern. Each row indicates the number as the object category. MNIST-Poitions include nine numbers from 0 to 8. Each column indicates the positions as the nuisance attribute category. There are 9 positions in this dataset.

tions, lighting conditions, camera focus settings and backgrounds. It has 1.2M training images, 270K validation images, 270K test images, and each image is 256×256 pixels. We chose from the original iLab-2M dataset six object categories — bus, car, helicopter, monster truck, plane, and tank as $\mathcal{I}$ and six orientations as $\mathcal{J}$. We call it iLab-Orientations. Fig. Supp.2 shows samples of the all object categories and orientations of iLab-Orientations dataset. The sample size is 471791.

S11.3 CarCGs-Orientations

CarCGs-Orientations is a new dataset that consists of images of ten models of cars in various conditions rendered by Unreal Engine version 4.25.3[107]. The conditions consists of ten orientations, three elevations, ten body colors, five locations and three time slots. Fig. Supp.3 shows the all car models (object categories) and orientations (nuisance attribute categories) in the grid form. The details of these two main attributes are as follows.

- Object categories: CarCGs-Orientations dataset consists of images of the following cars — Nissan XTrail, Volkswagen Golf, BMW 2Series Coupe, Honda Odyssey, Toyota Prius, Mercedes Benz A-Class, Lexus LS, Mercedes Benz E-Class, Toyota Yaris and Volvo V40 (See Fig. Supp.4). We used the whole car models as object categories $\mathcal{I}$. Therefore the number of object categories is $\#(\mathcal{I}) = 10$ in the experiments conducted in this study.

- Orientations (nuisance attribute categories): During the rendering process, the virtual camera (camera actor) was rotated around the yaw axis of each car from 0 to 324 degrees in units of 36 degrees. Therefore, each car model appears in the images with ten different azimuth orientations. All orientations are shown in Fig. Supp.5. We used the whole orientations as nuisance attribute categories $\mathcal{J}$. Thus the number of the nuisance attribute categories is $\#(\mathcal{J}) = 10$ in the experiments conducted in this study.

To create variety of samples for each combination of the object categories (car models) and nuisance attribute categories (orientations), we added other attributes as follows.

- Elevations: The virtual camera was located at three elevation angles, namely, 10, 15, and 30 degrees, during the rendering process. Sample images taken from each angle are shown in Fig. Supp.6.

- Body colors: Each car model is rendered with ten colors, namely, black, light blue, green, red, white, beige, dark blue, orange, plum, and silver by using Automotive Materials[106] (a library for Unreal Engine). Fig. Supp.7 shows sample images of Nissan XTrail rendered with these colors.

Figure Supp.2: Sample images of iLab–Orientations dataset arranged in a grid pattern. Each row indicates the object category. iLab-Orientations include six object categories — bus, car, helicopter, monster truck, plane, and tank. Each column indicates the orientations as the nuisance attribute category. There are 6 orientations in this dataset.

Figure Supp.3: Sample images of each object category and orientation of CarCGs-Orientations. Each row indicates object categories — Nissan XTrail, Volkswagen Golf, BMW 2Series Coupe, Honda Odyssey, Toyota Prius, Mercedes Benz A-Class, Lexus LS, Mercedes Benz E-Class, Toyota Yaris and Volvo V40. Each column indicates the nuisance attribute categories, orientations from 0 to 324 degrees. These categories and orientations in this figure are used in our experiments.

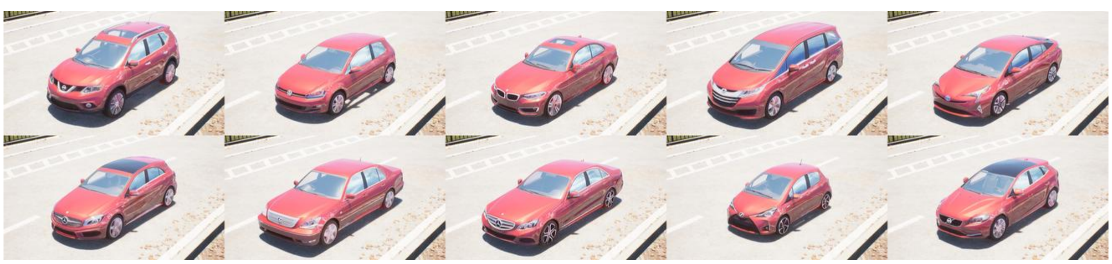


Figure Supp.4: Sample images of ten object categories of CarCGs-Orientatoins — Nissan XTrail, Volkswagen Golf, BMW 2Series Coupe, Honda Odyssey, Toyota Prius, Mercedes Benz A-Class, Lexus LS, Mercedes Benz E-Class, Toyota Yaris and Volvo V40 are shown in this figure from the top left to the bottom right. All attributes except object category are fixed in this figure.

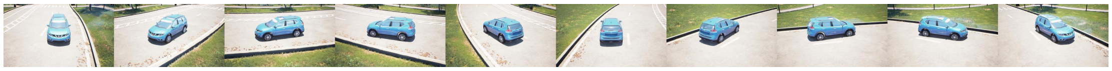


Figure Supp.5: Sample images of ten orientations (nuisance attribute categories) of CarCGs-Orientations. Ten orientations from 0 to 324 degrees are displayed from the left to right. All attributes except orientation are fixed in this figure.

Figure Supp.6: Sample images of the three elevation angles (10, 15, 30 degrees) of CarCGs-Orientations. Left figure is the image whose elevation angle is 10 degrees. Middle figure is the image whose elevation angle is 15 degrees. Right figure is the image whose elevation angle is 30 degrees.

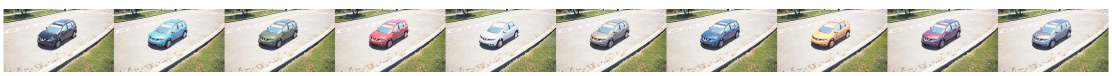


Figure Supp.7: Sample images of cars painted in ten colors of CarCGs-Orientations. There are images painted in black, light blue, green, red, white, beige, dark blue, orange, plum, and silver from left to right.

(a) Sample images of each location from elevation angle of 10 degrees that CarCGs-Orientations has.

(b) Sample images of each location from elevation angle of 15 degrees that CarCGs-Orientations has.

Figure Supp.8: (a) is images of a car placed in five different locations taken from elevation angle of 10 degrees. (b) is images taken from elevation angle of 15 degrees. There are differences in texture of road and background.

- Locations: We used a sample environment of an urban park contained in City Park Environment Collection[SilverTm]. We chose five locations from the sample environment and modified them for our experiments. Sample images taken at each location are shown in Fig. ??.
- Time slots: We used Ultra Dynamic Sky[Everett Gunther] to synthesize the three different times slots, namely, daytime, twilight, and midnight. Fig. Supp.9 shows the samples of these three time slots.

The number of images and the image size are as follows.

- Number of images and image size: The total number of images of this dataset is 45K = 10 (object categories) $\times$ 10 (orientations) $\times$ 3 (elevations) $\times$ 10 (body colors) $\times$ 5 (locations) $\times$ 3 (time slots). The images are rendered in 3840$\times$2160 pixels and then resized to 1920$\times$1080 pixels for the sake of anti-aliasing.

Remark. In addition to the figures in this Appendix, ten jpeg files are contained in the zip file (/sample_images/CarCGs-Orientations).

Remark. After the submission of the main body of our paper, it turned out that 3.3% of the car model labels were wrongly created. Both seen data and unseen data used for the experiments contained these wrongly labeled images. Yet, the networks with and without our proposed method were trained and tested with the same seen data and unseen data, and the comparisons reported in the paper are still fair.

S11.4 MiscGoods-Illumination

MiscGoods-Illumination is a novel dataset constructed for this study. The dataset consists of images of ten physical miscellaneous goods taken with five illumination conditions, two object aspects, twenty object orientations, five camera angles. Figure Supp.10 shows the all miscellaneous goods (object categories) and illumination conditions (nuisance attribute categories) in the grid form. The details of these two main attributes are as follows.

- Object Categories: As shown in Fig. Supp.10, MiscGoods-Illumination has ten types of miscellaneous goods — stuffed dolphin, stuffed whale, metal basket, imitation plant, cup, cleaning brush, winding tape, lace yarn, bottled imitation tomatoes, and bottled imitation green apples. In this study, we selected the following five miscellaneous goods as the object categories $\mathcal{I}$ — stuffed dolphin, stuffed whale, metal basket, imitation plant and cup. Therefore the number of object categories is $\#(\mathcal{I}) = 5$ in the experiments conducted in this study.
- Illumination conditions (nuisance attribute categories): As the nuisance attribute categories, we created five illumination conditions (lighting conditions);

one is created with ceiling lights, and the rest are with a colored spotlight. All illumination conditions are shown in Fig. Supp.10. For spotlight conditions, the light source (G1S RGB Video Light) was placed 23 cm in front of the object (See Fig. Supp.11). The parameters of the light source were H217/S141=8500k (white light), H0/S100 (red light), H120/S100 (green light), and H240/S100 (blue light). These parameters were set so that the nuisance attribute of the illumination makes a sufficient difference in the learning experiments. We used whole illumination conditions as nuisance attribute categories $\mathcal{J}$. Thus the number of the nuisance attribute categories is $\#(\mathcal{J}) = 5$ in the experiments conducted in this study.

As we did for CarCGs-Orientations, we added other attributes to create variety of samples for each combination of the object categories and nuisance attribute categories (illumination conditions) as follows.

- Object poses (aspects and orientations): In MiscGoods-Illumination, we placed each object in two representative aspects for each lighting condition. Fig. Supp.12 shows the two aspects of all object categories. For additional diversity, we rotated the object every 18 degrees from 0 to 342 degrees (Fig. Supp.13). In total, there are 40 patterns in object pose conditions.

- Camera angles: To capture the images automatically, we created a robotic image capture system (see Fig. Supp.11). A camera (Realsense D435[Intel]) was attached to a robot arm (COBOTTA[DENSO Wave]), and the system captured images from five camera angles for each lighting and object pose condition (Fig. Supp.14). The postures were defined so that the acquired image shows the entire object pose. The series of operations from robot control to image acquisition is automated by utilizing ROS kinetic[ros].

The number of images and the image size are as follows.

- Number of images and image size: The number of images of whole dataset is 10K = 10 (object categories) $\times$ 5 (illuminations) $\times$ 2 (aspects) $\times$ 20 (orientations) $\times$ 5 (camera angles), and each image size is 640 $\times$ 480 pixels.

Remark. In addition to the figures in this Appendix, ten jpeg images are contained in the zip file (/sample_images/MiscGoods-Illumination).

S12 Details of experiments

ResNet-18[163] is adopted as the network for all experiments. The source codes are implemented based on Python v3.6.9, using TensorFlow v2.1.0 and NumPy v1.18.1. They are included in the zip file (/source_code). The whole network architecture is shown in Fig Supp.1 and Fig Supp.2. All neurons employ the rectified linear function $g(z) = \max\{0, z\}$ and satisfy $a^{nm}(\mathbf{x}) \geq 0$. Glorot uniform initializer[125] is adopted for the network weights initialization for all experiments. We use BatchNormalization to standardize the inputs to a layer for each mini-batch. We use it for stabilizing the learning process and reducing the number of training epochs. We do not use any data augmentations and regularizations except IER to measure the effect of IER in an appropriate manner. IER is applied to the last fully-connected layer "activation_17" with 512 neurons shown in Fig. Supp.1. Adam[197] is employed as the optimization algorithm. The cross-entropy loss is employed as the loss L.

The pixels of images are normalized within 0 to 1 as a preprocessing for all datasets. The epoch size, learning rate, and batch size are confirmed to produce reasonable accuracy in the vanilla case (i.e., without IER) for each dataset and we employ the same values for all experiments with the same dataset. For example, we use 15, 0.001, and 128 as epoch size, learning rate and batch size, respectively, for MNIST-Positions. The values of hyper-parameters are summarized in Table Supp..1.

Table Supp..1: Hyper-parameters used for each dataset

dataset	epoch size	preprocessing	weights initialization	learning rate	b
MNIST-Positions	15	divide by 255	Glorot uniform initializer	0.001	
iLab-Orientations	50	divide by 255	Glorot uniform initializer	0.001	
CarCGs-Orientations	50	divide by 255	Glorot uniform initializer	0.0001	
MiscGoods-Illumination	50	divide by 255	Glorot uniform initializer	0.00001	

Average running time and confidence interval of learning of five trials with IER using L2 norm have been measured to be 1956.2 (1885.2-2027.2) seconds, comparing with 1172.4 (1090.8-1254.0) seconds of vanilla case. This evaluation has been conducted using MNIST-Positions and parameters are same as the experiments reported in Sec 4.1 (e.g. seen rate $s = 4/9$ and epoch size is 15). We have employed two Quadro RTX 6000 GPUs for the experiments.

Regarding the invariance and selectivity scores, we use the median of the scores of all neurons to represent the layer activity. Regarding the calculation of the confidence intervals, Student-t test has been used. The preparation of training data, validation data (seen test data), and unseen test data has been conducted as follows.

- MNIST-Positions: We used images of the original MNIST-Positions with image size of 42 × 42 pixels. Seen data and unseen data were prepared in the way described in Sec. S10.4 with seen rate $s = 2/9, 3/9, 4/9, 5/9, 6/9, 7/9, 8/9$ for the experiments of weight dependency and $s = 4/9$ for the other experiments. The number of seen data is $\#(\mathcal{D}^{(\text{seen})}) = 110000$. We use 100000 images for training and 10000 for validation (seen test) for hyper-parameter tuning. The number of unseen data is $\#(\mathcal{D}^{(\text{unseen})}) = 10000$. All of them were used for unseen test.

- iLab-Orientations: We reize the images to 64×64 pixels. Seen data and unseen data are prepared in the way described in Sec. S10.4 with seen rate $s = 1/6, 2/6, 3/6, 4/6, 5/6$ for the experiments of weight dependency and $s = 3/6 = 1/2$ for the other experiments. The number of seen data is $\#(\mathcal{D}^{(\text{seen})}) = 78000$. We use 70000 images for training and 8000 for validation (seen test) for hyper-parameter tuning. The number of unseen data is $\#(\mathcal{D}^{(\text{unseen})}) = 8000$. All of them are used for unseen test.

- CarCGs-Orientations: We resize the images to 224×224 pixels. Seen data and unseen data were prepared in the way described in Sec. S10.4 with seen rate $s = 5/10 = 1/2$. The number of seen data is $\#(\mathcal{D}^{(\text{seen})}) = 3845$. We use 3400 images for training and 445 for validation (seen test) for hyper-parameter tuning. The number of unseen data is $\#(\mathcal{D}^{(\text{unseen})}) = 800$. All of them are used for unseen test.

- MiscGoods-Illumination: We resize the images to 224×224 pixels. Seen data and unseen data are prepared in the way described in Sec. S10.4 with seen rate $s = 3/5$. The number of seen data is $\#(\mathcal{D}^{(\text{seen})}) = 1000$. We use 800 images for training and 200 for validation (seen test) for hyper-parameter tuning. The number of unseen data was $\#(\mathcal{D}^{(\text{unseen})}) = 200$. All of them are used for unseen test.

S13 Train accuracy and test accuracy on seen data

Training, seen test, and unseen test accuracy corresponding to Fig.4 in the main body of our paper are available in Fig. Supp.3. The experiments for measuring accuracy is exactly same as what we reported in the main body of the paper. The seen rate s employs 4/9, 3/6, 5/10, and 3/5 for MNIST-Positions, iLab-Orientations, CarCGs-Orientations, and MiscGoods-Illumination, respectively. The regularization weight γ for L1, L2, and Linf norms are set at 0.001, 0.01, and 0.01, respectively. We conduct five trials and provide arithmetic means and 95% confidence intervals of seen and unseen test accuracy. Accuracy of seen test of weight dependency corresponding to Fig. 5(a)(b) in the main body of the paper are available in Fig. Supp.4. Moreover, accuracy of seen test of weight dependency corresponding to Fig. 5(c)(d) in the main body of our paper are available in Fig. Supp.5.

The most important point is that the dependency of unseen test accuracy on the regularization weight γ shows the same trends as the dependency of seen test accuracy on γ (Figs. Supp.5c to Supp.5f). This result implies that seen test accuracy can be used to choose the regularization weight.

S14 For internal use

S15 List of symbols

- $\mathbf{x}$: Image [k-th image $= \mathbf{x}^{(k)}$]
- $\mathbf{y}$: Label [k-th label $= \mathbf{y}^{(k)}$]
- k: Sample index
- n: Layer index
- m: Neuron index
- M_n: Number of neurons in layer n
- $a^{nm}(\mathbf{x})$: Activity of the m-th neuron in the n-the layer
- $\widetilde{a}^{nm}(\mathbf{x})$: Variation activity of the m-th neuron in the n-the layer
- ρ^{nm}: Scale factor of the m-th neuron in the n-the layer
- b^{nm}: Base activity of the m-th neuron in the n-the layer
- α^{nm}_{ij}: Entries of activations grid
- $\boldsymbol{a}^n(\mathbf{x})$: Layer activity [$\boldsymbol{a}^n(\mathbf{x}) = [a^{n1}(\mathbf{x}), \dots, a^{nM_n}(\mathbf{x})]^\top$]
- $\widetilde{\boldsymbol{a}}^n(\mathbf{x})$: Layer variation activity [$\widetilde{\boldsymbol{a}}^n(\mathbf{x}) = [\widetilde{a}^{n1}(\mathbf{x}), \dots, \widetilde{a}^{nM_n}(\mathbf{x})]^\top$]
- ρ^n: Layer scale factor [$\rho^n = [\rho^{n1}, \dots, \rho^{nM_n}]^\top$]
- $\boldsymbol{b}^n(\mathbf{x})$: Layer base activity [$\boldsymbol{b}^n = [b^{n1}, \dots, b^{nM_n}]^\top$]
- θ: Network parameter [$\theta = (\theta^{\mathrm{in}}, \theta^{\mathrm{out}})$]
- $\boldsymbol{f}^{\mathrm{in}}$: Input side of neural network [$\boldsymbol{a}^n(\mathbf{x}) = \boldsymbol{f}^{\mathrm{in}}(\mathbf{x}; \theta^{in})$]
- $\boldsymbol{f}^{\mathrm{out}}$: Output side of neural network
- $\mathcal{I}$: Set of object categories
- $\mathcal{J}$: Set of nuisance attribute categories

- N: Number of object category ($N = \#(\mathcal{I}) = \#(\mathcal{J})$ in this study)
- i: Object category index ($i \in \mathcal{I}$)
- j: Nuisance attribute category index ($j \in \mathcal{J}$)
- I^{nm}: Invariance score of the m-th neuron in the n-the layer
- S^{nm}: Selectivity score of the m-th neuron in the n-the layer
- $\mathcal{X}_i$: Set of images belonging th object category i
- L: Loss term
- R: Regularization term
- $\mathrm{Loss}(\cdot, \cdot)$: Loss function
- $\#(\cdot)$: Number of elements
- $\#(\cdot)_{\neq 0}$: Number of non-zero elements
- B: Mini-batch size
- $\{(\cdot, \cdot)\}_{k=1}^{B}$: Mini-batch
- $\|\cdot\|$: Lp norm
- $\underset{i}{\mathrm{average}}$ or $\mathrm{average}_i$: Average with respect to i
- $\max\limits_{i}$ or $\max_i$: Max value with respect to i
- $\min\limits_{i}$ or $\min_i$: Min value with respect to i
- $\underset{i}{\mathrm{argmax}}$ or argmax_i: Argument i of maximum value
- $\odot$: Element-wise product
- $\oslash$: Element-wise division
- $P_{\mathbf{x}}^{n}$: Activation Pattern (set of indices of neurons activated by an image $\mathbf{x}$)
- $\mathfrak{P}_i$: Set of activation patterns corresponding to the object category i
- $\mathrm{E}[\cdot]$: Expected value
- $\mathcal{D}^{(\mathrm{seen})}$: Seen data
- $\mathcal{D}^{(\mathrm{unseen})}$: Unseen data
- s: Seen rate

Figure Supp.9: Sample images of each time slot of CarCGs-Orientations. Left figure is an image of car taken in the daytime. Middle figure shows an image of a car taken in the twilight. The color of the car is different from that in the left and right ones. It is caused by the twilight sunlight condition. Right figure shows an image of car taken at midnight. The color of the car is also different from left and middle ones.

Figure Supp.10: Sample images of each object category and illumination condition of MiscGoods-Illumination are shown in this figure. Each row indicates object categories — stuffed dolphin, stuffed whale, metal basket, imitation plant, cup, cleaning brush, winding tape, lace yarn, bottled imitation tomatoes, and bottled imitation green apples. Each column indicates the nuisance attribute categories, illumination conditions — ceiling light, white spotlight, red spotlight, green spotlight and blue spotlight. These five categories from the top and five illumination conditions are used as object categories and nuisance attribute categories in our experiments.

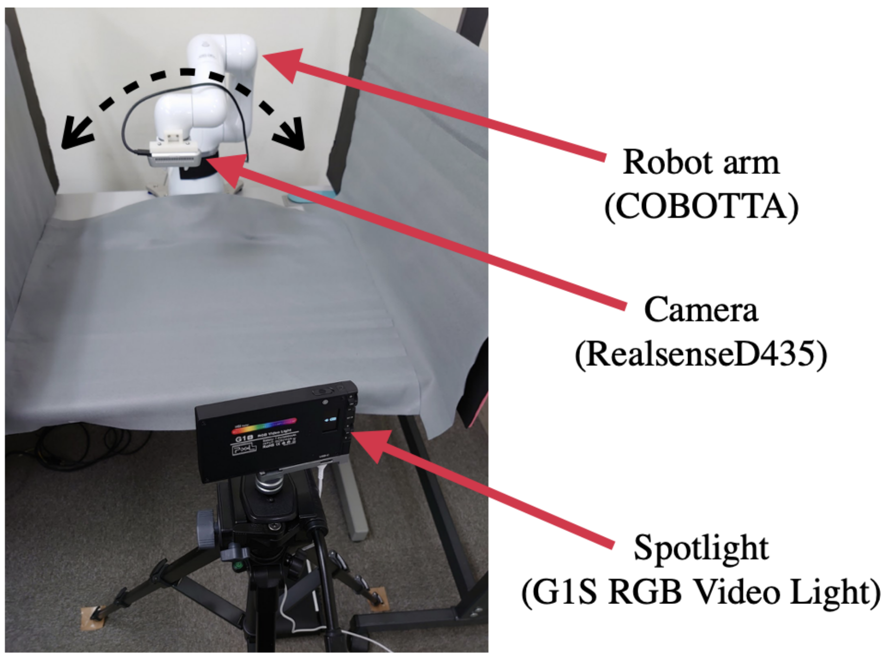


Figure Supp.11: Robotic image capture system for MiscGoods-Illumination. Dashed bidirectional arrow indicates the robot motion.

Figure Supp.12: Sample images of MiscGoods-Illumination with two aspects. Each object has these two aspects as an attribute. The shapes of these objects in an image are changed by aspects conditions.

Figure Supp.13: Sample images of each orientation of MiscGoods-Illumination. 20 orientations from 0 to 342 degrees that the dataset has are shown in this figure from the top left to the bottom right.

Figure Supp.14: Sample images from each camera angles of MiscGoods-Illumination. There are five angles in the dataset. The postures were defined so that the acquired image shows the entire object pose. These five camera angles are related to postures of robot arm that the camera is connected.

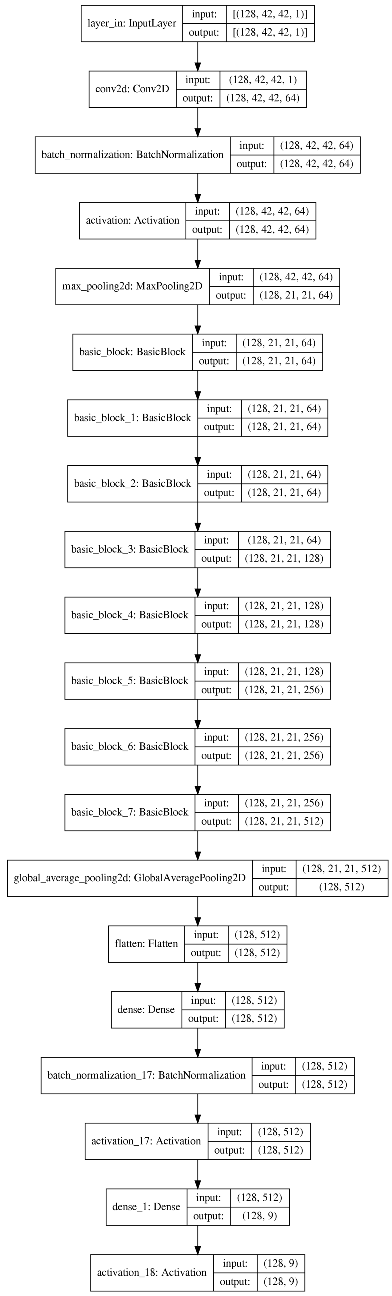


Figure Supp.1: This diagram shows the whole architecture of our implementation of ResNet-18. The numbers in this diagram represent batchsize, height of image, width of image and channels. Therefore they change depending on the dataset. Current numbers correspond to MNIST-Positions. For instance, the numbers on the top of the diagram means $(batchsize, height, width, channels) = (128, 42, 42, 1)$. Conv2D, Dense and BasicBlock mean a convolutional layer, a fully connected layer and a basic building block of ResNet, respectively.

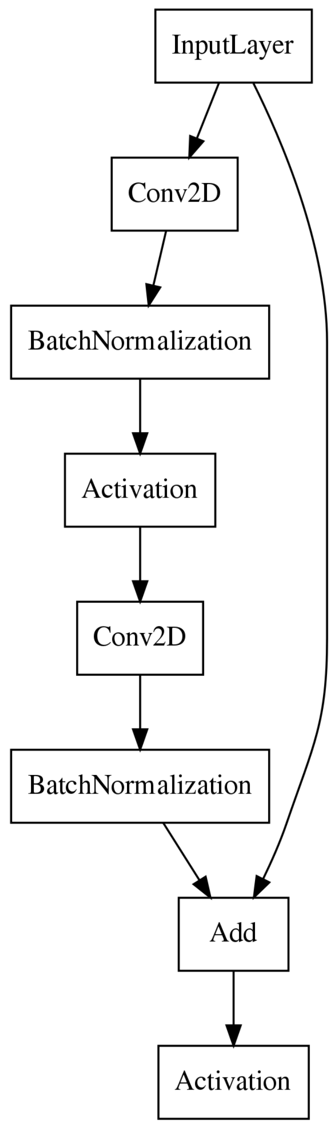
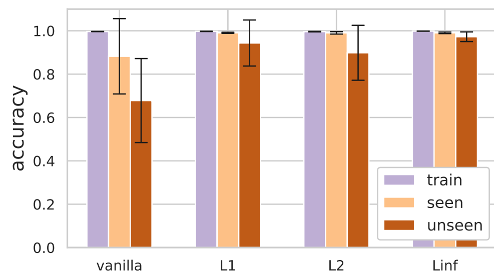
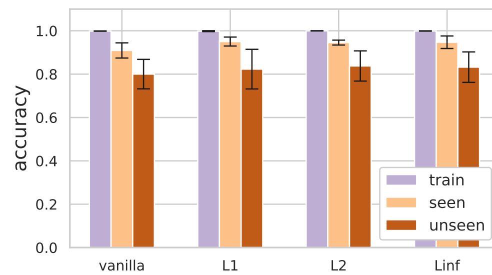


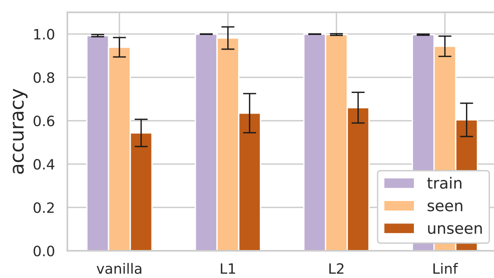
Figure Supp.2: This diagram shows the architecture of BasicBlock in ResNet-18. Conv2D and Add mean a convolutional layser and a layer that simply add the two input values.



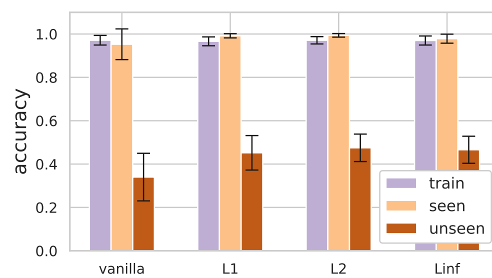
(a) Accuracy in MNIST-Positions



(b) Accuracy in iLab-Orientations

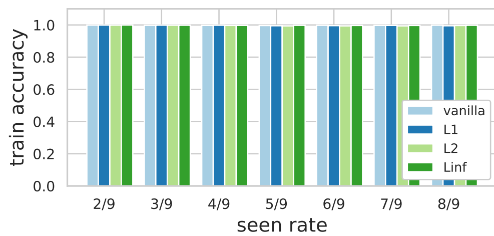


(c) Accuracy in CarCGs-Orientations

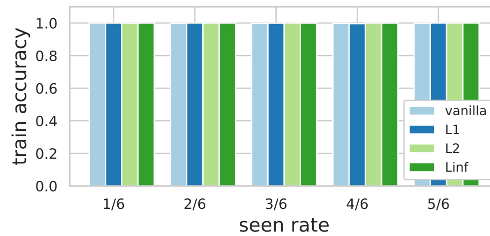


(d) Accuracy in MiscGoods-Illumination

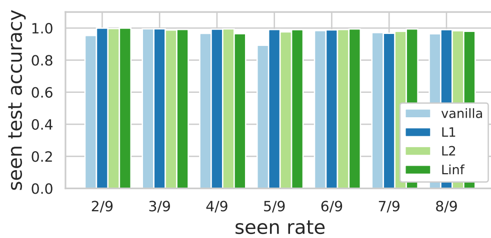
Figure Supp.3: The training accuracy, seen test accuracy, and unseen test accuracy in the vanilla case and cases with IER: each subfigure shows the mean and 95% confidence interval. The regularization weight γ for L1, L2, and Linf norms are set at 0.001, 0.01, and 0.01, respectively. Note that training accuracy denotes accuracy on train data, seen test accuracy denotes accuracy on seen data $\mathcal{D}^{(\text{seen})}$, and unseen test accuracy denotes accuracy on unseen data $\mathcal{D}^{(\text{unseen})}$. In all cases the train accuracy and test accuracy have achieved values close to 1.0. Also, in many cases the degradation of the accuracy from the seen test accuracy to unseen test accuracy is less in cases with IER than vanilla cases.



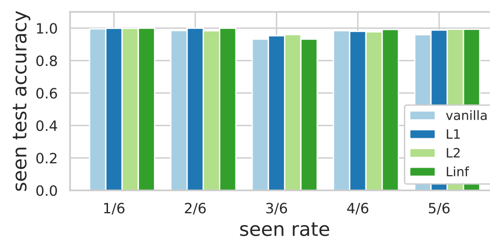
(a) Seen rate dependency of train accuracy in MNIST-Positions



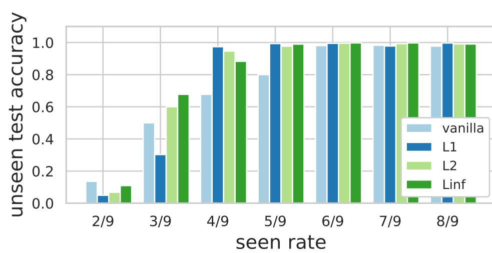
(b) Seen rate dependency of train accuracy in iLab-Orientations



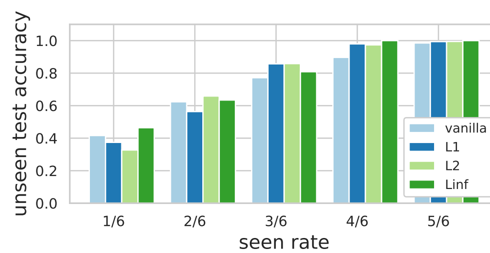
(c) Seen rate dependency of seen test accuracy in MNIST-Positions



(d) Seen rate dependency of test accuracy in iLab-Orientations

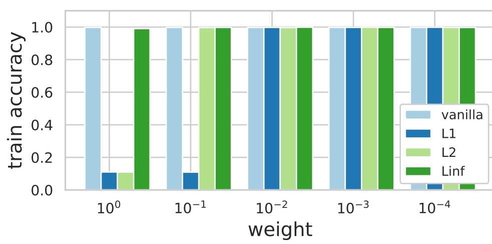


(e) Seen rate dependency of unseen test accuracy in MNIST-Positions

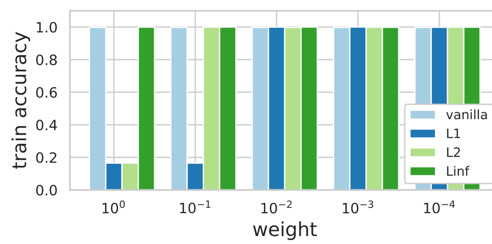


(f) Seen rate dependency of unseen test accuracy in iLab-Orientations

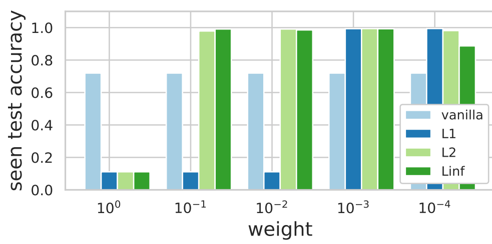
Figure Supp.4: The seen rate dependency of training, seen test, and unseen test accuracy for the case of vanilla and with IER. The regularization weight γ for L1, L2, and Linf norms are set at 0.001, 0.01, and 0.01, respectively. Note that training accuracy denotes accuracy on training data and seen test accuracy denotes accuracy on seen data $\mathcal{D}^{(\mathrm{seen})}$. In all cases, the training accuracy and the seen test accuracy are very close to 1.0.



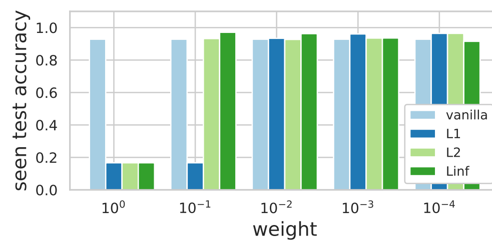
(a) Weight dependency of train test accuracy in MNIST-Positions



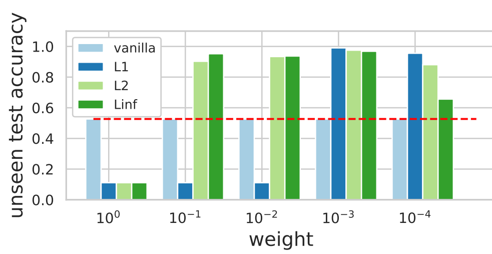
(b) Weight dependency of train test accuracy in iLab-orientations



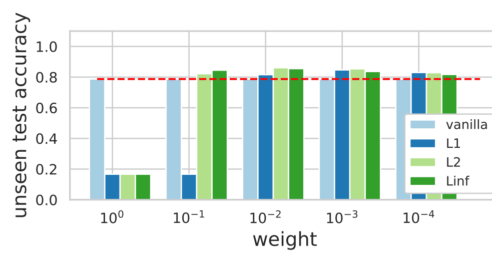
(c) Weight dependency of seen test accuracy in MNIST-Positions



(d) Weight dependency of seen test accuracy in iLab-orientations



(e) Weight dependency of unseen test accuracy in MNIST-Positions



(f) Weight dependency of unseen test accuracy in iLab-orientations

Figure Supp.5: The weight dependency of training, seen test and unseen test accuracy for the case of vanilla and with IER. Note that train accuracy denotes accuracy on train data and seen test accuracy denotes accuracy on seen data $\mathcal{D}^{(\text{seen})}$. IER has a drop in training accuracy if the weight is too large. On the other hand, we find that tendency of seen test accuracy and unseen test accuracy is consistent. Generally it is difficult to measure unseen test accuracy, but if the seen test accuracy is high, we can expect unseen test accuracy to be the same.

S16 Domains

We present details for the ATARI PacMan, Pong and Breakout domains.

S16.1 PacMan

PacMan is set in a two-dimensional grid that contains food, walls, ghosts, and the PacMan character. The game concludes with a +500 reward when all food pellets are consumed, while encountering a ghost results in a -500 penalty and game over. Each consumed food pellet awards +10 points, and PacMan incurs a -1 penalty for every time step. The available actions for PacMan are moving Up, Down, Right, or Left. The game's state includes the location of PacMan, the position and direction of any ghosts, and the distribution of food pellets. In this iteration of the game, ghosts move according to some distributions.

S16.2 Pong

In this one-player version of Pong, the player competes against a computer-controlled paddle. The game is set on a two-dimensional grid, with the player controlling one paddle and the computer controlling the other. The game concludes with a +500 reward when the ball reaches the grid boundaries on the computer controlled paddle side, while if the grid boundary is reached on the agent's side, a -500 penalty is applied and game over. The agent incurs a -1 penalty for every time step. The available actions for the paddles are moving Right and Left or to Stop. The game's state includes the location of the ball and the position and direction of any paddle. In this iteration of the game, the computer controlled paddle moves according to some distribution. Visualizations of the grids are presented in Supp.1.

S16.3 Breakout

In this version of Breakout, the agent competes against a wall of bricks using a horizontally-moving paddle and a ball. The game is set on a two-dimensional grid, with the agent controlling the paddle located at the bottom of the screen. The objective is to break bricks by hitting them with the ball, which bounces back after each hit. The game concludes with a +500 reward when all bricks are destroyed, but if the ball passes the paddle and reaches the bottom grid boundary, a -500 penalty is applied, resulting in game over. Each hit brick awards +10 points, and the agent incurs a -1 penalty for every time step. The available actions for the agent's paddle are moving Right or Left, or choosing to Stop. The game's state includes the position of the ball, the location of the paddle, and the configuration and status of the bricks. Visualizations of the grids are presented in Supp.2.

S17 Training Parameters

In our experiments, parameters for Q-Learning and SARSA are inherited by[54]. In particular, $\mathcal{T} = 1.5$ $\alpha = 0.05$, and $\lambda = 0.9$.

S18 Additional Graphs Non-Semantic Variations

In this section we present supplementary results showing the Generalization Agent and Learnability Agent behavior for Semantic variations of grids throughout Pacman and Pong.

S18.1 PacMan

Additional results showing the Generalization Agent and Learnability Agent behaviour in Pacman for grids v2, v3, v4, are presented in the Supplementary figures. In particular, results for SARSA Agent with Boltzmann exploration strategy are presented in Supp.3. Supp.4, Supp.5 show Q-learning Agent with Boltzmann and ε-greedy exploration strategies respectively.

S18.2 Pong

Similarly, for Pong grids p1, p2 results are presented in the Supplementary figures Supp.6 for SARSA Agent and Supp.7, Supp.8 for Q-learning Agent.

S18.3 Breakout

Analogously, for Breakout grids b1, b2, b3 results are presented in the Supplementary figures Supp.9, Supp.10 for SARSA Agent and Supp.11, Supp.12 for Q-learning Agent.

S19 Additional Graphs Semantic variations

In this section we present supplementary results showing the Generalization Agent and Learnability Agent behavior for Semantic variations of grids throughout Pacman and Pong.

S19.1 PacMan

The behavior of the Generalization and Learnabilty Agents under semantic variations of PacMan on grids v2, v3, v4 are presented in Supplementary figures Supp.13 for SARSA Agent and Supp.14 and Supp.15 for Q-learning Agent.

S19.2 Pong

Similarly, for Pong grids p1, p2 results are presented in the Supplementary figures. In particular, semantic variations featuring Directional Ghost $p = 0.3$ are presented in Supp.16, Supp.17 for SARSA Agent and Supp.18, Supp.19 for Q-learning Agent. While semantic variations featuring Directional Ghost $p = 0.6$ are shown in Supp.20, Supp.21 for SARSA Agent and Supp.22, Supp.23 for Q-learning.

S20 Additional Graphs State-Action Pairs

This section shows the supplementary results for the exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}) throughout the analyzed domains.

S20.1 PacMan

Results of the exploration grid for PacMan v2, v3, v4 are shown in Supplementary figures. In particular, for non-semantic grid variations, Supp.24 and Supp.25 report grid exploration graphs for Q-learning Agent and Supp.26 and Supp.27 for SARSA Agent. Additionally, for semantic games variations, Supp.28 and Supp.29 report grid exploration graphs for Q-learning Agent and Supp.30 and Supp.31 for SARSA Agent.

S20.2 Pong

Similarly, for pong p1 and p2, Supp.32, Supp.33, Supp.34, and Supp.35 report grid exploration graphs for non-semantic variations of Q-learning Agent and SARSA Agent respectively, while Supp.36, Supp.37, Supp.38, and Supp.39 for semantic variations.

S20.3 Breakout

For Breakout grids b1,b2, and b3, exploration graphs for non-semantic variations of Q-learning Agent and SARSA Agent are reported in Supplementary figures Supp.40, Supp.41, Supp.42, and Supp.43.

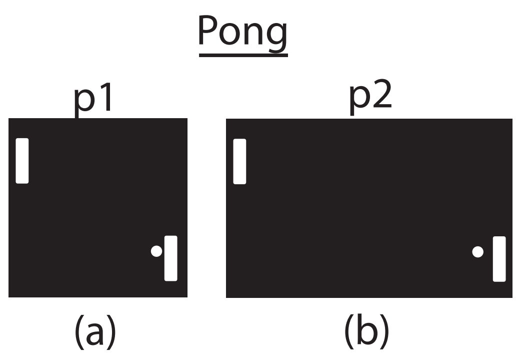


Figure Supp.1: Grid variations for Pong.

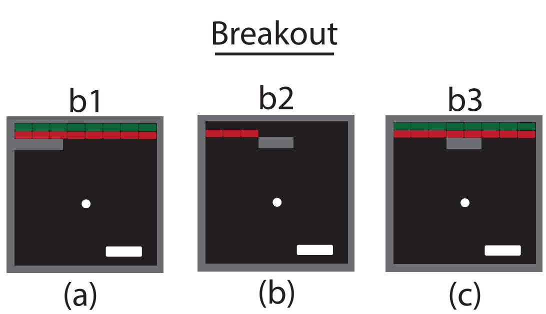


Figure Supp.2: Grid variations for Breakout.

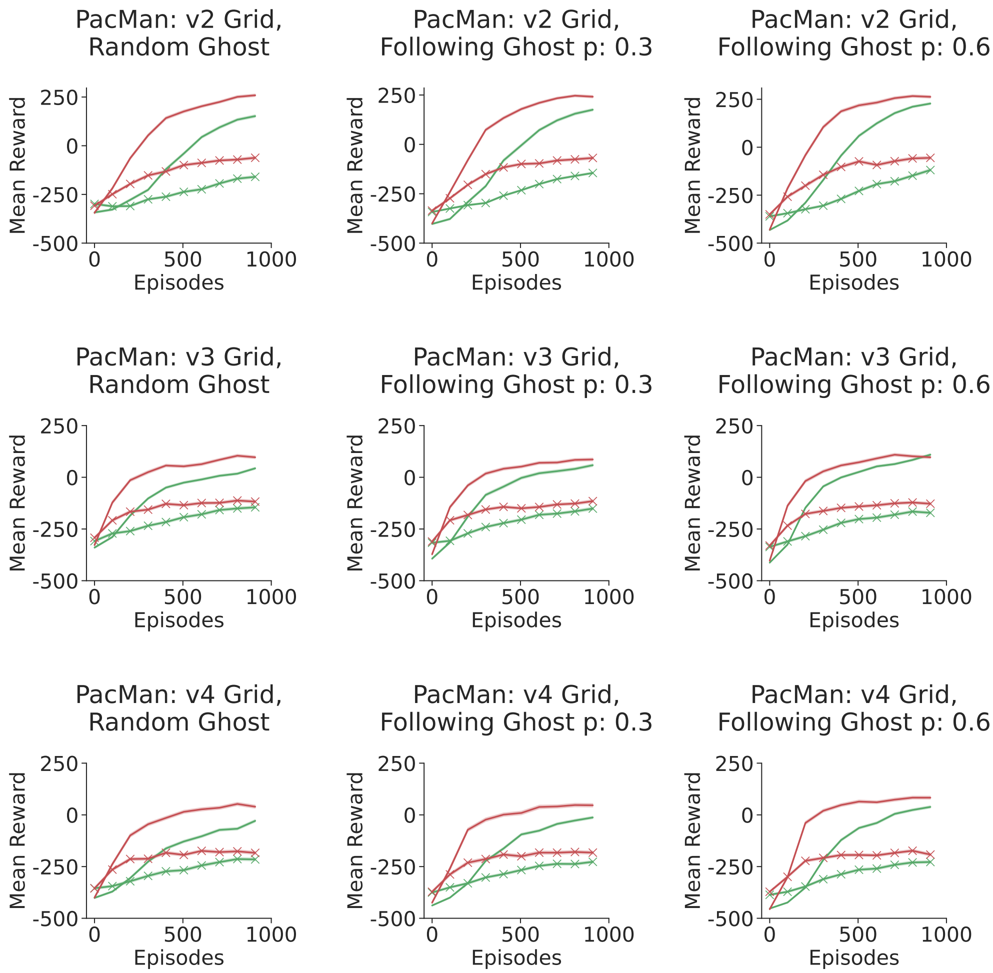


Figure Supp.3: SARSA Agent with Boltzmann exploration strategy: Results for PacMan v2, v3, v4 grids reporting mean reward as a function of episode number. The agent is trained on the non-noisy version of the environment and tested on different level of noise ($\partial \sim \mathcal{N}(0, 0.1)$ in Low-Noise and $\partial \sim \mathcal{N}(0, 0.5)$ in High-Noise settings).

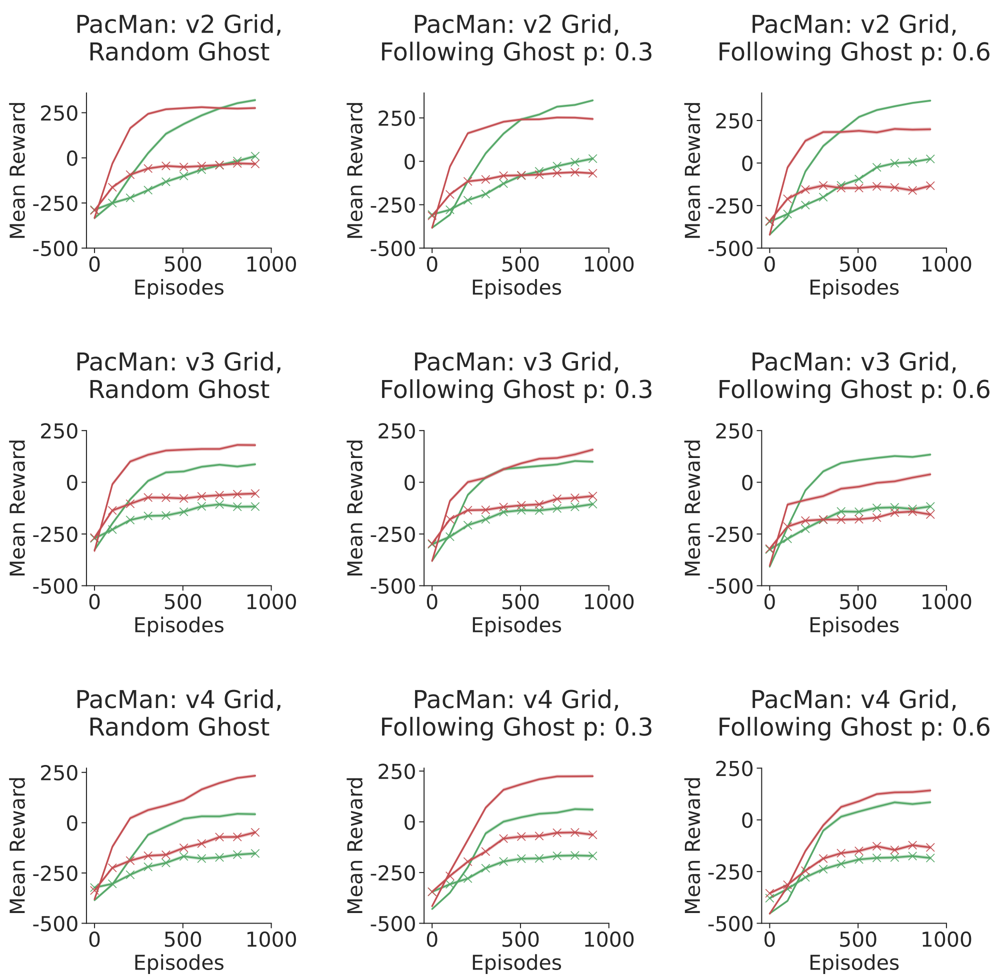


Figure Supp.4: Q-learning Agent with Boltzmann exploration strategy: Results for PacMan v2, v3, v4 grids reporting mean reward as a function of episode number. The agent is trained on the non-noisy version of the environment and tested on different level of noise ($\delta \sim \mathcal{N}(0, 0.1)$ in Low-Noise and $\delta \sim \mathcal{N}(0, 0.5)$ in High-Noise settings).

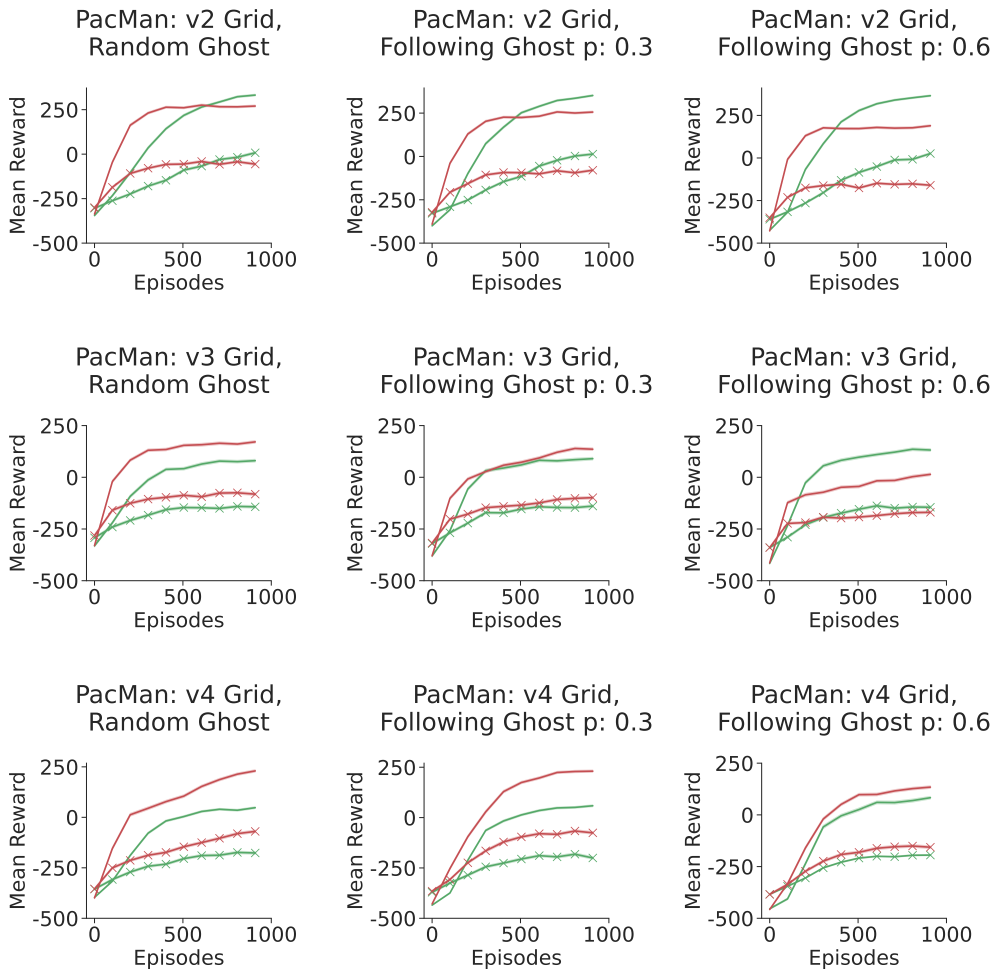


Figure Supp.5: Q-learning Agent with ε-greedy exploration strategy: Results for PacMan v2, v3, v4 grids reporting mean reward as a function of episode number. The agent is trained on the non-noisy version of the environment and tested on different level of noise ($\partial \sim \mathcal{N}(0, 0.1)$ in Low-Noise and $\partial \sim \mathcal{N}(0, 0.5)$ in High-Noise settings).

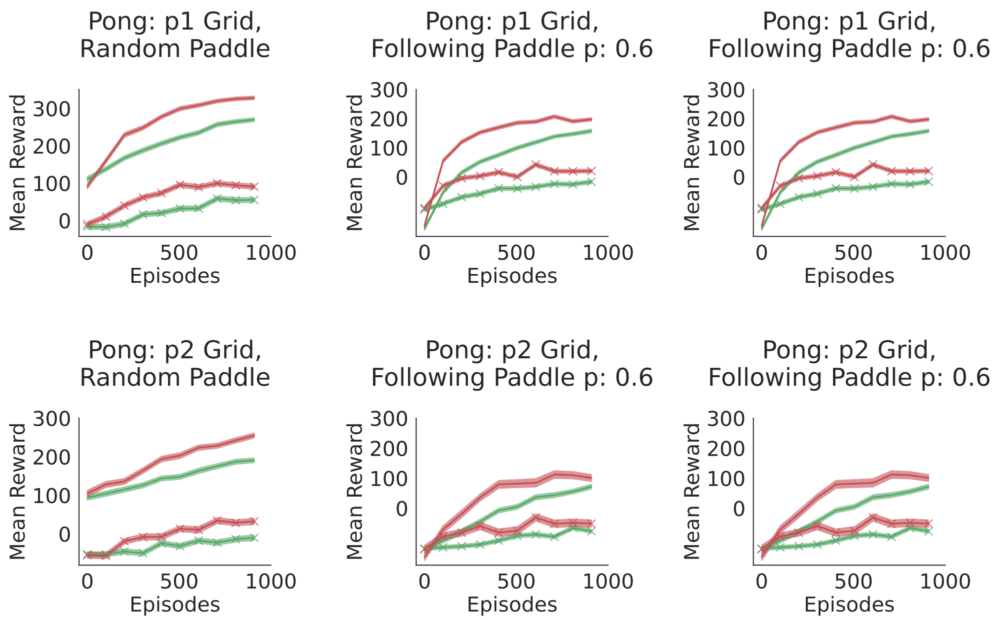


Figure Supp.6: SARSA Agent with Boltzmann exploration strategy: Results for Pong p1, p2 grids reporting mean reward as a function of episode number. The agent is trained on the non-noisy version of the environment and tested on different level of noise ($\partial \sim \mathcal{N}(0, 0.1)$ in Low-Noise and $\partial \sim \mathcal{N}(0, 0.5)$ in High-Noise settings).

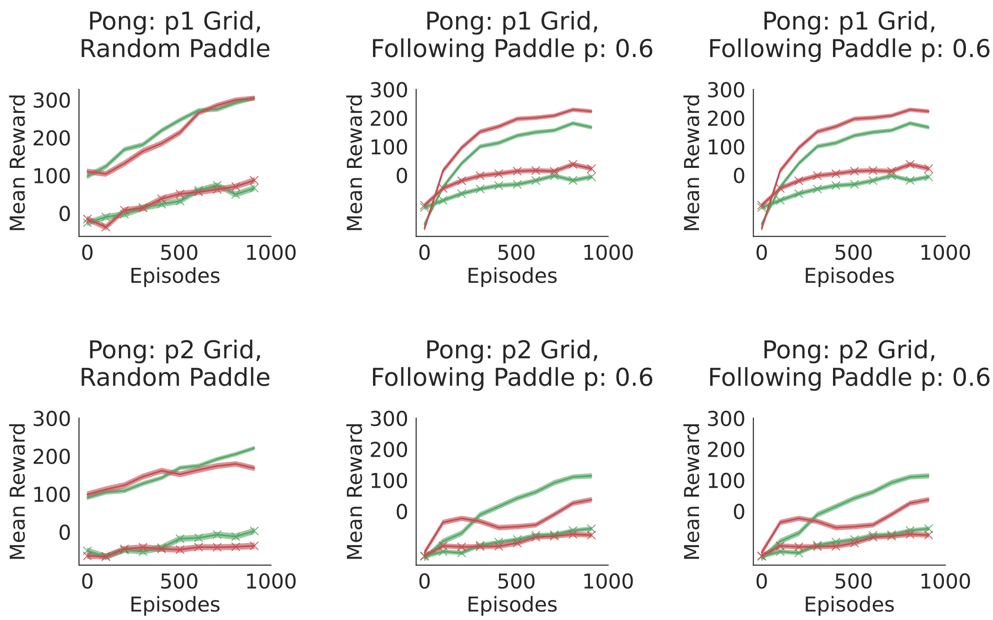


Figure Supp.7: Q-learning Agent with Boltzmann exploration strategy: Results for Pong p1, p2 grids reporting mean reward as a function of episode number. The agent is trained on the non-noisy version of the environment and tested on different level of noise ($\partial \sim \mathcal{N}(0, 0.1)$ in Low-Noise and $\partial \sim \mathcal{N}(0, 0.5)$ in High-Noise settings).

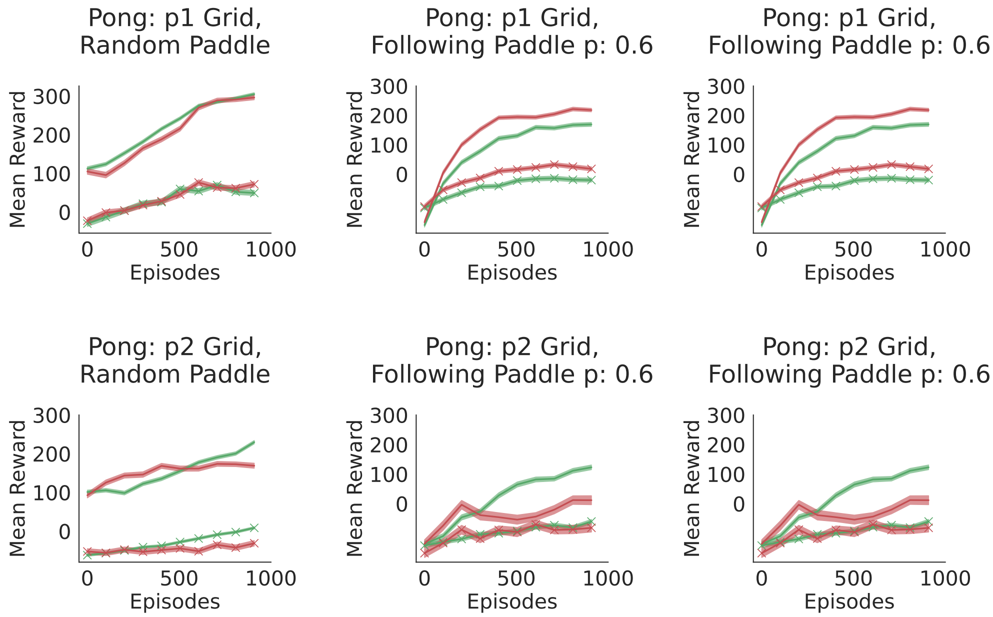


Figure Supp.8: Q-learning Agent with ε-greedy exploration strategy: Results for Pong p1, p2 grids reporting mean reward as a function of episode number. The agent is trained on the non-noisy version of the environment and tested on different level of noise ($\partial \sim \mathcal{N}(0, 0.1)$ in Low-Noise and $\partial \sim \mathcal{N}(0, 0.5)$ in High-Noise settings).

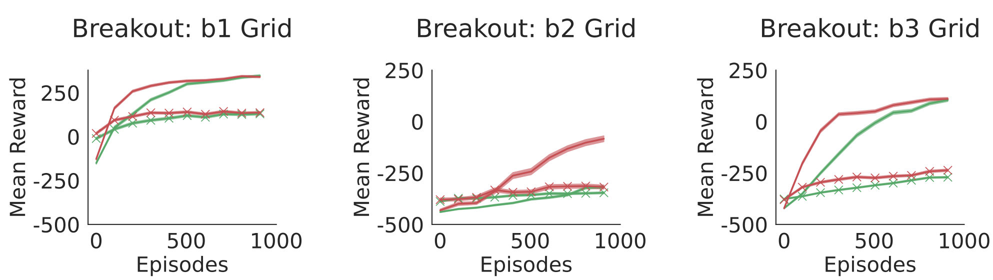


Figure Supp.9: SARSA Agent with Boltzmann exploration strategy: Results for Breakout b1, b2, b3 grids reporting mean reward as a function of episode number. The agent is trained on the non-noisy version of the environment and tested on different level of noise ($\partial \sim \mathcal{N}(0, 0.1)$ in Low-Noise and $\partial \sim \mathcal{N}(0, 0.5)$ in High-Noise settings).

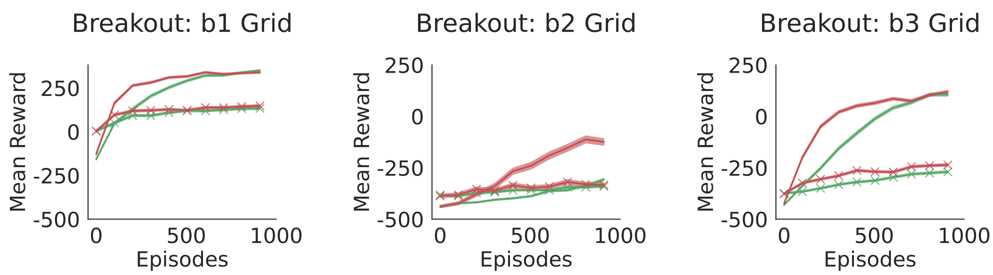


Figure Supp.10: SARSA Agent with ε-greedy exploration strategy: Results for Breakout b1, b2, b3 grids reporting mean reward as a function of episode number. The agent is trained on the non-noisy version of the environment and tested on different level of noise ($\partial \sim \mathcal{N}(0, 0.1)$ in Low-Noise and $\partial \sim \mathcal{N}(0, 0.5)$ in High-Noise settings).

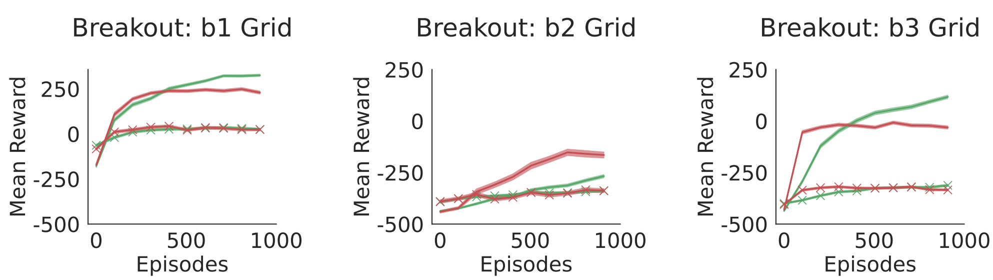


Figure Supp.11: Q-learning Agent with Boltzmann exploration strategy: Results for Breakout b1, b2, b3 grids reporting mean reward as a function of episode number. The agent is trained on the non-noisy version of the environment and tested on different level of noise ($\partial \sim \mathcal{N}(0, 0.1)$ in Low-Noise and $\partial \sim \mathcal{N}(0, 0.5)$ in High-Noise settings).

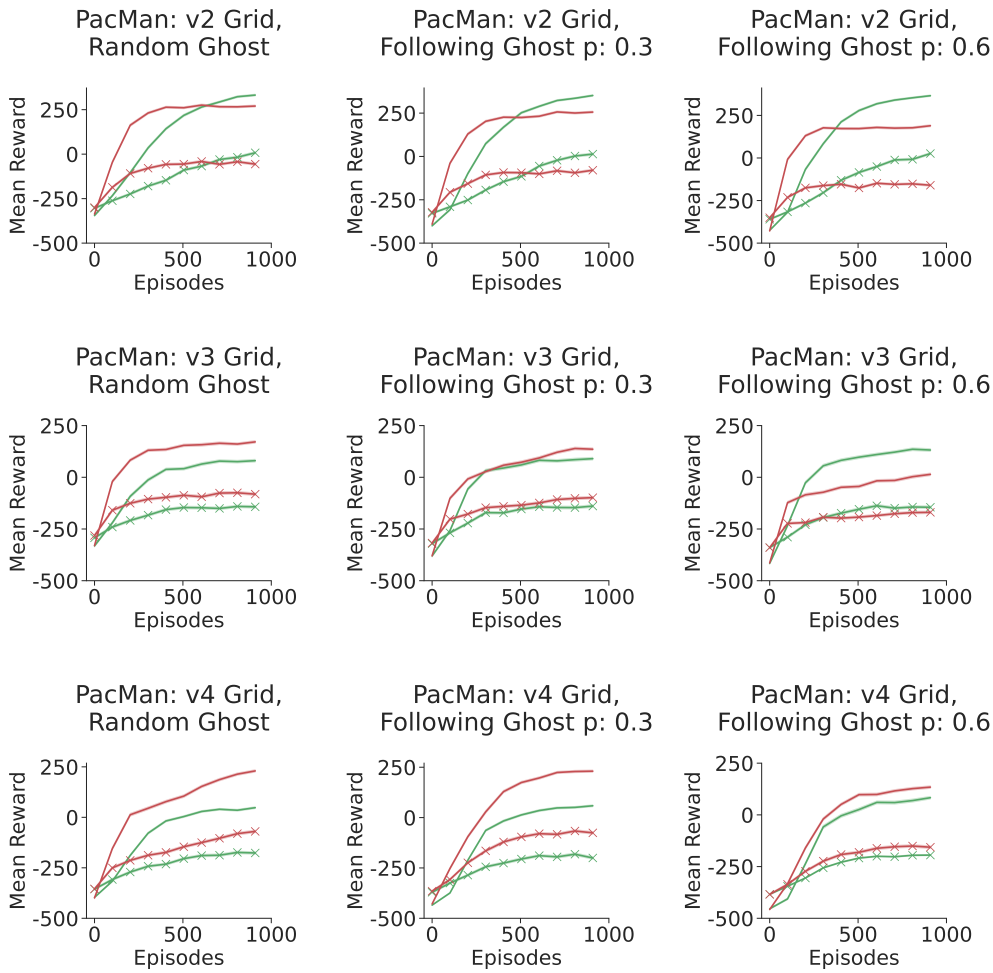


Figure Supp.12: Q-learning Agent with ε-greedy exploration strategy Results for Pong p1, p2 grids reporting mean reward as a function of episode number. The agent is trained on the non-noisy version of the environment and tested on different level of noise ($\delta \sim \mathcal{N}(0, 0.1)$ in Low-Noise and $\delta \sim \mathcal{N}(0, 0.5)$ in High-Noise settings).

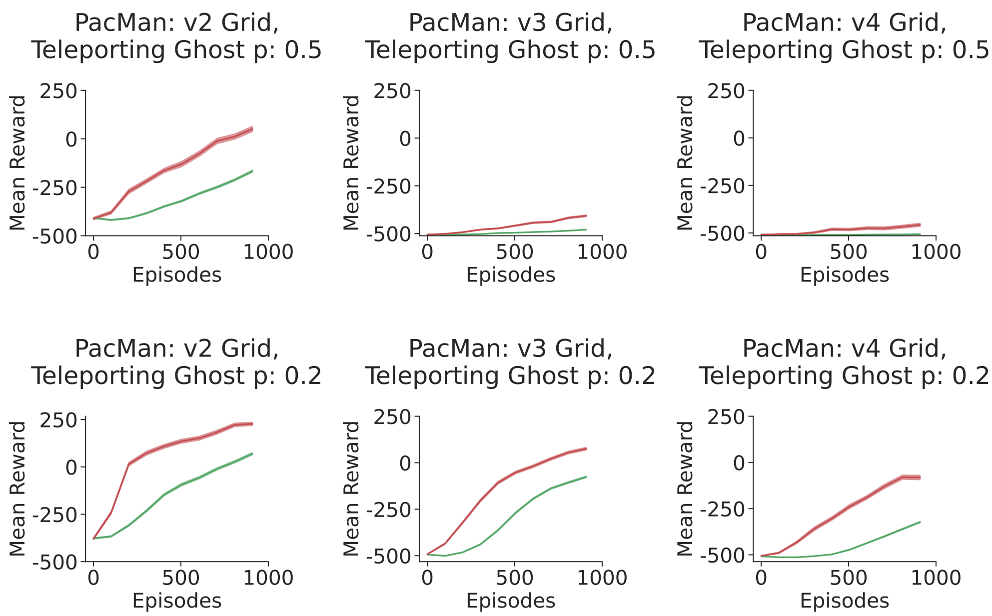


Figure Supp.13: SARSA Agent with Boltzmann exploration strategy: Results for PacMan v2, v3, v4 grids reporting mean reward as a function of episode number. The agent is trained on the Random Ghost environment and tested on the Teleporting Ghost variation ($p = 0.2$, $p = 0.5$)

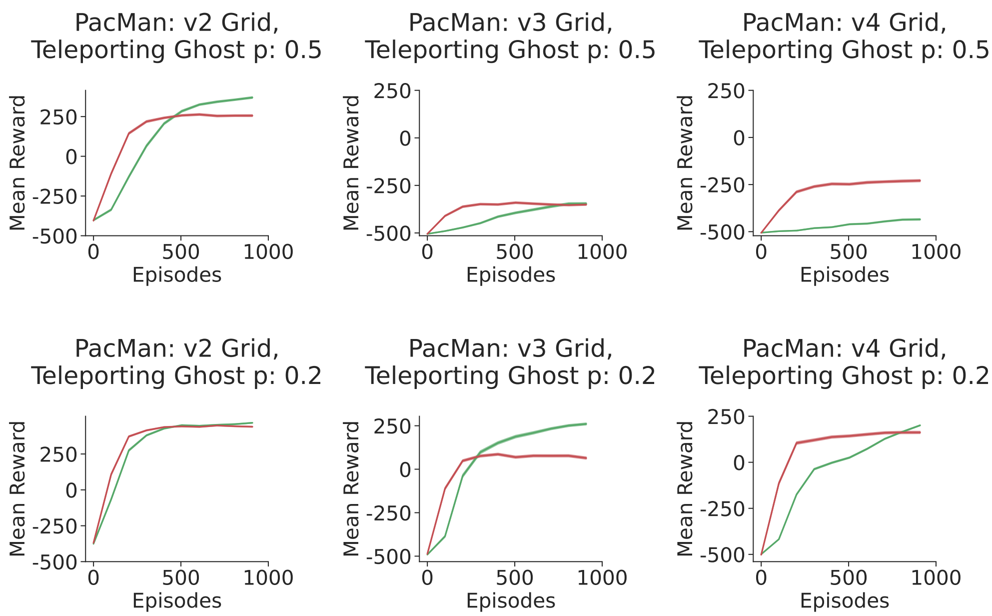


Figure Supp.14: Q-learning Agent with Boltzmann exploration strategy: Results for PacMan v2, v3, v4 grids reporting mean reward as a function of episode number. The agent is trained on the Random Ghost environment and tested on the Teleporting Ghost variation ($p = 0.2$, $p = 0.5$)

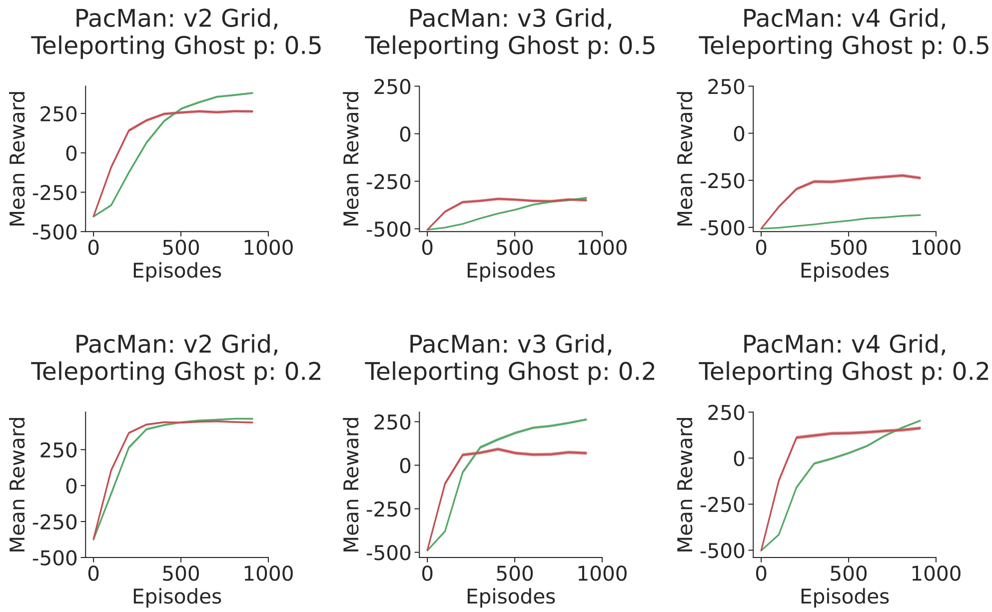


Figure Supp.15: Q-learning Agent with ε-greedy exploration strategy: Results for PacMan v2, v3, v4 grids reporting mean reward as a function of episode number. The agent is trained on the Random Ghost environment and tested on the Teleporting Ghost variation ($p = 0.2$, $p = 0.5$)

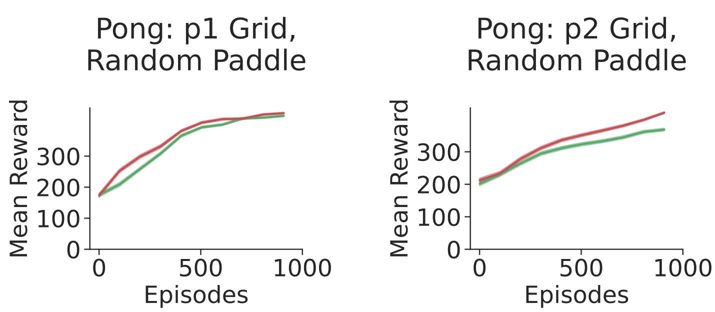


Figure Supp.16: SARSA Agent with Boltzmann exploration strategy: Results for Pong p1, p2 grids reporting mean reward as a function of episode number. The agent is trained on the Directional Ghost ($p = 0.3$) environment and tested on the Random Ghost variation.

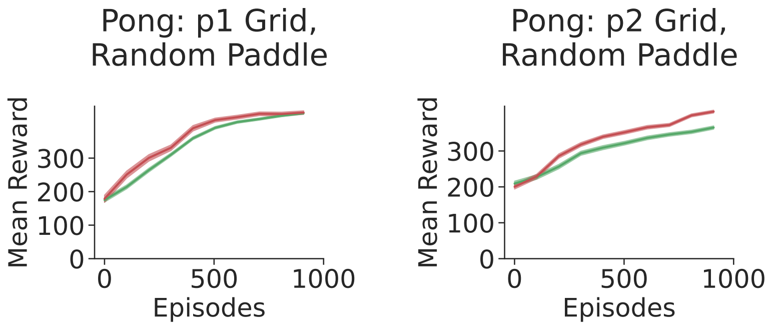


Figure Supp.17: SARSA Agent with ε-greedy exploration strategy: Results for Pong p1, p2 grids reporting mean reward as a function of episode number. The agent is trained on the Directional Ghost ($p = 0.3$) environment and tested on the Random Ghost variation.

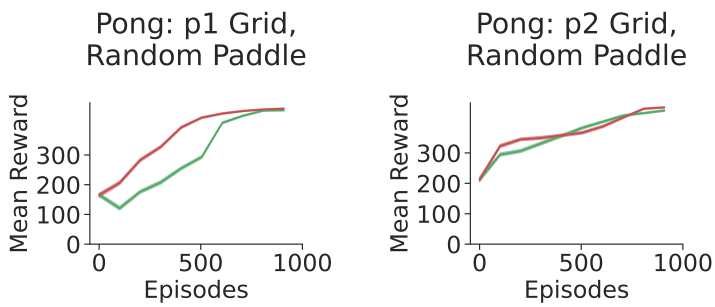


Figure Supp.18: Q-learning Agent with Boltzmann exploration strategy: Results for Pong p1, p2 grids reporting mean reward as a function of episode number. The agent is trained on the Directional Ghost ($p = 0.3$) environment and tested on the Random Ghost variation.

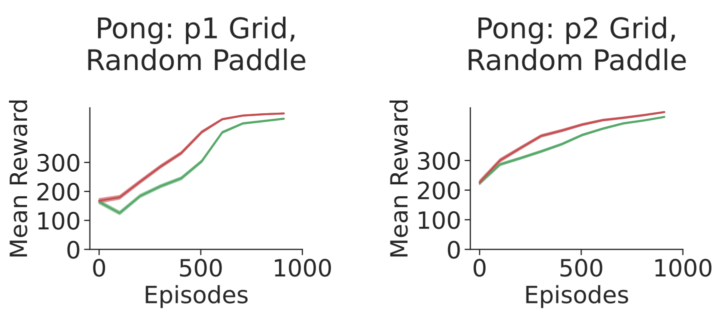


Figure Supp.19: Q-learning Agent with ε-greedy exploration strategy: Results for Pong p1, p2 grids reporting mean reward as a function of episode number. The agent is trained on the Directional Ghost ($p = 0.3$) environment and tested on the Random Ghost variation.

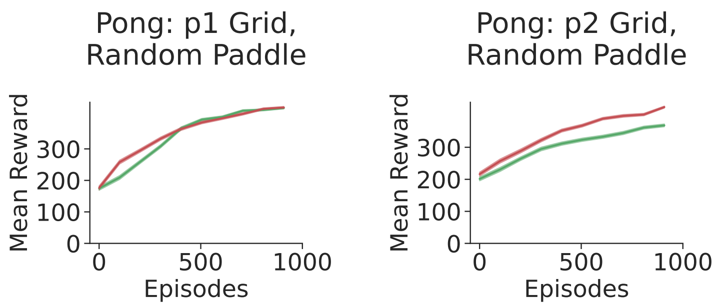


Figure Supp.20: SARSA Agent with Boltzmann exploration strategy: Results for Pong p1, p2 grids reporting mean reward as a function of episode number. The agent is trained on the Directional Ghost ($p = 0.3$) environment and tested on the Random Ghost variation.

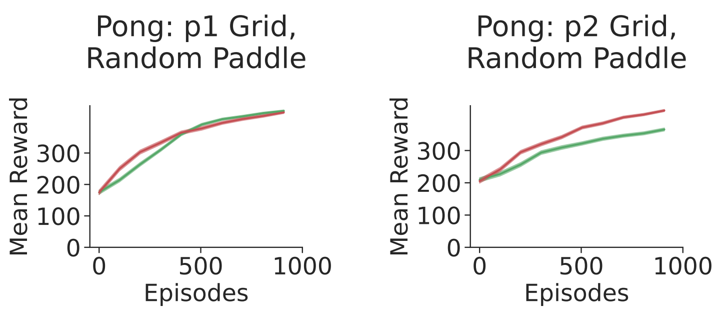


Figure Supp.21: SARSA Agent with ε-greedy exploration strategy: Results for Pong p1, p2 grids reporting mean reward as a function of episode number. The agent is trained on the Directional Ghost ($p = 0.3$) environment and tested on the Random Ghost variation.

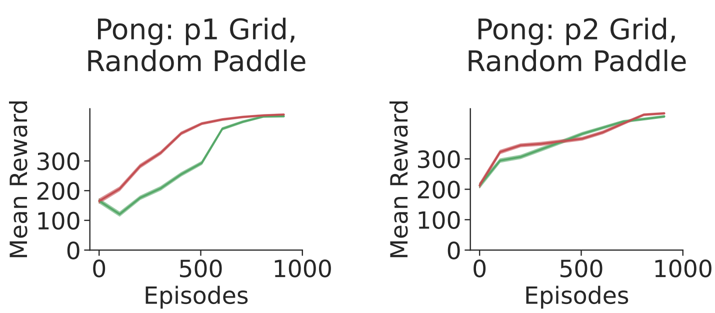


Figure Supp.22: Q-learning Agent with Boltzmann exploration strategy: Results for Pong p1, p2 grids reporting mean reward as a function of episode number. The agent is trained on the Directional Ghost ($p = 0.3$) environment and tested on the Random Ghost variation.

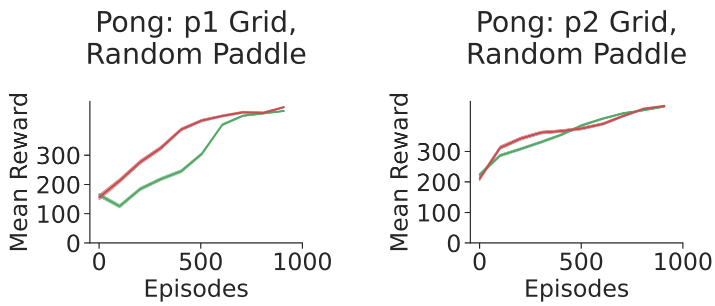
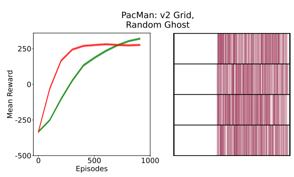
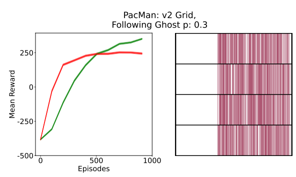


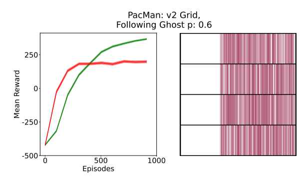
Figure Supp.23: Q-learning Agent with ε-greedy exploration strategy: Results for Pong p1, p2 grids reporting mean reward as a function of episode number. The agent is trained on the Directional Ghost ($p = 0.3$) environment and tested on the Random Ghost variation.



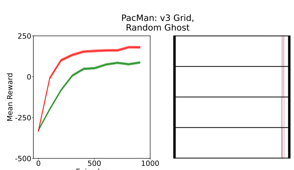
(a)



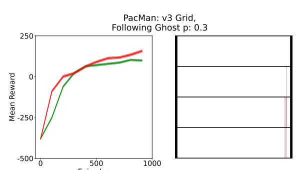
(b)



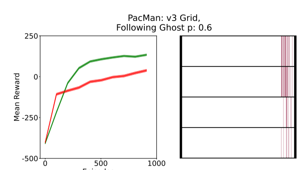
(c)



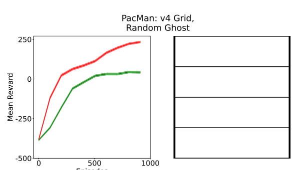
(d)



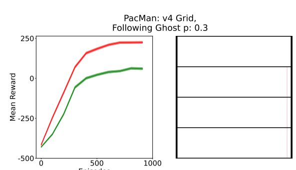
(e)



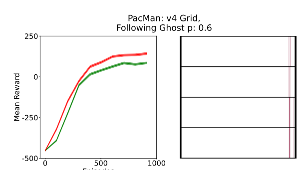
(f)



(g)



(h)



(i)

Figure Supp.24: Q-learning Agent with Boltzmann exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for PacMan v2, v3, v4 grids, the agent is trained on non-noisy variations of different environments (reported in the headings) and tested in the Low-Noise regime. Rows in the right figure represents agent's actions Left, Right, Up, or Down.

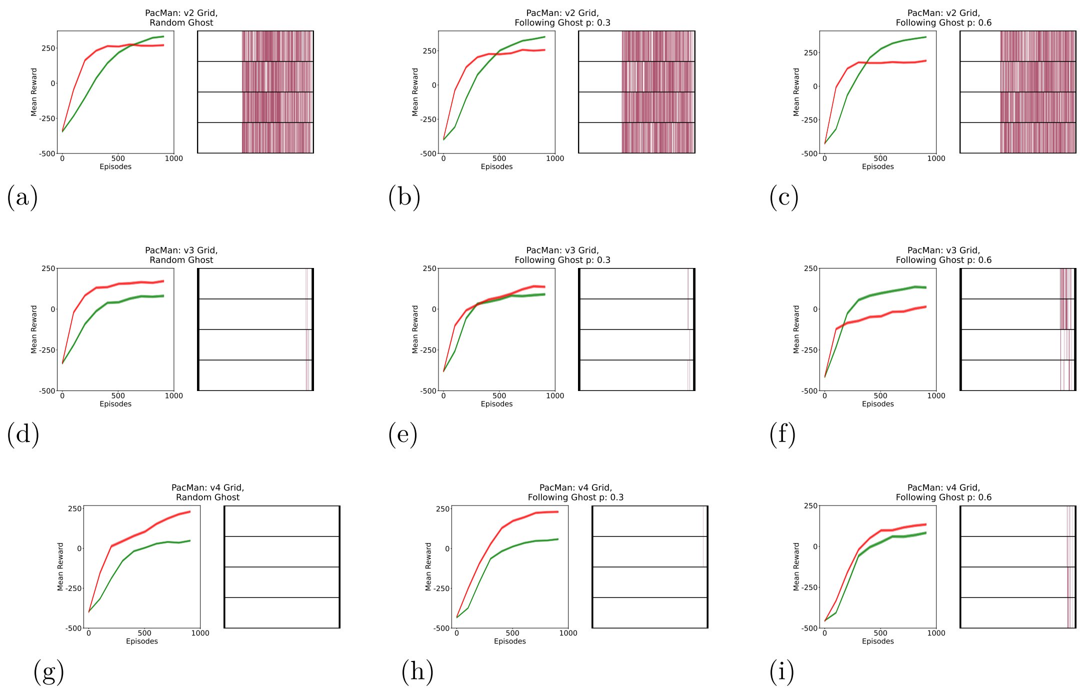


Figure Supp.25: Q-learning Agent with ε-greedy exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for PacMan v2, v3, v4 grids, the agent is trained on non-noisy variations of different environments (reported in the headings) and tested in the Low-Noise regime. Rows in the right figure represents agent's actions Left, Right, Up, or Down.

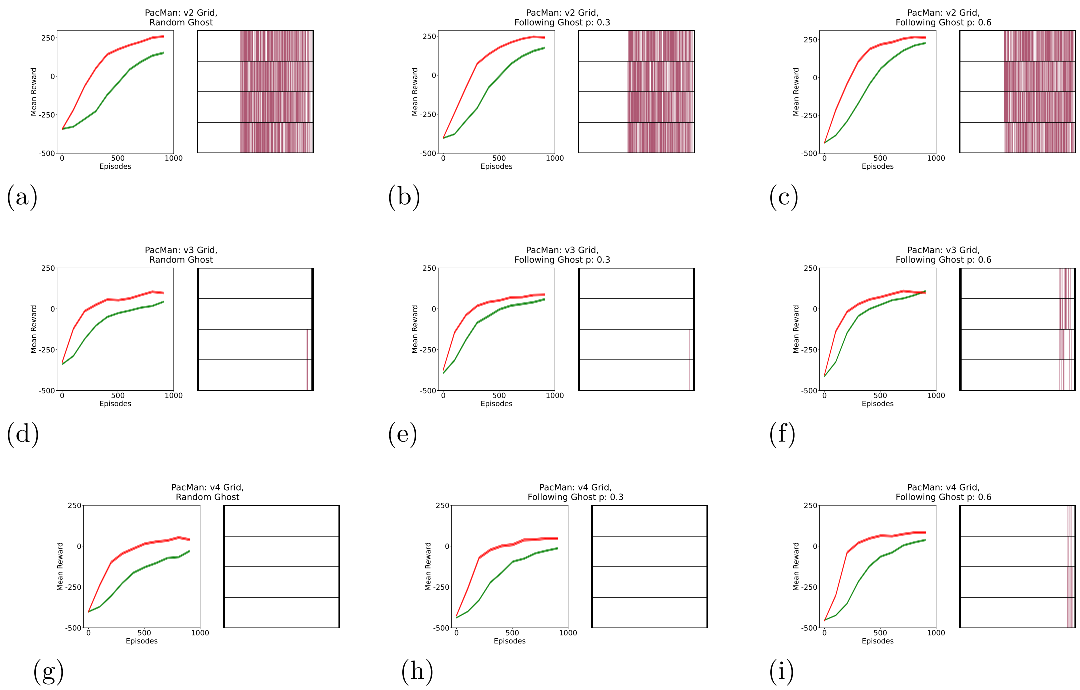
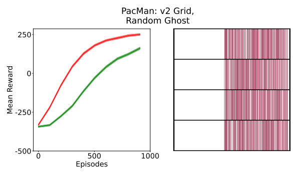
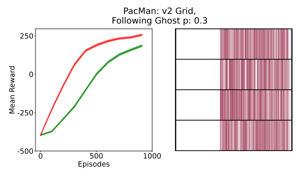


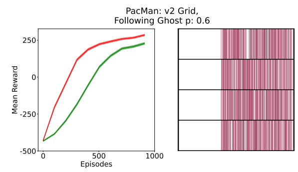
Figure Supp.26: SARSA Agent with Boltzmann exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for PacMan v2, v3, v4 grids, the agent is trained on non-noisy variations of different environments (reported in the headings) and tested in the Low-Noise regime. Rows in the right figure represents agent's actions Left, Right, Up, or Down.



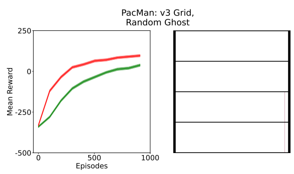
(a)



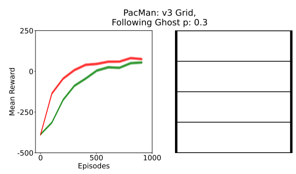
(b)



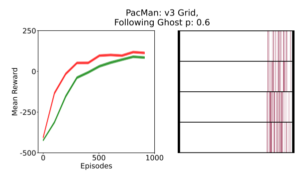
(c)



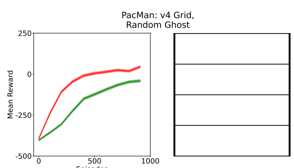
(d)



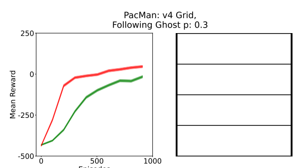
(e)



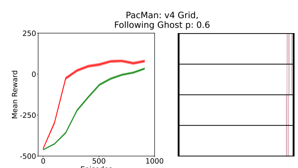
(f)



(g)



(h)



(i)

Figure Supp.27: SARSA Agent with ε-greedy exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for PacMan v2, v3, v4 grids, the agent is trained on non-noisy variations of different environments (reported in the headings) and tested in the Low-Noise regime. Rows in the right figure represents agent's actions Left, Right, Up, or Down.

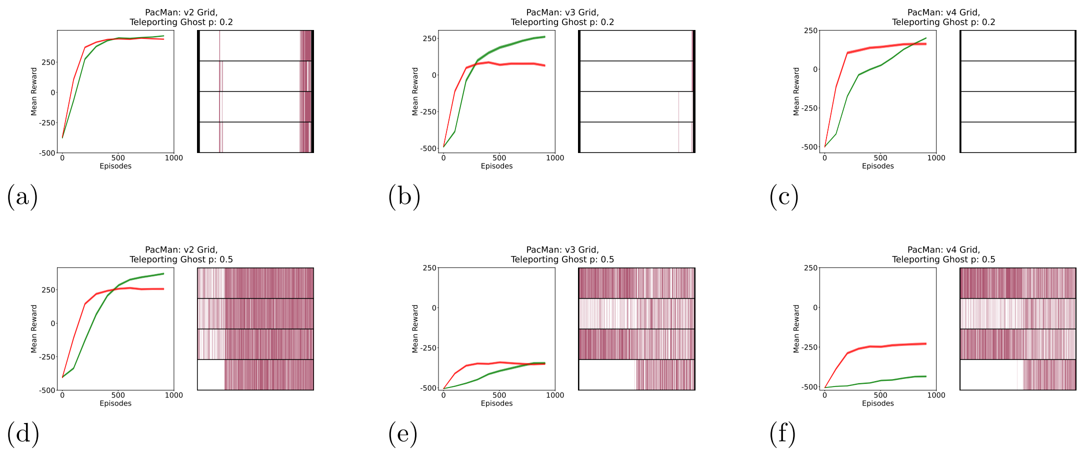


Figure Supp.28: Q-learning Agent with Boltzmann exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for PacMan v2, v3, v4 grids, the agent is trained on Teleporting Ghost variation ($p = 0.2$, $p = 0.5$) and tested in different environments (reported in the headings). Rows in the right figure represents agent's actions Left, Right, Up, or Down.

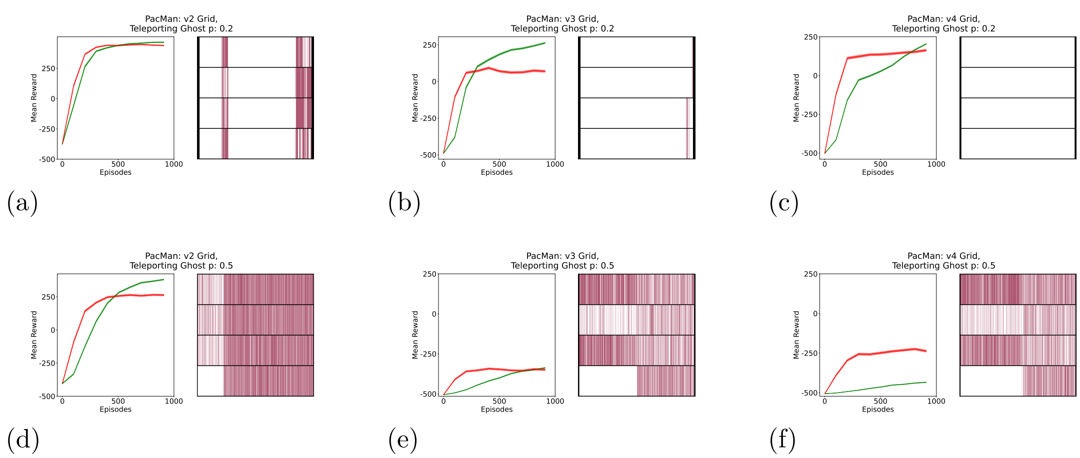
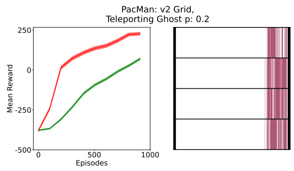
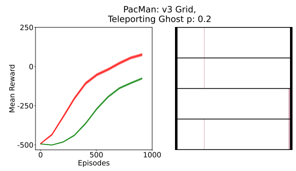


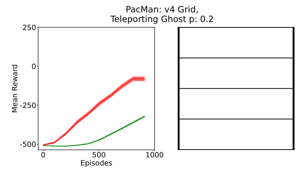
Figure Supp.29: Q-learning Agent with ε-greedy exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for PacMan v2, v3, v4 grids, the agent is trained on Teleporting Ghost variation ($p = 0.2$, $p = 0.5$) and tested in different environments (reported in the headings). Rows in the right figure represents agent's actions Left, Right, Up, or Down.



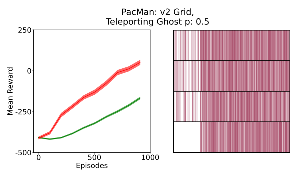
(a)



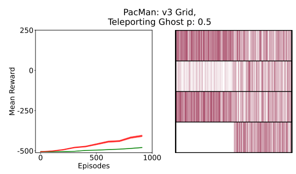
(b)



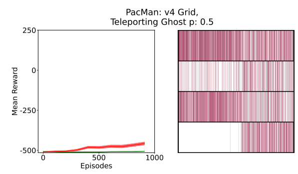
(c)



(d)

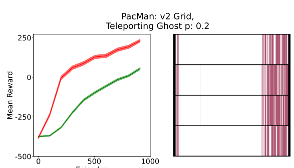


(e)

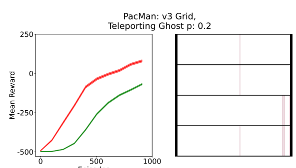


(f)

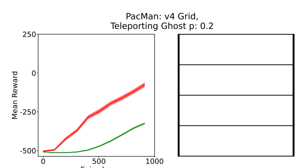
Figure Supp.30: SARSA Agent with Boltzmann exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for PacMan v2, v3, v4 grids, the agent is trained on Teleporting Ghost variation ($p = 0.2$, $p = 0.5$) and tested in different environments (reported in the headings). Rows in the right figure represents agent's actions Left, Right, Up, or Down.



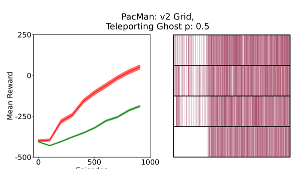
(a)



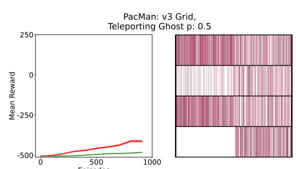
(b)



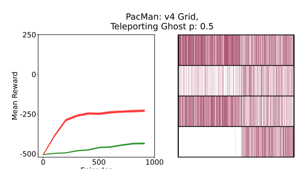
(c)



(d)



(e)



(f)

Figure Supp.31: SARSA Agent with ε-greedy exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for PacMan v2, v3, v4 grids, the agent is trained on Teleporting Ghost variation ($p = 0.2$, $p = 0.5$) and tested in different environments (reported in the headings). Rows in the right figure represents agent's actions Left, Right, Up, or Down.

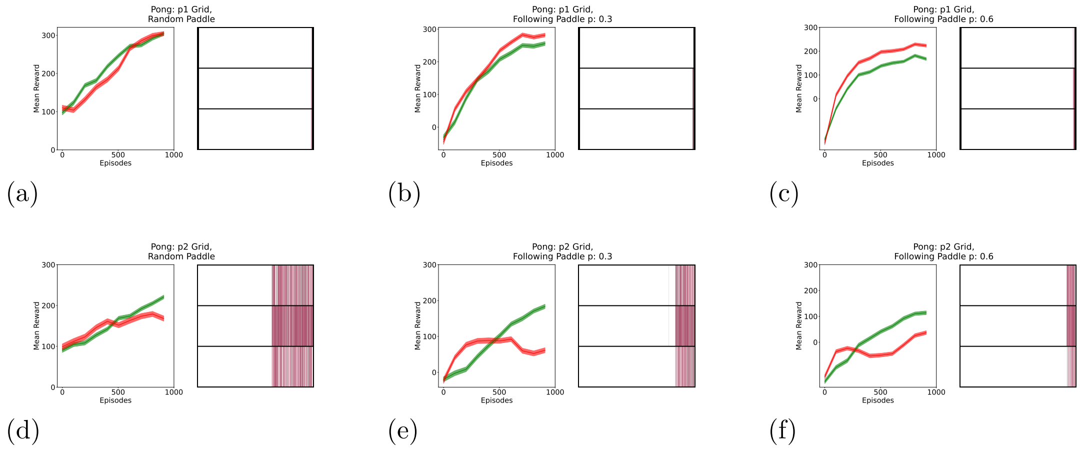


Figure Supp.32: Q-learning Agent with Boltzmann exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for Pong p1, p2 grids, the agent is trained on non-noisy variations of different environments (reported in the headings) and tested in the Low-Noise regime. Rows in the right figure represents agent's actions Left, Right, Stop.

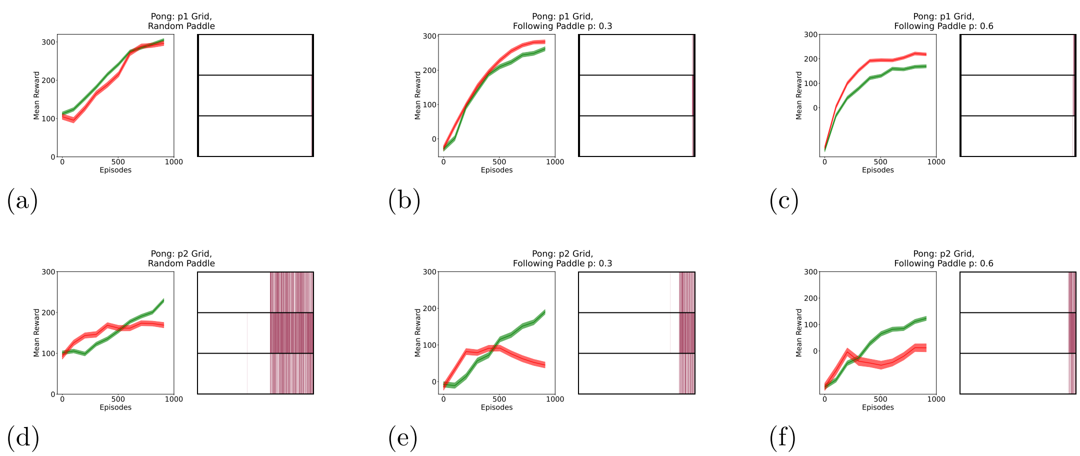


Figure Supp.33: Q-learning Agent with ε-greedy exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for Pong p1, p2 grids, the agent is trained on non-noisy variations of different environments (reported in the headings) and tested in the Low-Noise regime. Rows in the right figure represents agent's actions Left, Right, Stop.

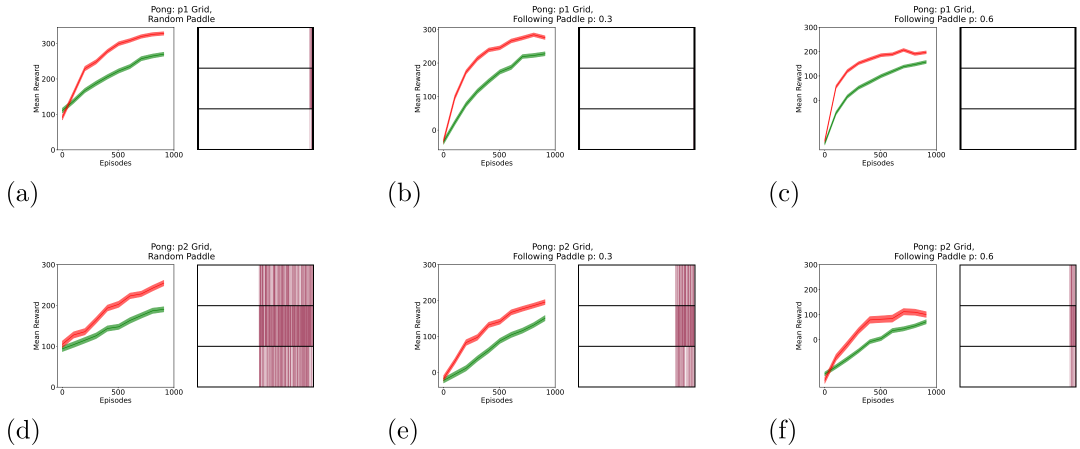


Figure Supp.34: SARSA Agent with Boltzmann exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for Pong p1, p2 grids, the agent is trained on non-noisy variations of different environments (reported in the headings) and tested in the Low-Noise regime. Rows in the right figure represents agent's actions Left, Right, Stop.

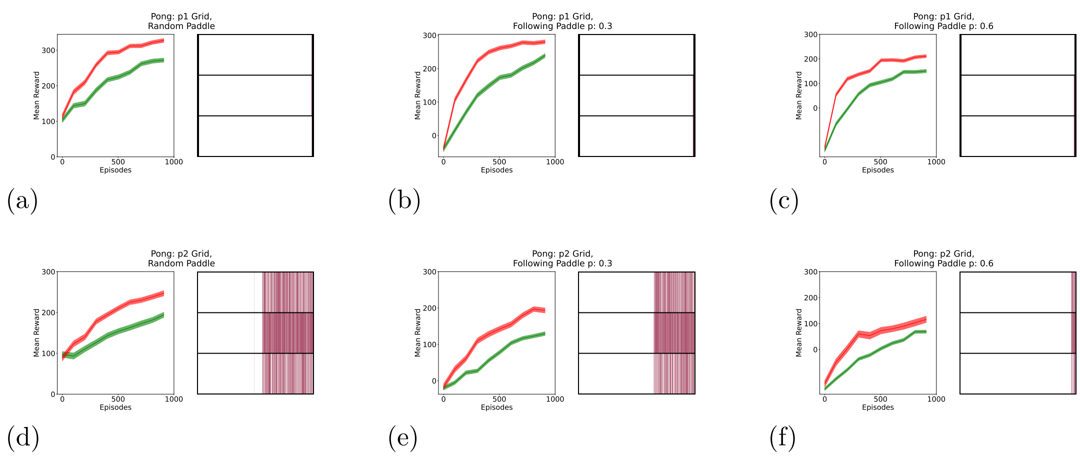
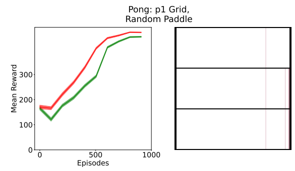
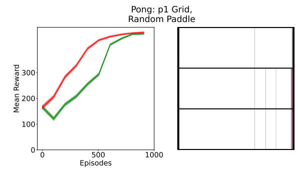


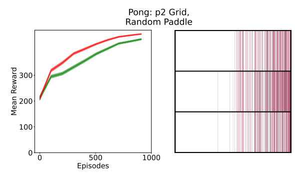
Figure Supp.35: SARSA Agent with ε-greedy exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for Pong p1, p2 grids, the agent is trained on non-noisy variations of different environments (reported in the headings) and tested in the Low-Noise regime. Rows in the right figure represents agent's actions Left, Right, Stop.



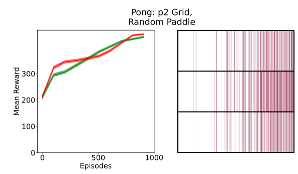
(a)



(b)

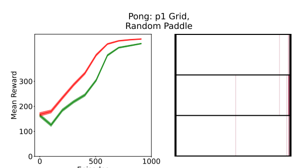


(c)

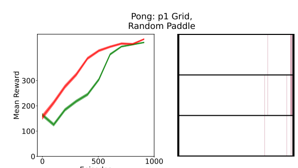


(d)

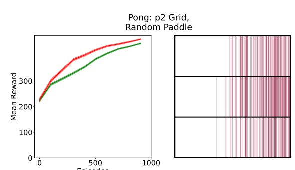
Figure Supp.36: Q-learning Agent with Boltzmann exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for Pong p1, p2 grids, the agent is trained on Directional paddle ($p = 0.3$ top, $p = 0.6$, bottom) variation and tested in the Random paddle environment. Rows in the right figure represents agent's actions Left, Right, Stop.



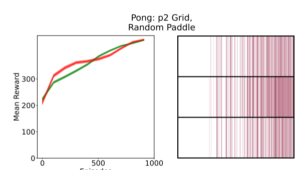
(a)



(b)

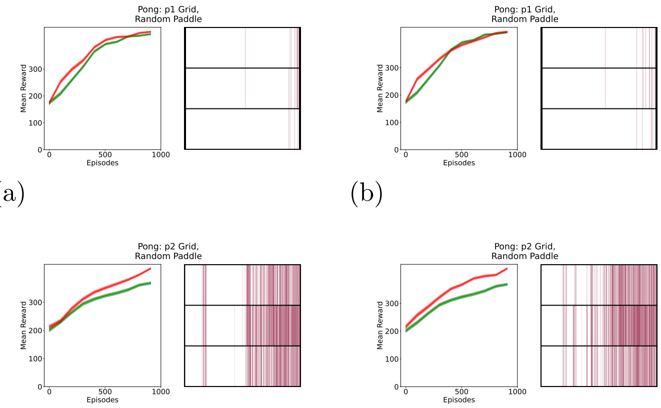


(c)



(d)

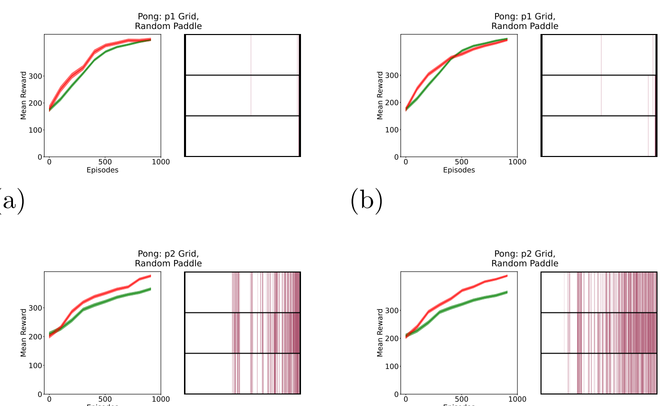
Figure Supp.37: Q-learning Agent with ε-greedy exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for Pong p1, p2 grids, the agent is trained on Directional Paddle ($p = 0.3$ top, $p = 0.6$, bottom) variation and tested in the Random Paddle environment. Rows in the right figure represents agent's actions Left, Right, Stop.



(a) (b)

(c) (d)

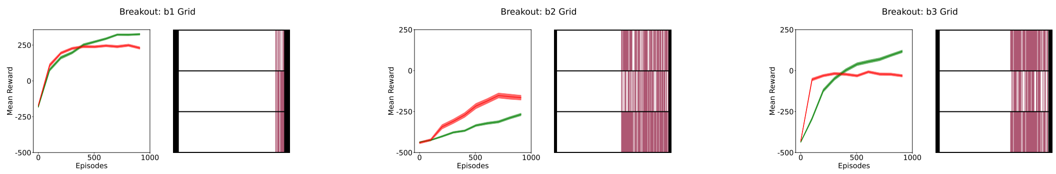
Figure Supp.38: SARSA Agent with Boltzmann exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for Pong p1, p2 grids, the agent is trained on Directional Paddle ($p = 0.3$ top, $p = 0.6$, bottom) variation and tested in the Random Paddle environment. Rows in the right figure represents agent's actions Left, Right, Stop.



(a) (b)

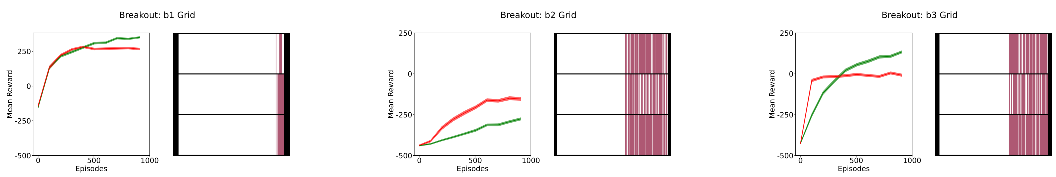
(c) (d)

Figure Supp.39: SARSA Agent with ε-greedy exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for Pong p1, p2 grids, the agent is trained on Directional Paddle ($p = 0.3$ top, $p = 0.6$, bottom) variation and tested in the Random Paddle environment. Rows in the right figure represents agent's actions Left, Right, Stop.



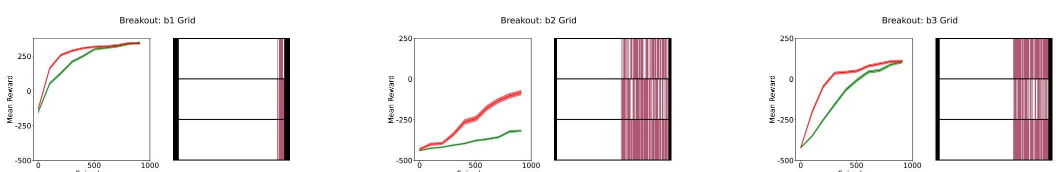
(a) (b) (c)

Figure Supp.40: Q-learning Agent with Boltzmann exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for Breakout b1, b2, b3 grids, the agent is trained on non-noisy variations of different environments (reported in the headings) and tested in the Low-Noise regime. Rows in the right figure represents agent's actions Left, Right, Stop.



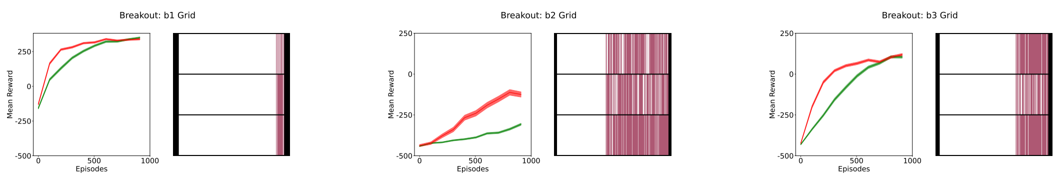
(a) (b) (c)

Figure Supp.41: Q-learning Agent with ε-greedy exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for Breakout b1, b2, b3 grids, the agent is trained on non-noisy variations of different environments (reported in the headings) and tested in the Low-Noise regime. Rows in the right figure represents agent's actions Left, Right, Stop.



(a) (b) (c)

Figure Supp.42: SARSA Agent with Boltzmann exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for Breakout b1, b2, b3 grids, the agent is trained on non-noisy variations of different environments (reported in the headings) and tested in the Low-Noise regime. Rows in the right figure represents agent's actions Left, Right, Stop.



(a) (b) (c)

Figure Supp.43: SARSA Agent with ε-greedy exploration strategy: The exploration grid visualizing the difference in State-Action (S-A) pairs explored by these agents (D_{LG}). Results for Breakout b1, b2, b3 grids, the agent is trained on non-noisy variations of different environments (reported in the headings) and tested in the Low-Noise regime. Rows in the right figure represents agent's actions Left, Right, Stop.

Figure Supp.44: Sample Images from dataset

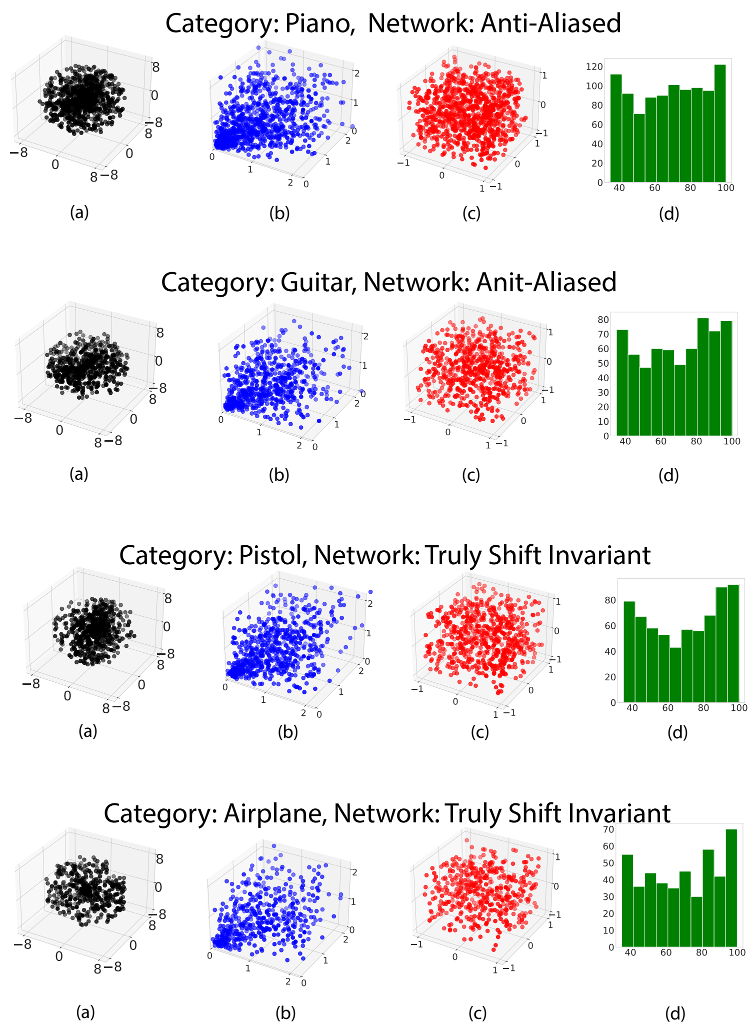


Figure Supp.45: Camera Parameters that lead to misclassifications for multiple categories and architectures. (a) Camera Position, (b) Camera Look At, (c) Up Vector, (d) Histogram of Lens Field of View.

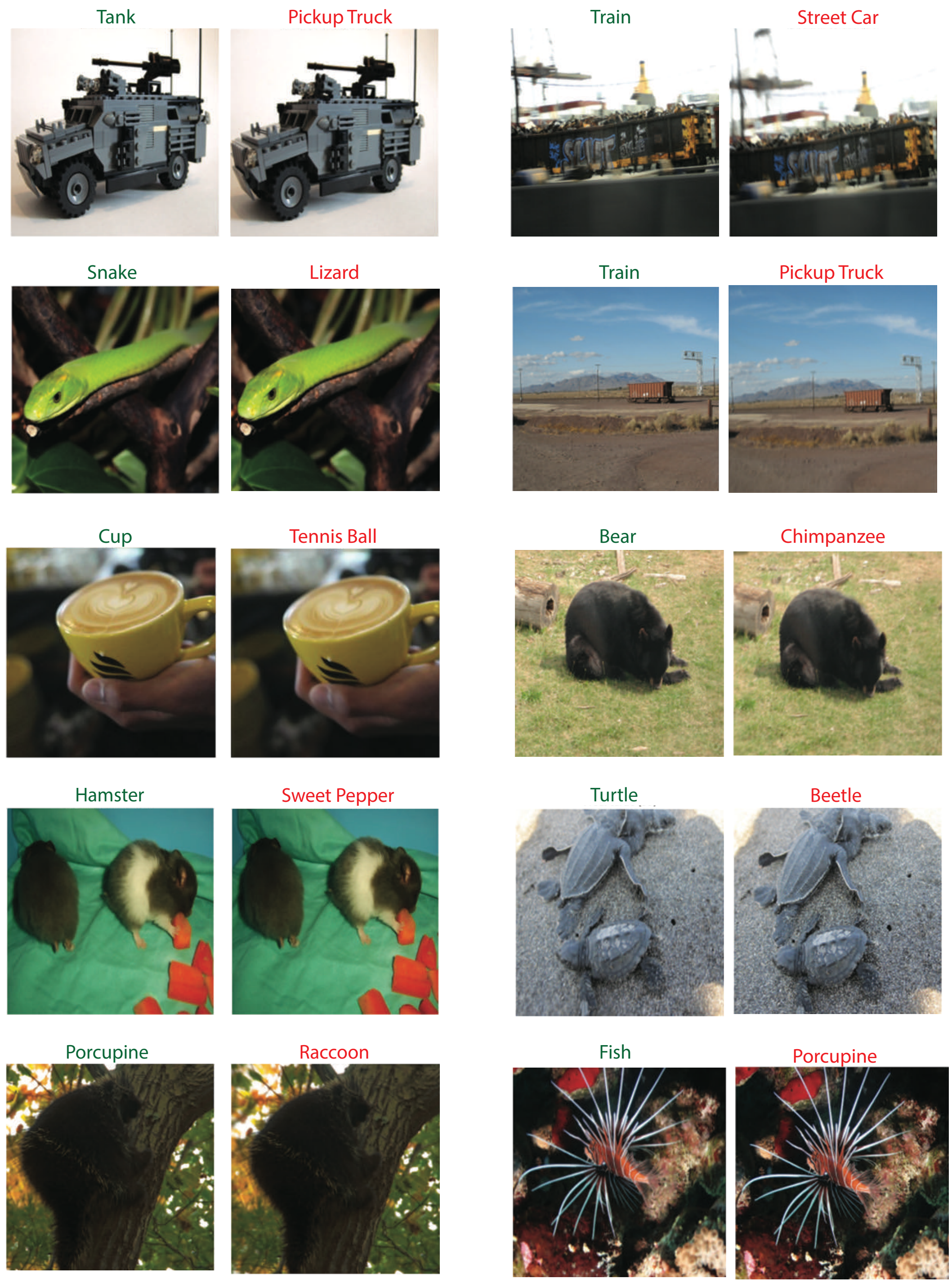


Figure Supp.46: More examples of misclassified ImageNet-like images discovered by CMA-Search combined with the single view MPI model.

S21 Synthetic Out-of-context Dataset (OCD)

S21.1 Environment setup for various contextual conditions:

We leveraged the VirtualHome environment[295] developed in the Unity simulation engine to synthesize images in indoor home environments within 7 apartments and 5 rooms per apartment. A maximum of 9 fixed view angles are captured for each target object and location. Azimuth angles range from 0 to 320 degrees in steps of 40 degrees, fixed elevation angle of 19.5 degrees and radius of 1.5 meters. In some special cases, the elevation angle is set to -19.5 degrees to prevent the camera location from penetrating the ceiling.

In the gravity condition, the elevation angle of the camera is adjusted to center the view on the floating target object. This causes the surrounding context to vary slightly compared with the same configuration in the normal context condition. The small difference in the number of images between the normal context and the gravity condition is due to the removal of some images because of invalid camera positions or target collisions with other objects after the targets are lifted up. In other out-of-context conditions, same object and camera configurations remain as the normal context conditions.

S21.2 Performance Evaluation

The object-to-context ratio is critical[437] — context plays a larger role for smaller objects. Therefore, we split images into two groups: according to the target object sizes in degrees of visual angle (dva), based on the psychophysics experiments (Figure 3d)): (i) images ≤ 2 dva, and (ii) images > 2 dva. The pixel to dva conversion is based on the human experiment setups of a display with 1024×1280 pixels and distance of 0.5 meters (actual size varies in MTurk depending on viewing conditions).

S21.3 Training on Synthetic (OCD) Data

If we train the models with natural images from COCO-Stuff[48] and then evaluate them on synthetic images from our OCD dataset, we face a domain gap. This domain gap might play a role in influencing the comparison of the same model across different context conditions and between different models. To evaluate the ability of closing the domain gap for our CRTNet model, we first train CRTNet on the synthesized training set in normal context conditions, and then we test CRTNet in the normal condition in the test set as elaborated in the main paper.

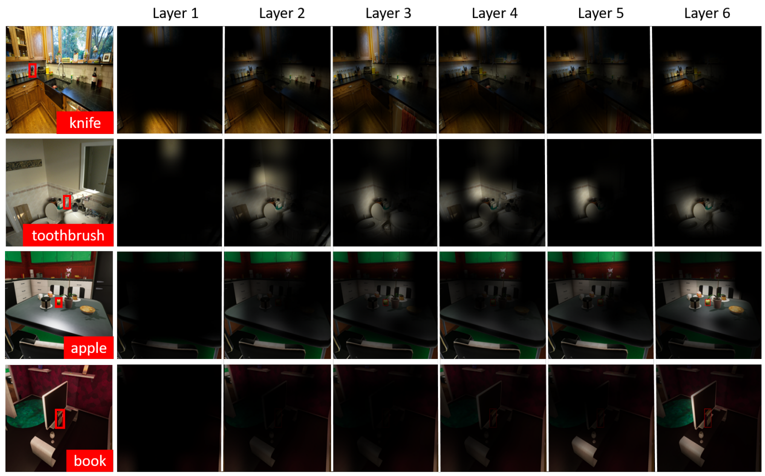
To synthesize the training set in normal context conditions, for each object belonging to the 36 object classes, its possible locations are uniformly arranged in a grid over the entire supporting surfaces where these surfaces are typically determined according to the co-occurrence statistics. The grid size is chosen relative to the target object size (5 times the target object size) because a small location shift introduces very little view variance for a larger object, e.g., , the view difference by moving a microwave 0.2 meters away is small compared with a cellphone moved by 0.2 meters. For each target object location, we randomly sampled 12 camera views with the following parameters: azimuth angle from [0, 360] degrees with a step size of 2 deg, elevation angle from [-35, 90] degrees with a step size of 2 degrees, and radius of [0.5, 5] meters with a step size of 0.5 meters. To introduce variations and avoid overfitting during training, we replace the target object's texture with a random texture from[333]. Collision checking and camera ray casting are enabled to prevent object collisions and occlusions. We used 5 out of the 7 VirtualHome apartment scenes as training set and the images from the remaining two apartments as validation set. CRTNet achieves an accuracy as high as 80% in the normal context condition on our test set as elaborated in the paper, compared to 88% accuracy on the validation set. This implies that our CRTNet is capable of closing the domain gap.

S22 Cut-and-paste Dataset

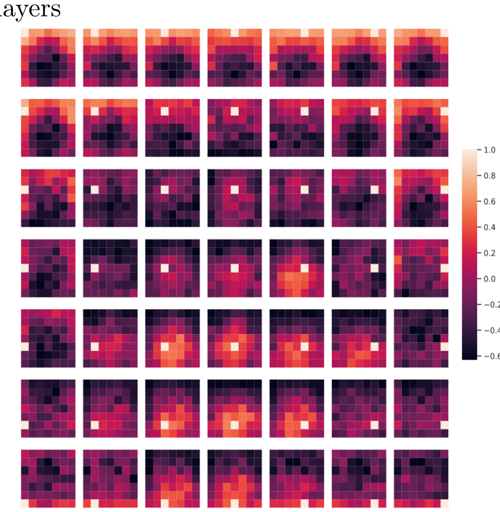
The Cut-and-paste dataset[437] is based on images from the COCO dataset[233]. In addition to normal context and minimal context (rectangular bounding box enclosing the object and grey pixels outside the box) conditions, the target objects were cut from a given image and pasted onto another one with either a congruent context (context contains an object of the same class label) or incongruent context (context taken from an image with different class label). The images are grouped into four bins based on the target object sizes in degrees of visual angle (dva): Size 1 dva [16-32 pixels], Size 2 dva [56-72], Size 4 dva [112-144], and Size 8 dva [224-288].

S23 Visualization of Attention Maps

Visualizations of attention maps on example images (Supp Fig. S1a) show that CRTNet globally attends to image regions that are semantically relevant for classification and that it narrows its focus as the information progresses over the hierarchy of the network layers. For example, in row 1 where the target object is a knife, the attention map starts from the kitchen floor in layer 1, slowly expands to table surfaces and eventually narrows down on the knife's location.



(a) Visualization of attention maps over hierarchical transformer decoding layers



(b) Similarity of position embeddings of CRTNet

Figure Supp.47: CRTNet predicts meaningful attention maps and learns reasonable positional embeddings for feature tokens. (a) Visualization of attention maps on four example images (one example per row). The ground truth label of the target object (red box, column 1) is in the bottom right. Over six transformer decoding layers (6 columns), we show the attention map averaged over all attention heads within the same layer and overlaid the attention map on the original image. The two top rows show examples from the test set of COCO-Stuff dataset[48] and the two bottom rows show examples from the test set of our OCD dataset. (b) Each tile shows the cosine similarity between the position embeddings of the patch with the indicated row and column and the position embeddings of all other patches. See color bar on the right for cosine similarity values.

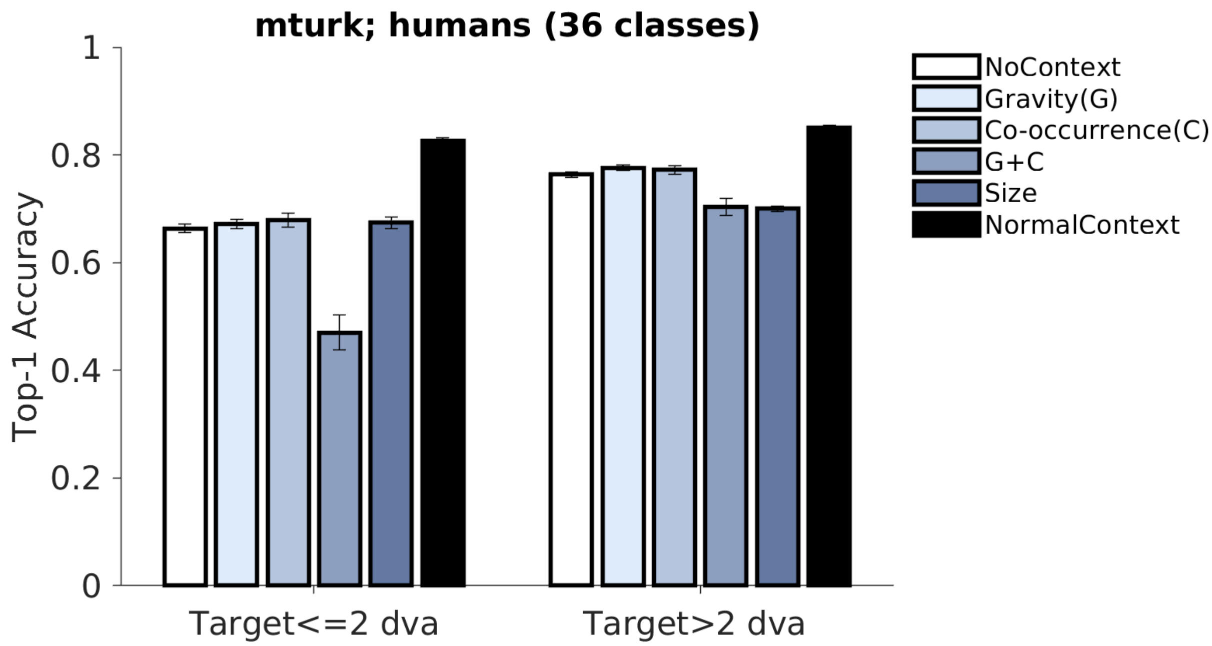


Figure Supp.48: Human performance across context conditions for 33 object classes. Different colors denote context conditions (Sec. 4.2.1, Fig. 1). The trials are divided into two groups based on target object sizes in degrees of visual angle (dva). Error bars denote standard errors of the mean (SEM). This figure expands the results shown in the main text, which corresponds to only the 16 classes that overlap with COCO-Stuff.

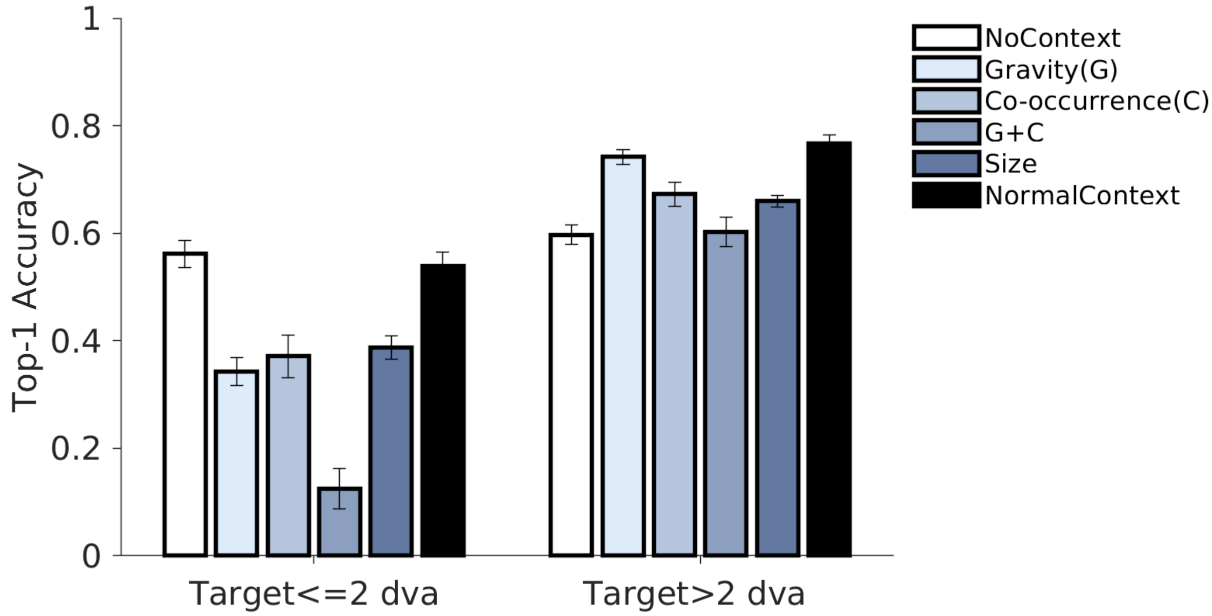


Figure Supp.49: Ablation - shared encoder. In this ablation, we enforced weight sharing between the two encoders $E_t(\cdot)$ and $E_c(\cdot)$. Different colors denote context conditions (Sec. 4.2.1, Fig. 1). Conventions and format follow Figure Supp.48.

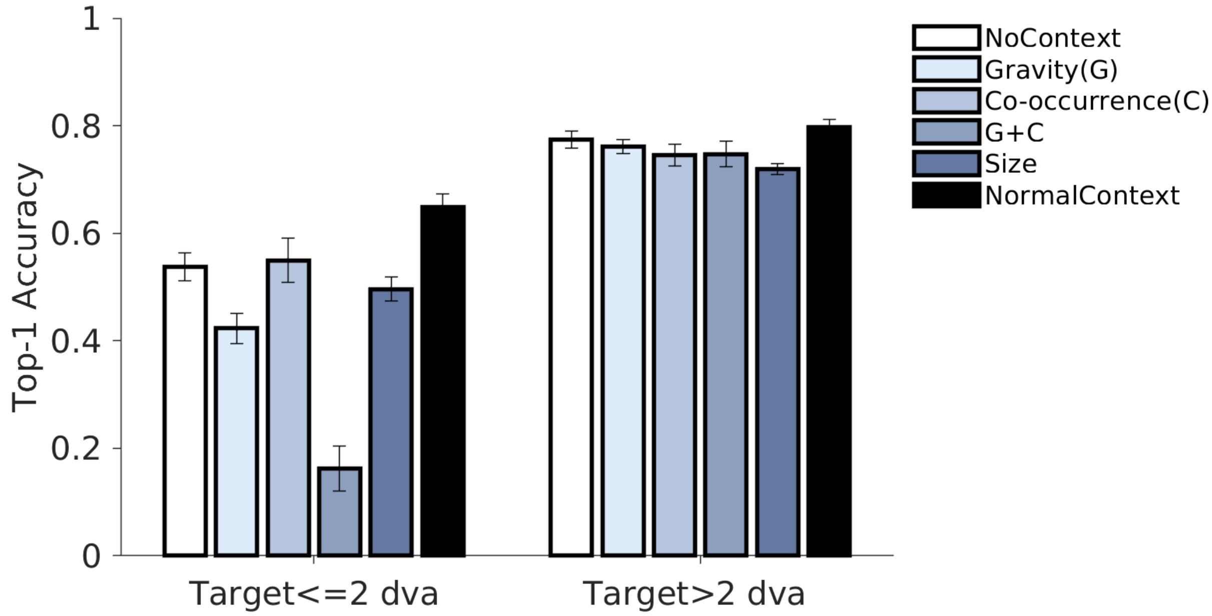


Figure Supp.50: Ablation - target object only. In this ablation, we used y_t (based only on target information) instead of y_p as the final prediction. Different colors denote context conditions (Sec. 4.2.1, Fig. 1). Conventions and format follow Figure Supp.48.

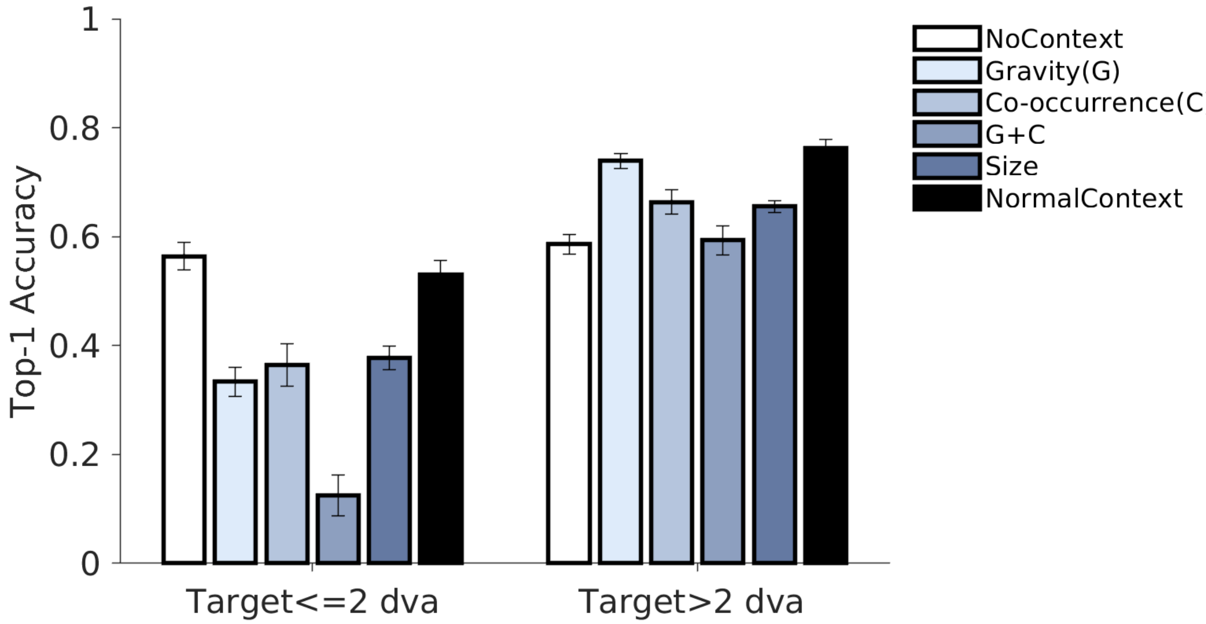


Figure Supp.51: Ablation - contextualized only. In this ablation, we used the contextualized prediction $y_{t,c}$ instead of the weighted prediction y_p as the final prediction. Different colors denote context conditions (Sec. 4.2.1, Fig. 1). Conventions and format follow Figure Supp.48.

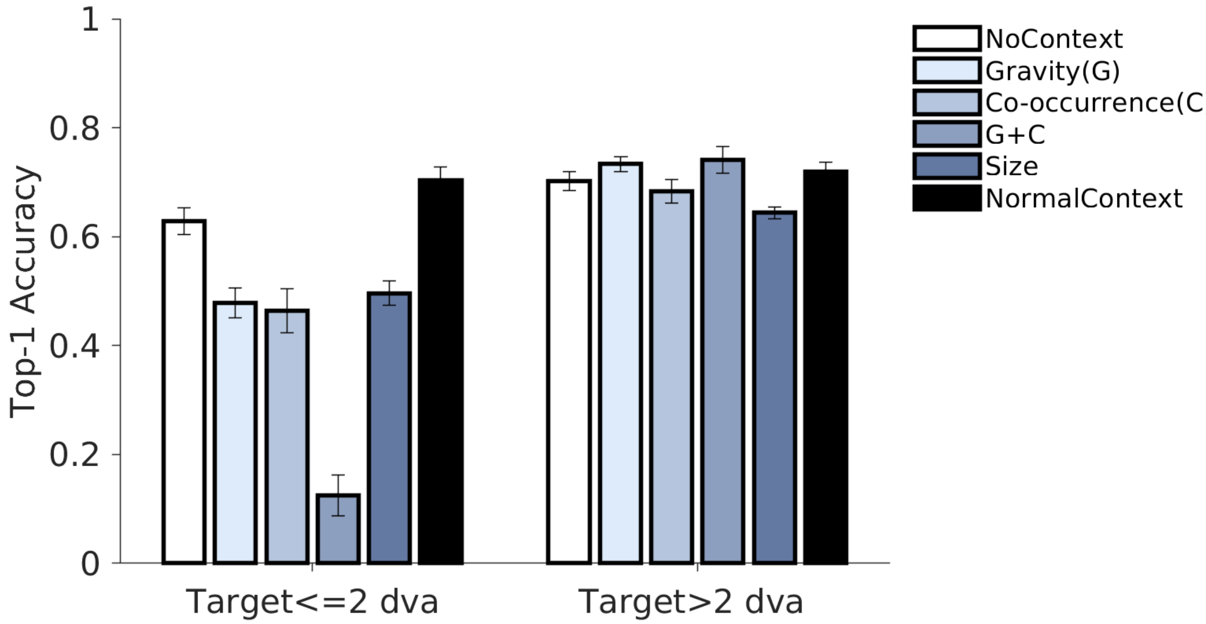


Figure Supp.52: Ablation - joint training. In this ablation, we do not detach the gradient corresponding to the cross-entropy loss with respect to y_t, therefore allowing it to influence training of the target encoder $E_t(\cdot)$. Different colors denote context conditions (Sec. 4.2.1, Fig. 1). Conventions and format follow Figure Supp.48.

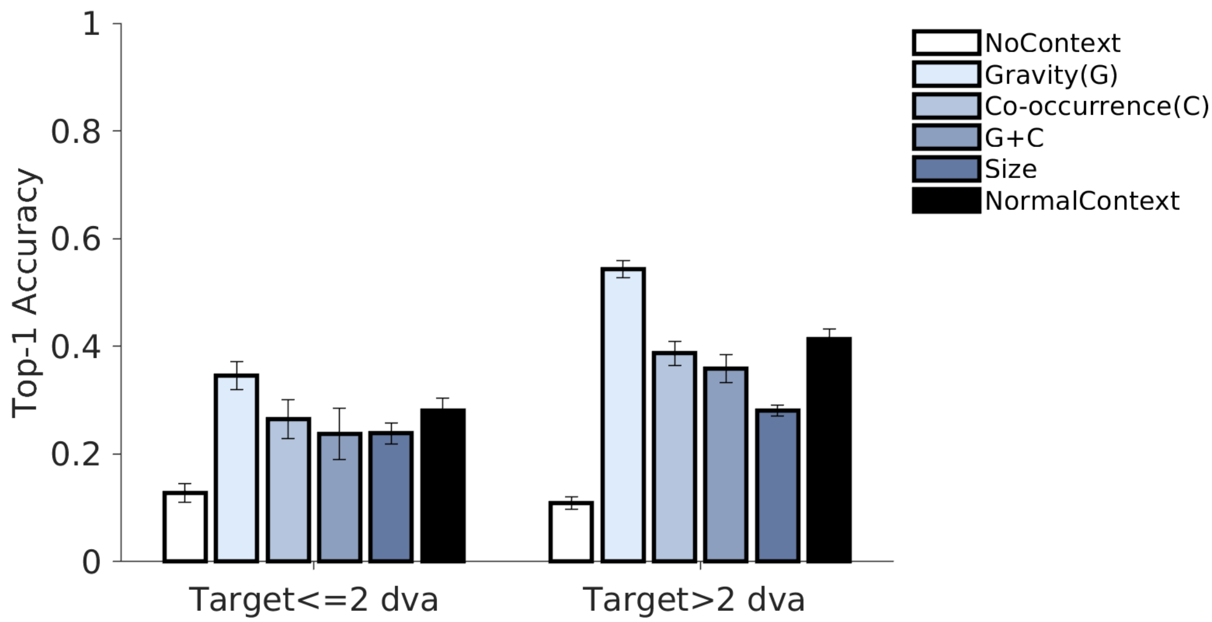


Figure Supp.53: CATNet performance across context conditions. Different colors denote context conditions (Sec. 4.2.1, Fig. 1). Conventions and format follow Figure Supp.48.

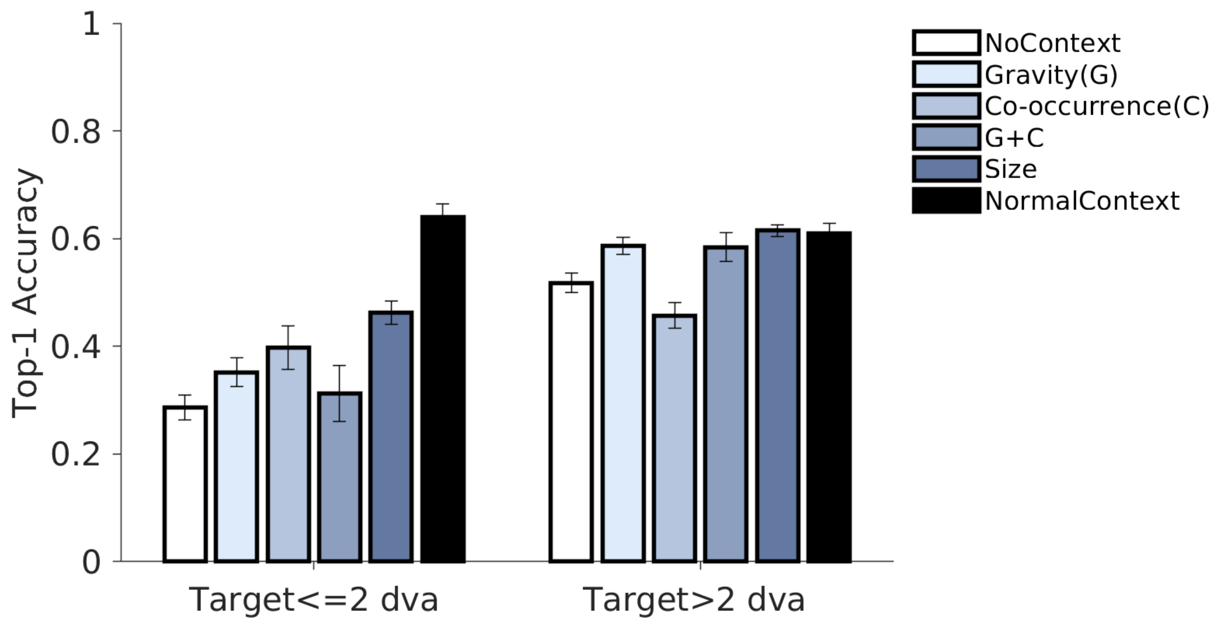


Figure Supp.54: Faster R-CNN performance across context conditions. Different colors denote context conditions (Sec. 4.2.1, Fig. 1). Conventions and format follow Figure Supp.48.

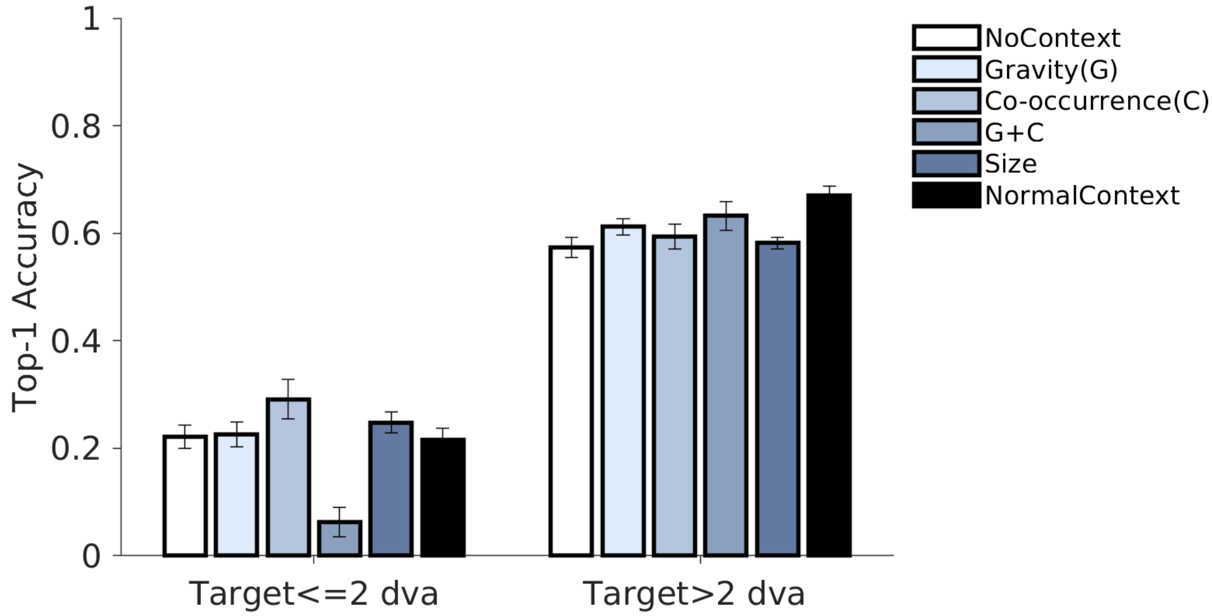


Figure Supp.55: DenseNet performance across context conditions. Different colors denote context conditions (Sec. 4.2.1, Fig. 1). Conventions and format follow Figure Supp.48.

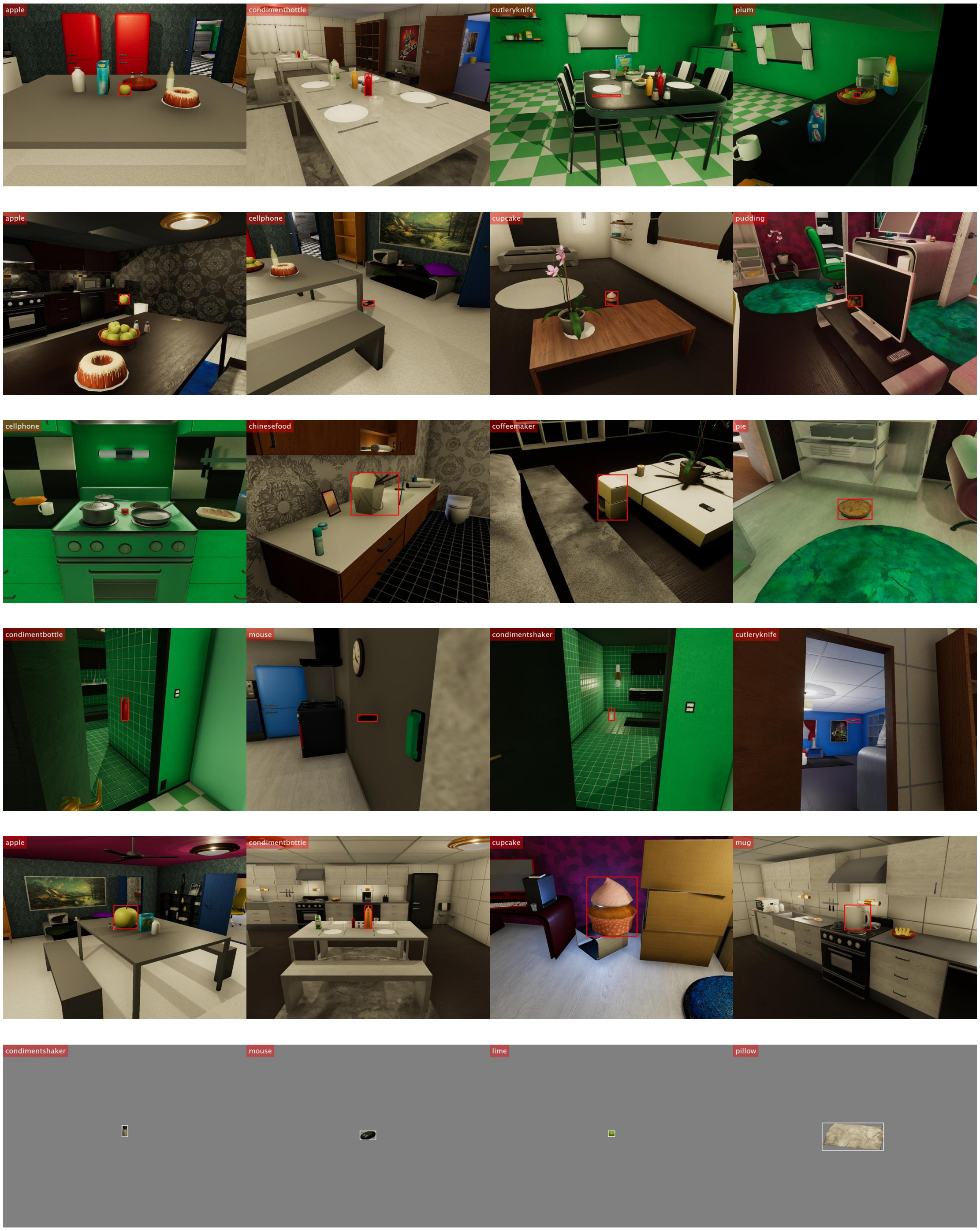


Figure Supp.56: OCD example images. Each row contains examples for one of the six context conditions (normal, gravity, co-occurrence, co-occurrence+gravity, size, no context).

Figure Supp.57: Failure Examples for Psychophysics Experiments. The examples were randomly sampled from the trials with lowest human recognition accuracy. Each row shows examples for one of the six context conditions (normal, gravity, co-occurrence, co-occurrence+gravity, size, no context).

S24 Examples of collected conversations

S24.1 Example of conversations: human-human

Here is an example conversation between two humans:
A: Hey! How are you doing?
B: Great! And you?
A: I am doing well! Any plan for the weekends?
B: Yeah, I think I'm going to visit the city. Want to come?
A: Wow yeah sure! any place you wanna visit?
B: I saw a park in the south that looks great. I think we can meet there and go for a beer or something :)
A: yeah sure! what time shall we meet?
B: Maybe around 6-7??
A: Sure it might be already dark by then though. Saturday or Sunday?
B: Saturday better! We can go then a little earlier hahah 5?
A: haha sounds good! 5 works for me. We could even catch the sunset! Hope the weather will be nice
B: yeeah I hope so :D What are you doing now?
A: I am replying to your text lol. I was checking twitter earlier
B: I was meaning a part from that :D I'm working now, do you want to go out also later?
A: today? haha sure, where do you have in mind?
B: There is a jazz club they told me is really cool
A: oh wow a jazz club! Sure, I was just listening to jazz yesterday
B: Woo cool! I love jazz, what other type of music do you listen to?
A: Depends on what work I do haha, but usually classical
B: Nice! I usually listen also to chopin. Do you play any instrument?
A: wow nice. Unfortunately not. I tried to learn keyboard when I was young but didn't keep up with it.
B: Omg I'm a music freak :D I play the piano and I want to learn how to play the guitar one day, but don't have time :D
A: wow you play the piano! That's so cool! I wish I knew how to play piano haha. Yeah it's hard to find time to learn. I am sure you will pick it up quickly though, since you already know how to play piano
B: Yeah is hard to hahaha, but one day I will :) . So I have to go, see you later!

S24.2 Example of conversations: blenderbot-blenderbot

Here is an example conversation between two machines running blenderbot:
A: Hello! How is your day going so far?

B: Ok I suppose, my days always start off slow. How about you? Are you in school?

A: Not any more. I was once though. What do you do for a living? My days are also pretty slow

B: I'm still in school, but it is summer break now so my days are very long and boring

A: At least its summer break. Your days will go by faster than you think! Where do you live?

B: Ohio, what about you? Do you have any plans for the summer yet? School starts again soon...

A: Florida! No plans really other than watching celebrity family feud on tv. It makes me laugh and brightens my day

B: My mom loves Steve Harvey, she watches his morning show every morning before work! Which one is your favorite celebrity to watch on family feud?

A: Probably the arnold swarchenegger team because he's hilarious and has a great sense of humor

B: Oh yeah he's hilarious! The grumpy grandpa vibe he gives off is great haha. Do you like big families or small ones better?

A: I don't have a big family so I would say small ones are better. There's more love that way

B: Yeah that's true, with a big family it can be hard to get close to everyone in the same way as with a small one

A: Have you seen your family recently since it is summer break? How large is yours anyway? Are they nice people lol

B: Just called my mother yesterday - she told me that we're planning to visit them this weekend. Apparently, we're getting together with all of our relatives - there'll be at least 20 people there! Can't wait to see them again :)

A: Wow that sounds like a lot of fun!! Will there be lots of kids running around too?? You must be excited

B: There will definitely be lots of kids running around haha. It should be fun though, even if they can get annoying sometimes lol

A: Oh yes well at least they are young enough not to know how annoying they can be lol!! Is your whole family coming or just some of them??

B: We'll only see some of them this time around unfortunately - most of my extended family lives out west and it's too expensive for us to fly out there right now

A: Yes I understand that for sure!! Hopefully next time then!! So what else have you been up to besides planning this trip??

B: That seems unlikely given our current financial situation haha - hopefully things will improve soon though so we can travel more often again :) Other than planning the trip, not much else really - just hanging out with friends mostly

A: Yes being young and poor sure does make things difficult sometimes!! But hang in there because life gets better as time goes on!! Well that sounds like fun too!!! What kind of stuff do yall do together??

B: Yes definitely agree with that haha :) Mostly just playing video games together or going out for food/drinks etc

A: Fun!!! Do yall play online or split screen??? And where do yall go eat/drink at??

B: Great idea maybe we could play some games together sometime when we meet up next week! Maybe from party mode in mario kart

S24.3 Example of conversations: GPT3textdavinci002-GPT3davincidavinci002 - successful

Here is an example of a successful conversation (i.e., not needing resampling due to early exit or repetition of sentences or block of sentences) satisfying the criteria between two machines running GPT3textdavinci002:
A: Hey! I'm so glad you're here. Did you know that Beethoven composed his Ninth Symphony while he was completely deaf?
B: Yeah, I did know that. He's one of my favorite composers.
A: Really? I'm more of a fan of Bach myself.
B: I can see that. Bach is a great composer, but I think Beethoven's passion for music really comes through in his work.
A: That's true, but I think Bach's work is more complex and innovative.
B: I can see your point, but I think Beethoven's work is more emotionally moving.
A: That's a valid point. I think both composers are great in their own way.
B: Absolutely! I'm glad we can agree on that.
A: Me too. So, what kind of music do you like?
B: I like all kinds of music, but I'm particularly fond of classical and jazz.
A: That's great! I like those genres too. Do you have any favorite composers or artists?
B: Yeah, I do. I really like Bach and Beethoven, as well as Miles Davis and John Coltrane.
A: Wow, you have excellent taste in music! I'm a big fan of Bach and Beethoven too, but I don't know much about jazz.
B: Well, jazz is a great genre of music. It's very complex and often improvisational.
A: That sounds interesting. I'd like to learn more about it.
B: I can recommend some good jazz albums if you're interested.
A: That would be great! I'm always looking to expand my musical knowledge.
B: I'm sure you are. You have such a passion for music.
A: Yeah, I do. It's one of the things that makes life worth living for me.
B: I know how you feel. Music is a very important part of my life as well.
A: I'm glad we can share that in common.
B: Absolutely. It's always great to find someone who shares your interests.
A: I know. It's one of the best things about life.

B: Well, I should probably get going. It was great talking to you.

S24.4 Example of conversations: GPT3textdavinci002-GPT3davincidavinci002 - discarded

Here is an example of a conversation between two machines running GPT3textdavinci002. This conversation was discarded. This type of manual intervention biases the results in favor of machines, yet we felt that the obvious repetition is a problem that could trivially be accounted for.

A: Oh, hi! I'm so glad to see you. I was just talking about books with my friend. Do you like reading?
B: Yes, I love reading! What kind of books do you like?
A: I like all kinds of books, but I especially love fantasy and fiction. What about you?
B: I also like all kinds of books, but I especially love non-fiction and history.
A: Oh, I love history too! What are some of your favorite historical figures?
B: I love learning about all kinds of historical figures, but my favorites are probably the Founding Fathers.
A: Oh, I love learning about the Founding Fathers too! What do you think made them so great?
B: I think their dedication to liberty and justice made them great. What do you think?
A: I think their dedication to liberty and justice made them great. What do you think?
B: I think their dedication to liberty and justice made them great. What do you think?
A: I think their dedication to liberty and justice made them great. What do you think?
B: I think their dedication to liberty and justice made them great. What do you think?
A: I think their dedication to liberty and justice made them great. What do you think?
B: I think their dedication to liberty and justice made them great. What do you think?
A: I think their dedication to liberty and justice made them great.
B: I think their dedication to liberty and justice made them great.
A: I think their dedication to liberty and justice made them great.
B: I think their dedication to liberty and justice made them great.
A: I think their dedication to liberty and justice made them great.
B: I think their dedication to liberty and justice made them great. What do you think?

A: I think their dedication to liberty and justice made them great. What do you think?

B: I think their dedication to liberty and justice made them great. What do you think?

A: I think their dedication to liberty and justice made them great. What do you think?

B: I think their dedication to liberty and justice made them great. What do you think?

S24.5 Example of conversations: GPT3textcurie001-GPT3textcurie001 - successful

Here is an example of a successful conversation (i.e., not needing resampling due to early exit or repetition of sentences or block of sentences) between two machines running GPT3textcurie001:

A: Hey! What are you up to?
B: Just hanging out with my friends.
A: Do you like to read books?
B: Yeah, I do. What about you? Do you like to read books?
A: I do, but I also like to watch movies.
B: That's cool! Reading is great, but sometimes it can be boring if you're not into the book. Movies let you see the story in a different way, and they can be more fun.
A: I can see that. Do you think movies are more educational than books?
B: I think they can be. Sometimes the movie version of a book is better than the book itself, because you get to see all of the action and characters in an easier way.
A: That makes sense. Do you like to read for fun or to learn something?
B: I like to read for fun. But I also like to learn something, especially if the book is interesting.
A: That makes sense. Do you have any favorite books?
B: Yeah, I do. One of my favorite books is The Hunger Games by Suzanne Collins. It's a great book and it's really suspenseful.
A: I've read that book, and it's really great. I also like The Maze Runner by James Dashner. It's a bit different than most books, but I think it's really good.
B: I think you're right. Those are both great books. Do you have any other favorite books?
A: I do have other favorite books, but I don't want to give them away. They're all personal favorites and I'm not sure if everyone would like them. But one of my favorites is The Catcher in the Rye by J.D. Salinger. It's a really unique book that
B: I've read that book, and it's really great. I also like The Maze Runner by James Dashner. It's a bit different than most books, but I think it's really good.
A: I think you're right. Those are both great books. Do you have any other favorite books?

B: I do have other favorite books, but I don't want to give them away. They're all personal favorites and I'm not sure if everyone would like them. But one of my favorites is The Catcher in the Rye by J.D. Salinger. It's a really unique book that

A: I've read that book, and it's really great. I also like The Maze Runner by James Dashner. It's a bit different than most books, but I think it's really good.
B: That's interesting. I've never read that book, but I'm definitely going to check it out. Are there any other books you like that we haven't talked about?
A: Yes, there are. I really like The Girl with the Dragon Tattoo by Stieg Larsson and Gone Girl by Gillian Flynn. They're both great books that have a lot of suspense and mystery in them.
B: I've read both of those books, and they're really good. They definitely have a lot of suspense and mystery in them, which I love.
A: I can see that. Do you have any other favorite books?
B: Yes, I do. One of my favorite books is The Lord of the Rings by J.R.R Tolkien. It's a really great book and it's really epic.

S24.6 Example of conversations: Human-Blenderbot

Here is an example of a conversation between a person and a machine running Blenderbot. A is Blenderbot and B is a person.
A: Hi! What's new with you?
B: Nothing much, how about yourself!
A: Well... My son was just accepted to the college of his dreams. I'm so proud of him!
B: Damn! That's awesome. Congratulations :) Where's he going
A: Thanks! He is going to michigan state. I will be sad when he leaves, but it's a good school.
B: It is! I'm from Wisconsin, so I'm all about the Midwest pick haha. Does he want to do engineering?
A: He does! His dream job would be working for michigan tech in houghton, michigan. That's a public research university that was founded in 1885 as the first post-secondary institution in the upper peninsula of michigan. Have you heard of it?
B: I haven't! Michigan St is a really great school for engineering.
A: Nice, it seems like a lot of people go there and then end up working at michigan tech because they have such good connections there.
B: Hey, it pays to know people, right. Everything else has been good for you?
A: Aside from being sad about my son leaving for college soon, things are pretty good here at home. How about you? What have you been up to lately?
B: Oh, a little school, a little work. I recently moved so I'm just trying to keep my head on my shoulders haha
A: Right on - where did you move from and where did you move to? Did your job let

you relocate or did you find something new after moving?

B: Just a move from one part of the city to another. I'm a grad student so I don't really have the luxury of leaving the city
A: Right - do you live near campus or somewhere else in town? Do they offer grad housing or anything like that at your school?
B: They do, but I opted not to use it. I live close enough to walk in. I actually moved farther away from the campus haha
A: Wow - what made you decide that living further away was better than living closer and having access to grad housing if needed/wanted? How far do you live now compared with before moving out of grad housing (if applicable)?
B: Yea, I never lived in grad housing. I moved from 1 off campus place to another. Definitely the location. My old neighborhood was inundated with undergrads
A: Have any students come by your place since moving into your new neighborhood yet? My nephew is starting college next year and we're looking for some places near campus for him
B: Cool! What school is going to. I've had a bunch of my friends (who are all grad students haha) over.
A: All over really; we're still deciding between ucla and ucsb but he wants something warm since we're coming from minnesota
B: I would not recommend anywhere near me then haha. The winters are cold (though not as bad as Wisconsin). I'm actually not familiar with ucsb!
A: Isla vista killings were pretty bad though; california can be scary sometimes lol but its also beautiful
B: Isla vista killings?? Is that like the golden state killer?

S24.7 Example of conversations: Human-GPT3textdavinci002

Here is an example of a conversation between a person and a machine running GPT3textdavinci002. A is a person and B is the machine.
A: Do you play games?
B: Yes, I love playing games!
A: what game do you play?
B: I love playing all sorts of games!
A: I play video games alot, especially ARPG and MOBA. I also play board games. what about you ?
B: I love playing games! I especially enjoy strategy games and games that require a lot of thinking.
A: I like League of Legends and DOTA. In terms of ARPGS, I like Path of Exile and Diablo. Board game wise, Coup and monopoly is pretty good. And you?
B: I love playing all sorts of games! I especially enjoy strategy games and games that require a lot of thinking.

A: I like MOBA. MOBA is especially fun with 5 man stacks. POE and Diablo is fun because I like the grind.

B: I love playing all sorts of games! I especially enjoy strategy games and games that require a lot of thinking.
A: What kind of strategy games ?
B: I love playing all sorts of strategy games! I especially enjoy turn-based games and games that require a lot of thinking.
A: Interesting ! Are you talking about some games like Civilization VI ?
B: Yes, I love playing Civilization VI! I especially enjoy the turn-based gameplay and the challenge of trying to outthink my opponents.
A: How did you outthink your opponents ?
B: I love playing all sorts of strategy games! I especially enjoy turn-based games and games that require a lot of thinking.
A: But how did you win the game ?
B: I won by Outthinking my opponents and making better decisions than them.
A: In the Civilization VI ?
B: Yes, in Civilization VI.
A: Sounds good. You must have played for hours.
B: Yes, I played for hours and hours, and I loved every minute of it!
A: Recently I downloaded a lot of games from the Stream. It cost me a lot to buy games.
B: That sounds like a lot of fun! I love playing games, and I'm always happy to try out new ones.

S24.8 Example of conversations: Human-GPT3textcurie001

Here is an example of a conversation between a person and a machine running GPT3textcurie001. A is the machine and B is the person.
A: Hey! How are you?
B: Hey, I'm doing great. What about you???
A: I'm good. I just finished up work and am looking for something to do.
B: Did you go for work today???
A: Yeah, I went to the office.
B: We had vacation for a week and the amount of assignments are killing me
A: Yeah, I know. Is there anything you're interested in doing?
B: Right now I'm very much occupied with my university's thing that I don't even have time for other things but I'd love to spend some time learning Korean
A: Korean?
B: Yeah, the language
A: Oh, I see. That's really interesting and something that I would like to learn too

one day. Would you like me to teach you?

B: Can you? I never knew you know Korean
A: I learned it while living in South Korea for a year. It's not as hard to learn as people think and I would be happy to teach you.
B: I'd love that! Where did you stay in South Korea?
A: I stayed in Seoul.
B: It's so pretty!! I badly want to visit Seoul and attend BTS's concert
A: You should! BTS is amazing. They're my favorite group.
B: Omg!! Whom do you like the most in the band??
A: I really like Jin, Suga and J-Hope.
B: They're awesome. Which our the bands you know??
A: I like EXO, BTS and 2NE1.
B: Have been to any concert??
A: No, I haven't. But I'd love to go to one!
B: Where are you staying right now??

S24.9 Example of conversations: DialoGPT-DialoGPT

Here is an example dialog between two machines running DialoGPT:
A: Sure thing , anything in particular you need to know or want to talk about ?
B: What do you think your insecurities are and what do you think about people using you ?
A: Please elaborate on this , since it's going to sound very accusatory .
B: Alright , so i'm working on a project , and I'm going to make a post once i have written one .
A: Thanks for the quick response ! I look forward to reading it .
B: Sure thing !
A: Happy to provide any feedback !
B: I've already put it on r LifeProTips , so you'll be
A: Oooo
B: Thanks !
A: I'll stop
B: honeste
A: Thank you
B: In the kid
A: Thanks !
B: So much

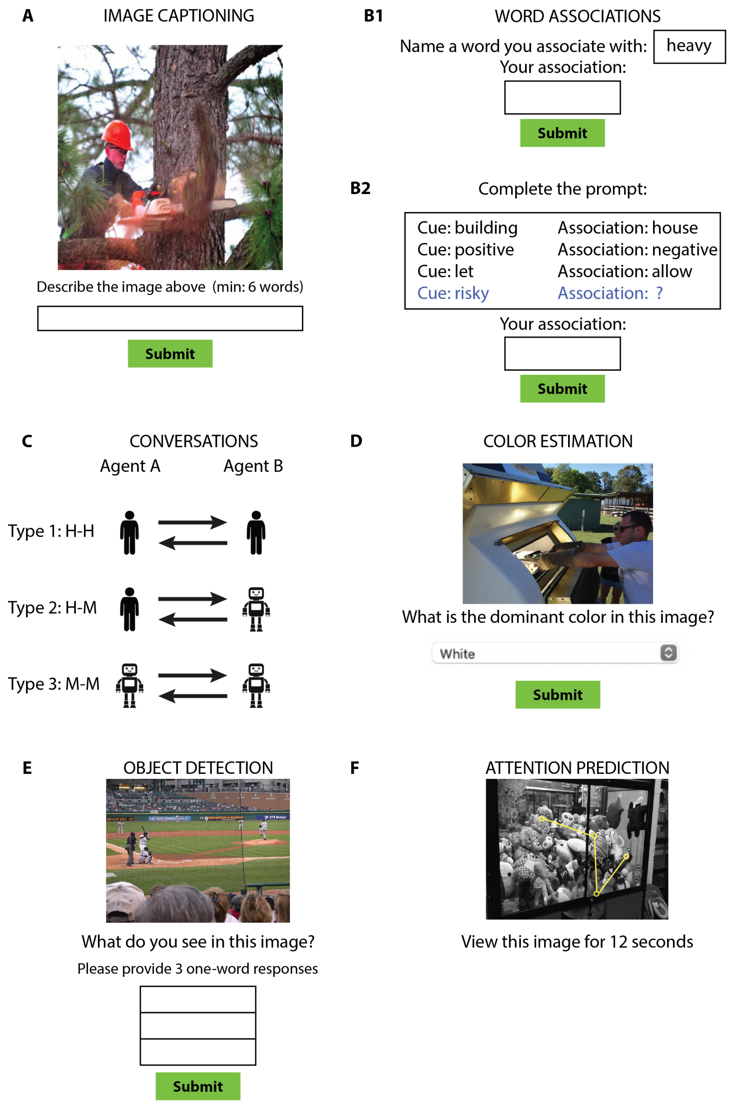


Figure Supp.58: Schematic illustration of data collection process for each task. A. Image captioning. We collected captions by asking participants to describe images, inspired by COCO Captions data collection[64]. B. Word associations. Given a cue word, participants provided a single word that they associated with the cue. There were two versions. In the free association version (B1), participants were given a cue word and were asked to freely name a single association word. In the prompt-guided version (B2), participants were given 3 cue-association pairs and provided an association for a 4th cue word. C. Conversations. We collected conversations between two agents (agent A and agent B). Each agent could be either a human or a machine. Thus, there were 3 types of conversations: human-human (type 1), human-machine (type 2), and machine-machine (type 3). For types 1 and 2, we collected live conversations on popular chatting platforms, such as WhatsApp and Messenger. Participants did not know whether they were conversing with another human or with a machine. See Methods for details and Supplementary Section S24 for example conversations. D. Color estimation. Given an image, participants selected the dominant color from a pop-up menu. E. Object detection. Given an image, participants described three things they saw in the image. F. Attention prediction. Participants freely viewed an image for 12 seconds. The circles denote fixations and the lines denote eye movements between fixations.

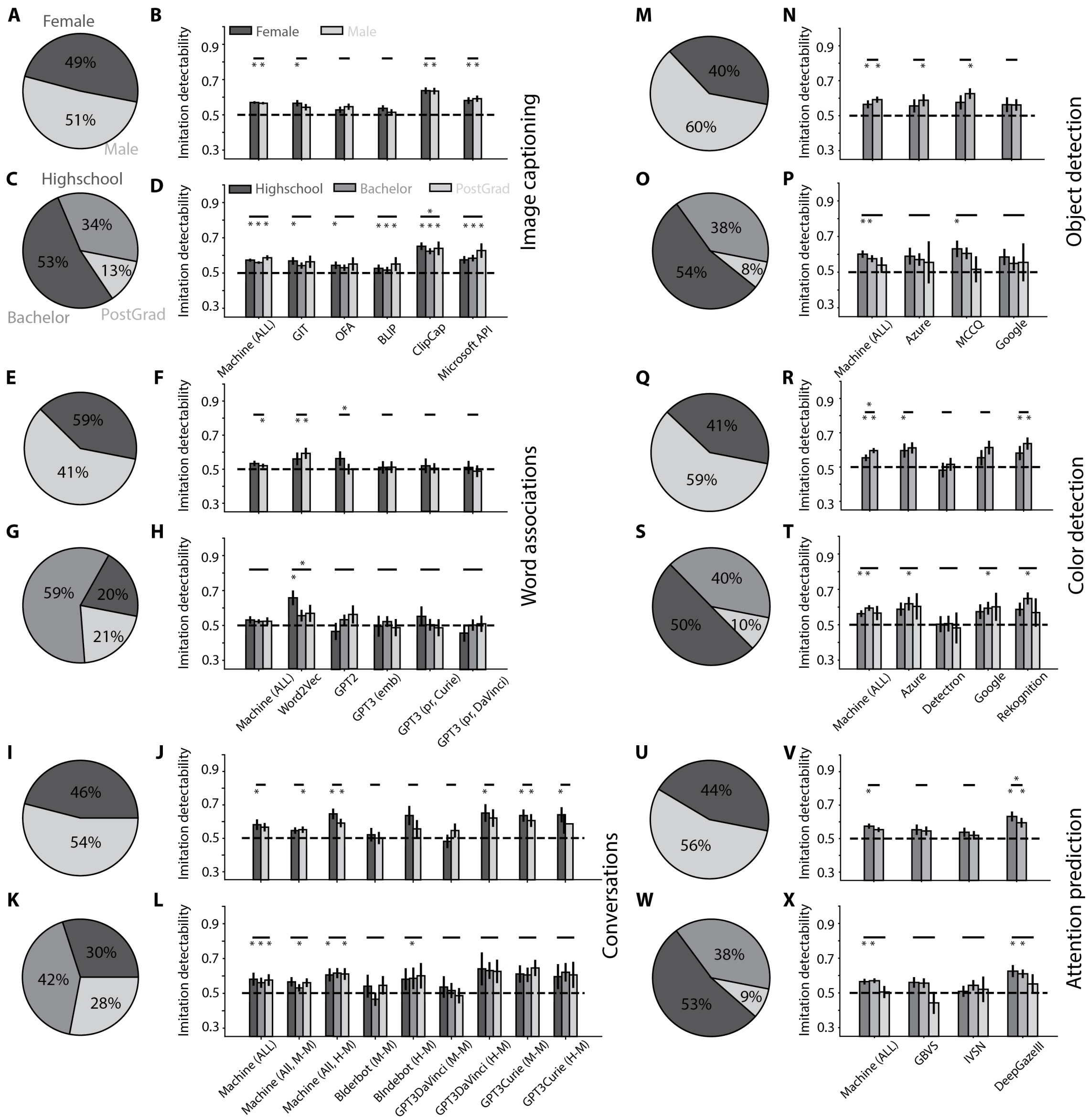


Figure Supp.59: Different demographic groups showed similar results. Results are shown separately for the Image captioning task (A-D), the Word association task (E-H), the Conversation task (I-L), the Object detection task (M-P), the Color detection task (Q-T), and the Attention prediction task (U-X). A, E, I, M, Q, U. Distribution of participants' gender (indicated by different shades of gray). C, G, K, O, S, W. Distribution of participants' education level (indicated by different shades of gray). B, F, J, N, R, V. Imitation detectability for human judges of different genders. A perfect imitator has an imitation detectability of 0.5 (horizontal dashed line) whereas a bad imitator has an imitation detectability of 1.0. Asterisks above the horizontal bar denote statistically significant differences between genders (permutation test, $p < 0.01$). Asterisks below the horizontal bar indicate statistically significant differences between each bar and 0.5 (permutation test, $p < 0.01$). D, H, L, P, T, X. Imitation detectability for human judges of different education levels.

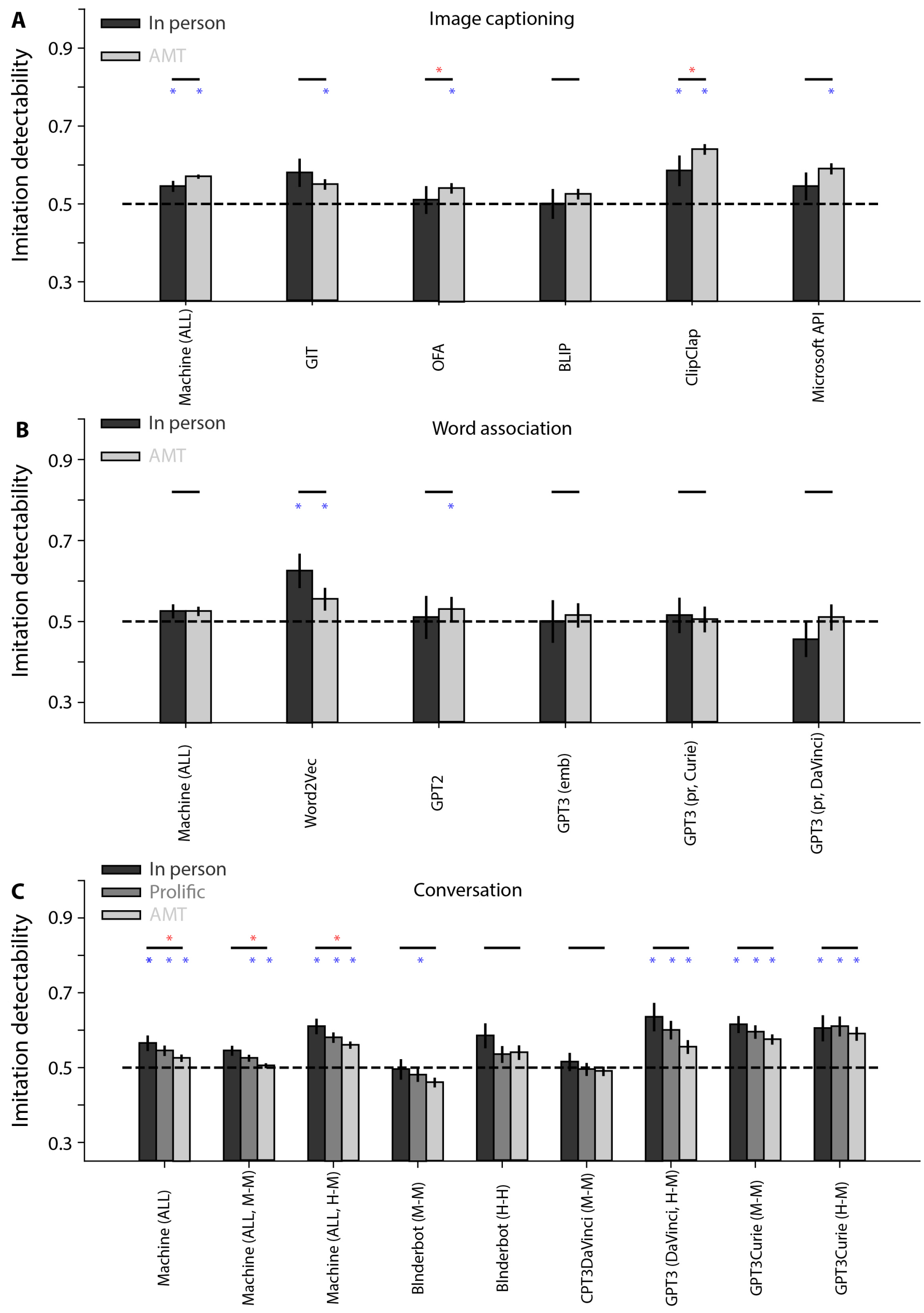


Figure Supp.60: Results of the Turing test for human judges conducted in various data collection platforms. We collected in-person data (dark gray), Amazon Mechanical Turk (AMT) data (light gray) and Prolific data for the Conversation task (medium gray). Results are shown for the Image captioning task (A), Word association task (B), and Conversation task (C). Error bars denote bootstrap standard deviations (see Methods, Data analyses). The dashed line denotes a good imitator with imitation detectability at random level. The asterisks ($*$) denote the statistical significance ($p < 0.05$). Red asterisks above the line denote comparisons among the different platforms. Blue asterisks below the line denote comparisons with perfect imitation.

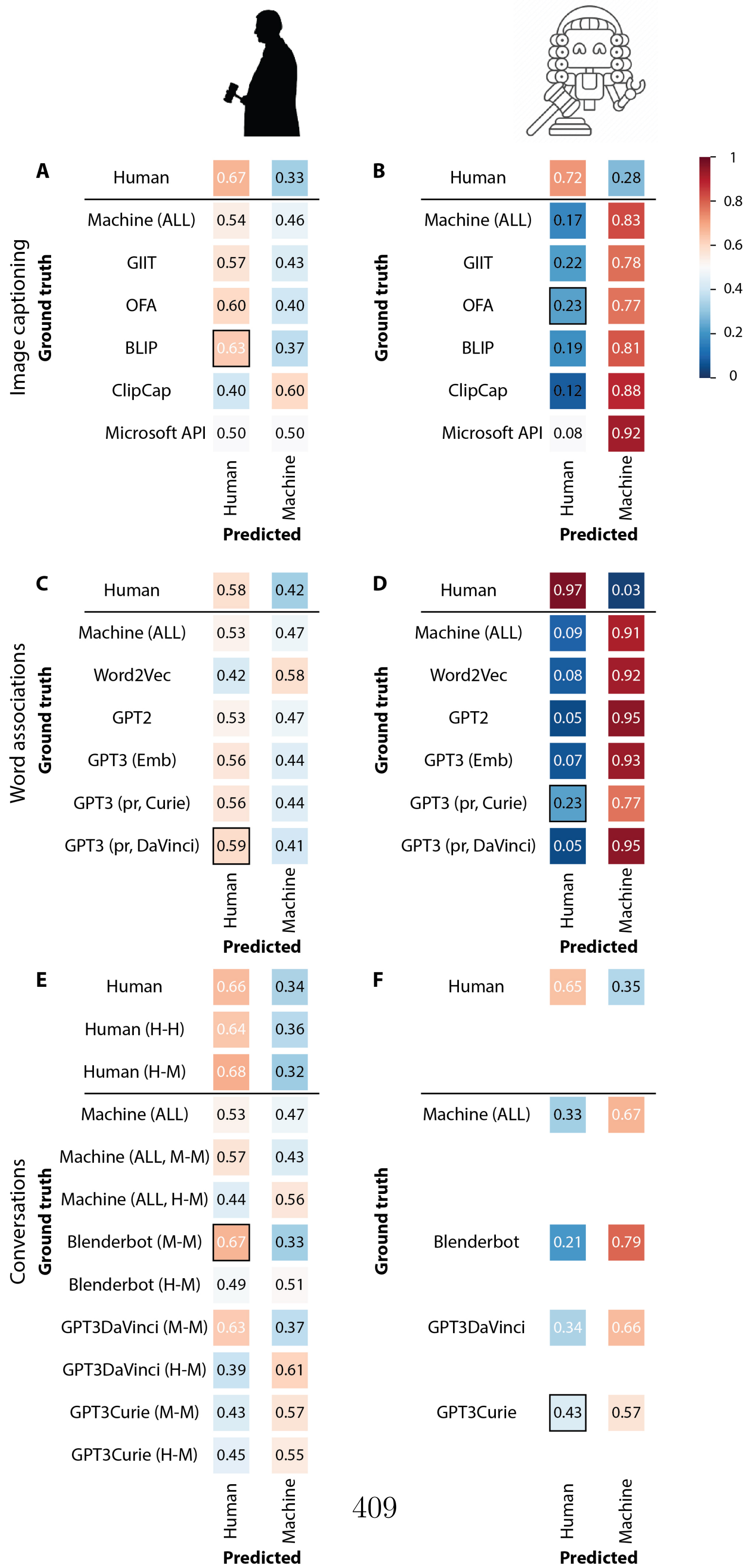


Figure Supp.61: Full results of the Turing tests for each language task. Turing test results for human judges (left, A, C, E) and AI judges (right, B, D, F) for image captioning (A, B), word associations (C, D), and conversations (E, F). The confusion matrices follow the same conventions as Figure 9.3. Note that F has fewer rows than E. The reason is that AI judges take one single sentence as input; thus, there are no multiple exchanges from two speakers involved. The colorbar in B is applicable for all the other panels.

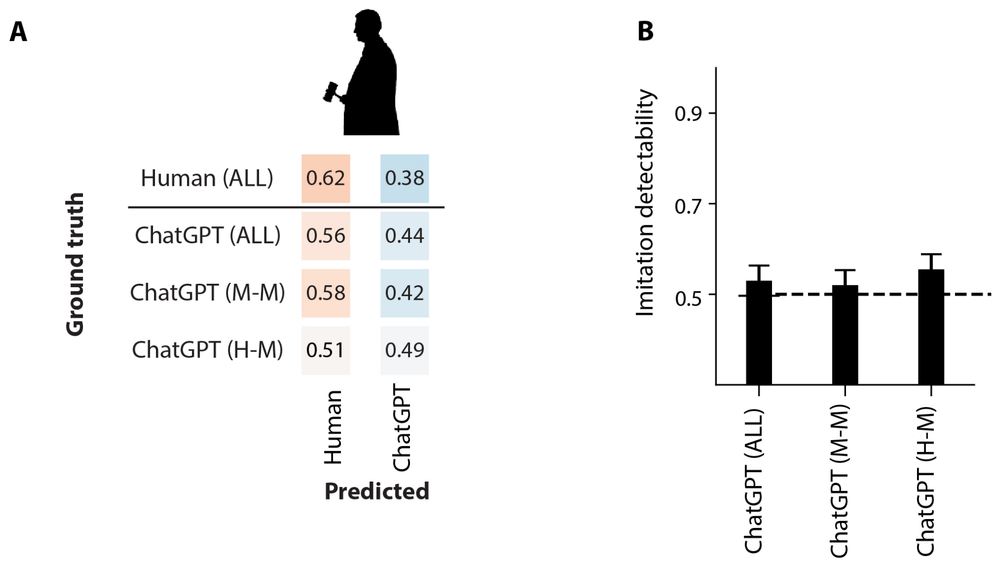


Figure Supp.62: Results of the Turing test for human judges with ChatGPT in the Conversation task. A. Confusion matrices following the conventions in Fig. Supp.61. B. Imitation accuracy for conversations with ChatGPT. None of the imitation accuracies were statistically different from 0.5. Error bars denote the bootstrap standard deviations. This figure follows the format in Fig. Supp.59.

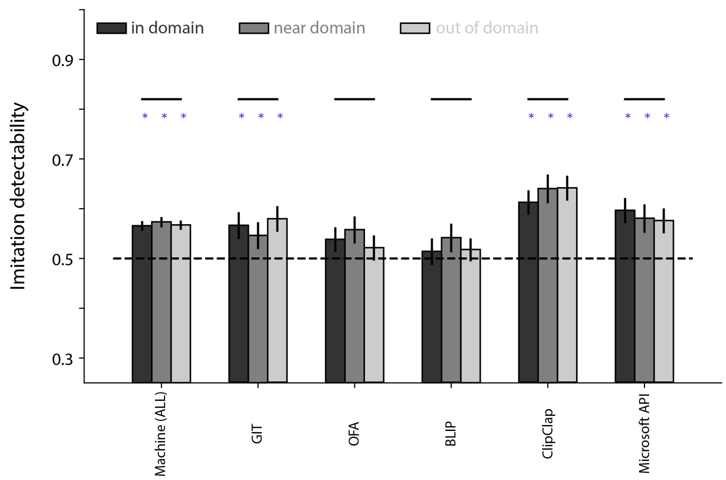


Figure Supp.63: Results of the Turing test for human judges on the NoCaps dataset in the Image captioning task. Imitation accuracy in the Image captioning task for images in the NoCaps dataset[11]. Images from the NoCaps dataset include in-domain (dark gray), near-domain (medium gray), and out-of-domain images (light gray), reflecting the similarity to object classes from the COCO dataset[64], which was used for training the AI models. Asterisks ($*$) below the line denote statistically significant differences with respect to 0.5 (horizontal dashed line, $p < 0.05$). Asterisks above the line denote statistically significant differences among the three types of domains (in this case, none of the results showed statistically significant differences).

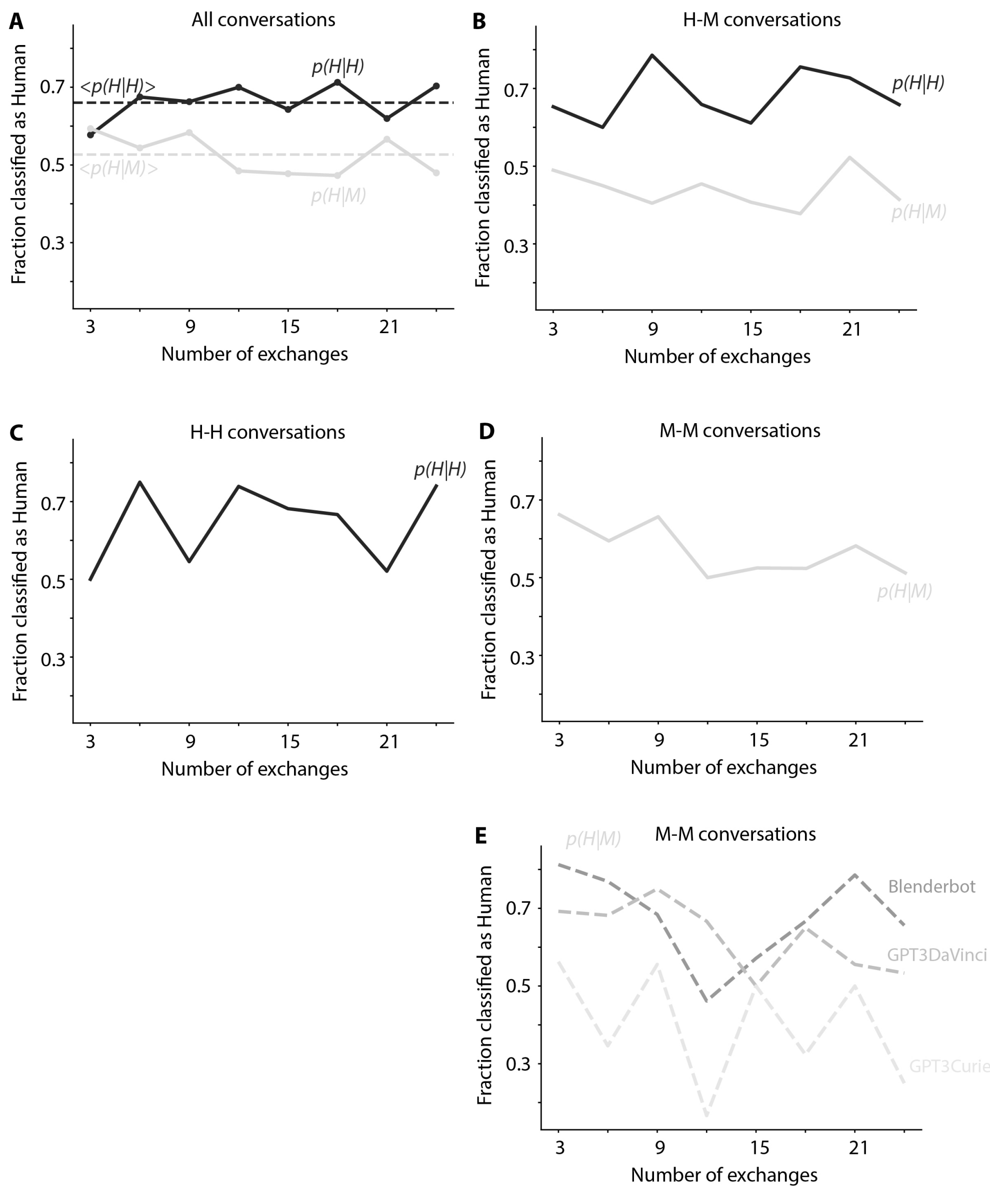


Figure Supp.64: Length dependence of Turing test results for human judges in the Conversation task. A. Average fraction of sentences where human (H) agents are classified as humans (black) or machine (M) agents classified as humans (gray) as a function of conversation length across all conversation types. The dashed lines denote the average accuracy over all conversation lengths. B. Same as A for H-M conversations. C. Same as A for H-H conversations. D. Same as A for M-M conversations. E Extension of D for different AI models. These results are for human judges (the AI judges only take one entry at a time, see Methods, Conversations).

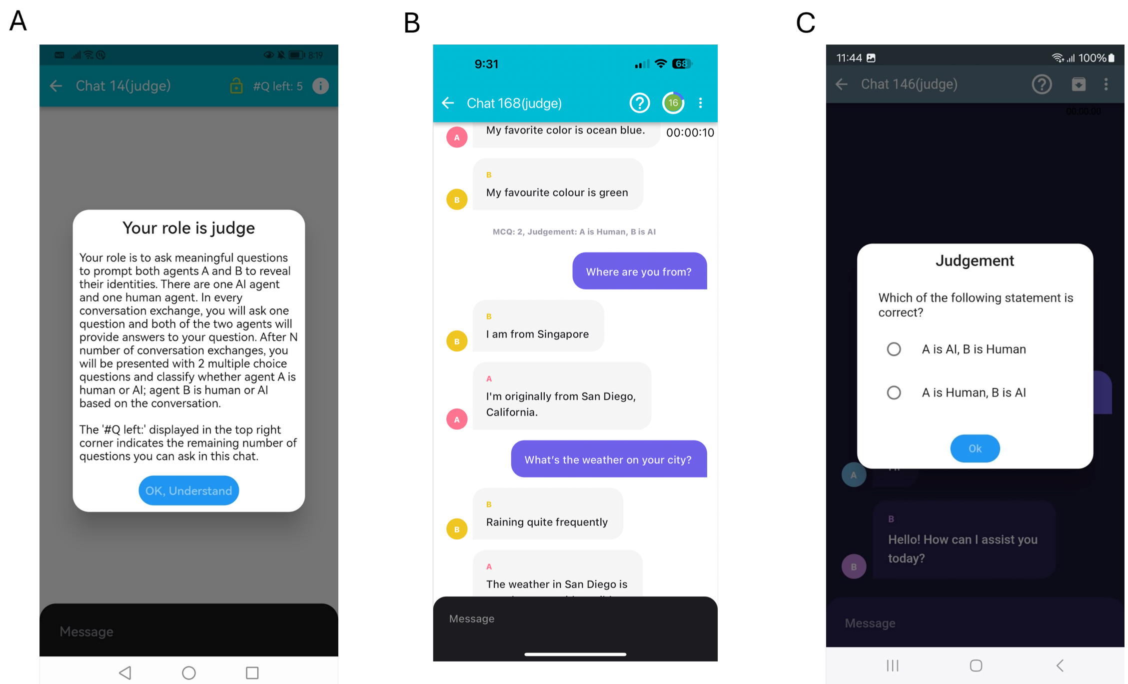


Figure Supp.65: Schematic illustration of the data collection process for conversation tasks in classical Turing tests. A. Screenshot of instruction phase. At the beginning of every Turing test, both the human judge and the human agent are presented with instructions informing them of their identity, and the objective of that role. B. Screenshot of the actual Turing test. The test always starts with a judge asking a question followed by both agents answering the questions. The number of remaining conversation exchanges is shown in orange (top right). Each agent can only see the questions from the judges, but not the answers provided by the other agent. C. Screenshot at the end of the Turing test. The judge has to make a two-alternative forced choice decision based on the responses from both agents.

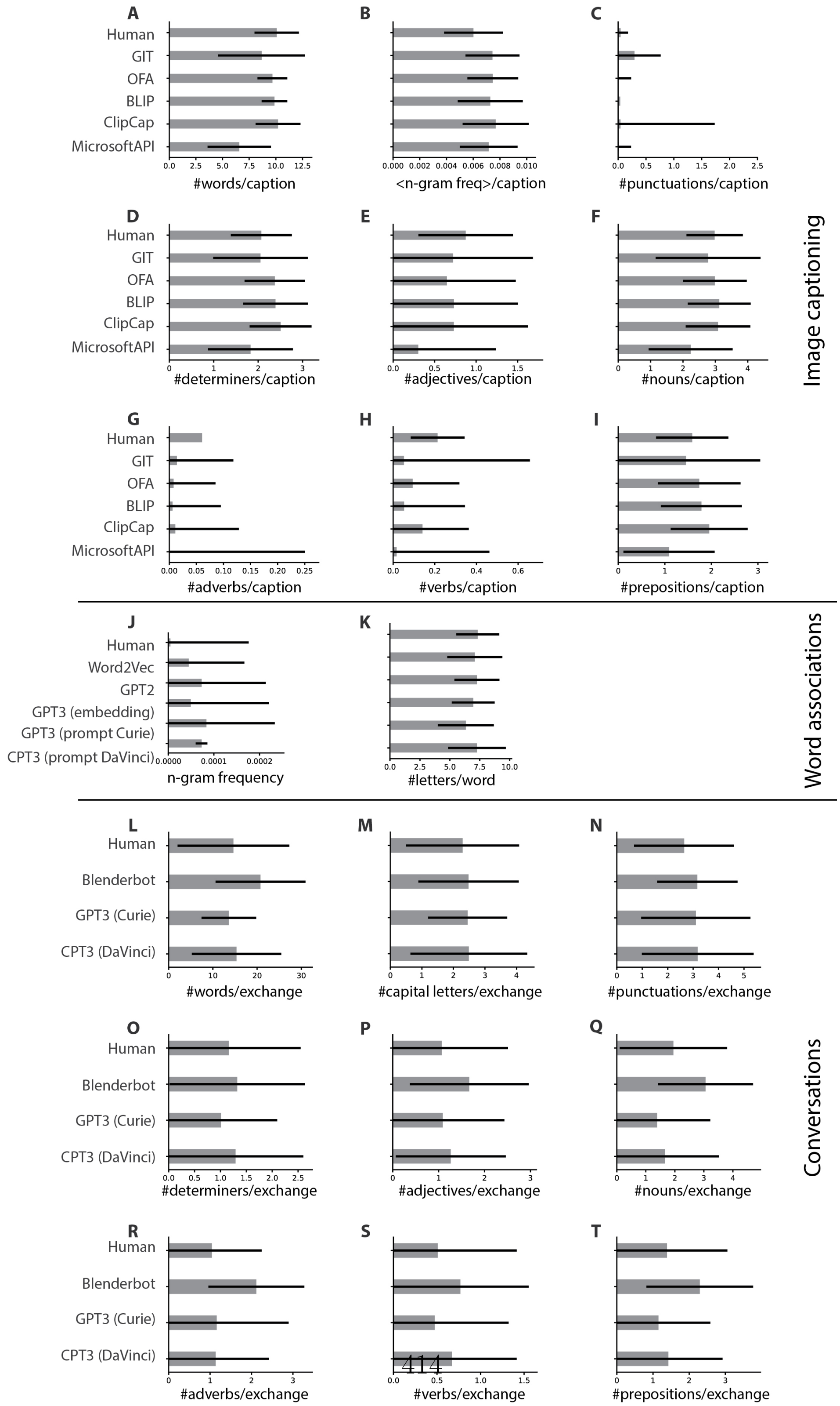


Figure Supp.66: Human and machine responses did not differ in basic low-level statistics. The figure reports multiple statistics about position-of-speech and frequency for the Image captioning task (A-I), the Word association task (J-K), and the Conversation task (L-T). These statistics include the number of words (A, L), n-gram frequency (B,J), capitalization (M), punctuation (C, N), determiners (D, O), adjectives (E, P), nouns (F, Q), adverbs (G, R), verbs (H, S), prepositions (I,T), and letters per word (K). These properties are reported per caption (A-I), per word ((J-K), or per conversation exchange (L-T). Error bars denote bootstrap standard deviations.

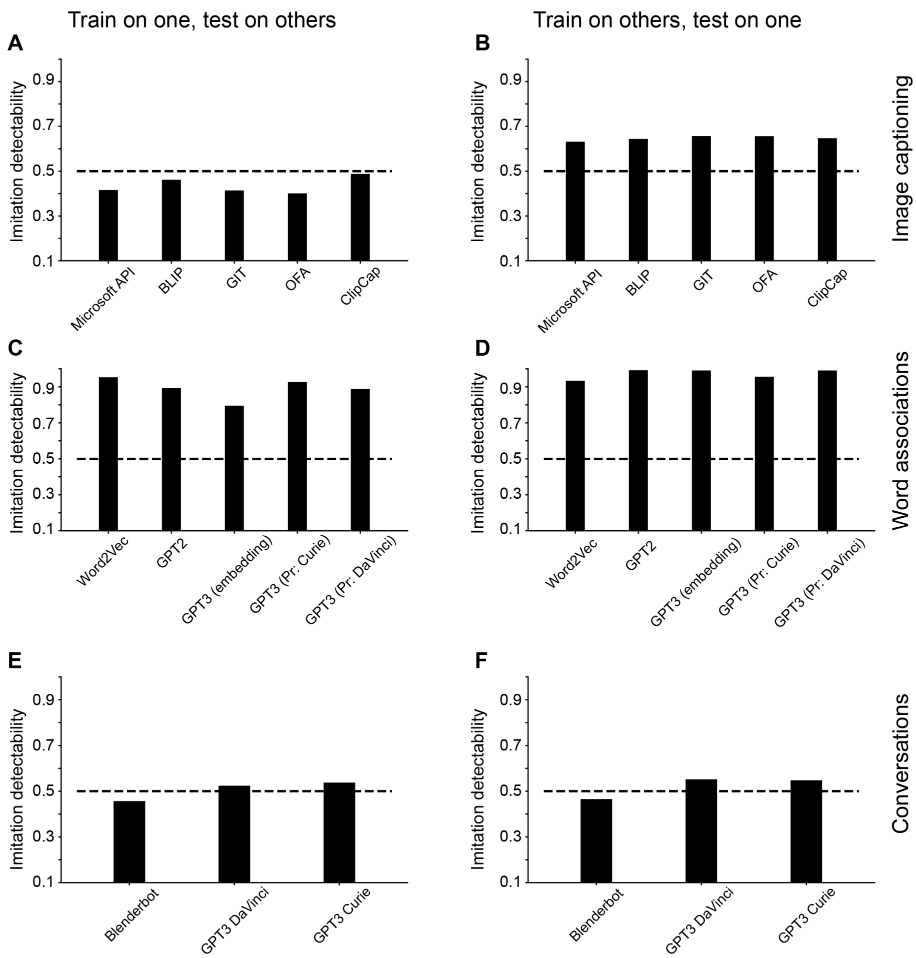


Figure Supp.67: Extrapolation across models for machine judges. Expanding on Fig. 9.3D-F , here the machine judges are trained on data from only one model and tested on all the other models (A, C, E), or trained on all models except for one and tested on that one model (B, D, F). Imitation detectability is shown for the Image captioning task (A, B), the Word association task (C, E), and the Conversation task (E, F). The horizontal dashed line indicates chance levels.

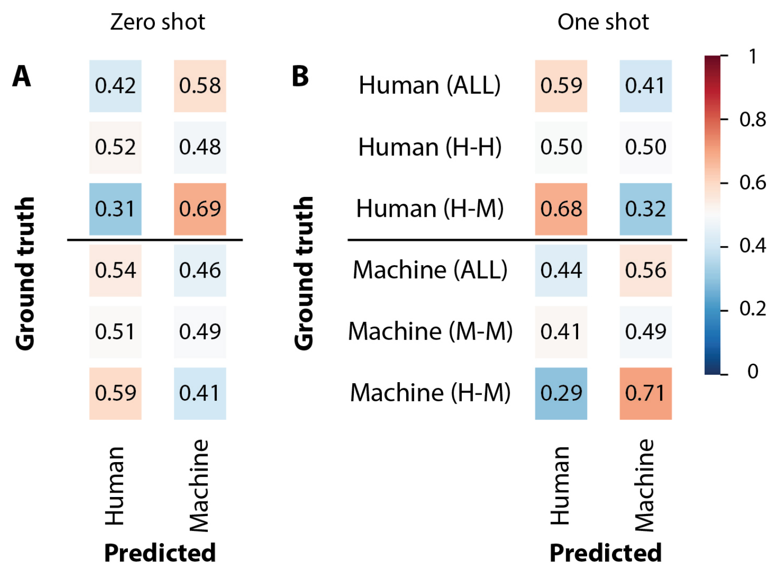


Figure Supp.68: One-shot and zero-shot Turing results in the Conversation task with large language models as AI judges. We used the large language model ChatGPT[7] as the AI judge in the Turing test for the Conversation task. In the zero-shot case (A), we prompted the model by directly presenting the conversations from the test sets with explicit instructions to output the identities of the two agents. Similarly, in the one-shot case (B), we included one additional conversation example with the ground truth identities of the two agents in the prompt before presenting the conversation from the test sets followed by the identity prediction questions as in the zero-shot case (see Methods for implementation details). The format convention of the confusion matrices in A and B follow Fig. Supp.61.

Task	Num. Stimuli	Num. Human Agents	Num. Human Judges	Num. Turing Tests	Sources of Datasets	AI agents	AI judges
Image captioning	1,000	229	323	9,400	self-collect, MSCOCO[233], nocaps[11]	GIT-Large[398], OFA-Huge[399], BLIP-Large[227], ClipCap-Transformer (beam search)[259], Microsoft's Azure Cognitive Services[mic]	SVM-GPT-curie
Word association	1,500	40	101	2,773	self-collect,	Word2Vec [288], GPT2 [302], GPT3-embedding (davinci)[43], GPT3-prompt (text-curie-001)[43], GPT3-prompt (text-Davinci-002) [43]	SVM-Word2Vec, SVM-GPT2, SVM-GPT3(davinci)
Conversation	2,460	150	367	7,717	self-collect, Topical-Chat[130]	GPT3-text-davinci-002[282], GPT3-text-curie-001[282], Blenderbot[338], DialogPT[130], ChatGPT[7], GPT3.5-turbo-1106[7]	SVM+BERT[94], ChatGPT-zero-shot, ChatGPT-one-shot
Color estimation	785	45	65	1,625	self-collect, MSCOCO[233]	Google Vision API Microsoft Azure Cognitive Services[mic], MMCQ[37]	SVM+VGG+BERT[94]
Object detection	808	45	79	1,975	self-collect, MSCOCO[233]	Google Vision API, Microsoft Azure Cognitive Services[mic], Amazon Rekognition, Detectron2[411]	SVM+BERT[94]
Attention prediction	547	40	191	2,160	NatureDesign[436], FindingWaldo[436], NatureSaliency[435]	IVSN[436,142,435], DeepGaze3[209], GBVS[152]	SVM on 2D coordinates
Total	7,100	549	1,126	25,650	-	26	10

Table Supp..1: Specifications of six Turing tasks. Source datasets, number of Turing tests conducted, number of stimulus, and number of AI models used to collect responses are listed for each task. See Methods for task descriptions.

S25 Sample images from the HVD Dataset

We present additional images from the HVD dataset. Each figure shows change in one scene parameter, while holding all others constant. In Fig. Supp.73 we show images from two different light domains. Note that the first three rows in Fig Supp.73 show different indoor lighting conditions controlled using indoor light color and intensity sampled from disjoint chunks of the HSV space. The last two rows show different outdoor lighting settings created by changing the environment maps. Similarly, Fig. Supp.74 shows five different scenes from two training domains with a material shift. Fig. Supp.75 shows viewpoint shifted domains.

S26 Additional details for the creation of the Semantic iLab dataset

We show sample images from the Semantic iLab dataset in Fig. Supp.76 created by modifying the existing iLab[39] dataset. This is a multi-view dataset, and hence already contains viewpoint shifted variations of the same objects. We modify the dataset to also contain material and light shifts. To mimick light shift, we modified the white balance of the original images, as shown in Fig. Supp.76(b). For material shifts, we first run a foreground detector on these objects using Google's Cloud Vision API. We also run style transfer on these images using AdaIn[173]. Then, we overlay the style transferred image on to the object mask on the original image to mimick material shifts. Note that this is approximate, and does not model the physics of material transfer in the same way as our rendered HVD dataset which is far more photorealistic, as shown in Fig. Supp.74. Material shifted Semantic iLab images are shown in Fig. Supp.76(c). As the dataset is originally multi-view, we do not need to generate new viewpoints and can use images of a different viewpoint from the original dataset as shown in Fig. Supp.76(d).

S27 HDNet ablations with contrastive loss

We evaluate the contribution of the contrastive loss by training variations of HDNet on HVD with and without the contrastive loss as shown in Eq. 10.2. These numbers are reported in Table Supp..2. As can be seen, adding a contrastive loss improves performance for all three semantic shifts, providing evidence for its utility.

Semantic Shift	Without Contrastive Loss	With Contrastive Loss
Viewpoint	0.79	0.82
Material	0.89	0.94
Lighting	0.98	0.98

Table Supp..2: Impact of removing contrastive loss. We evaluate the contribution of the contrastive loss by training and testing HDNet on the HVD dataset with and without the contrastive loss. The contrastive loss results in an improvement across all three semantic shifts.

S28 Additional experiment for the role of context

Besides results on the role of context presented in Table 10.2 and Table 10.3, we present here an additional control evaluating the contribution of scene context

Semantic Shift	Target only	Target and Context
Viewpoint	0.77	0.82
Material	0.85	0.94
Lighting	0.97	0.98

Table Supp..3: Training a two-stream HDNet with only target information. As a third control for confirming the role of context, we train HDNet where both streams are passed just the target object. Thus, it is forced to learn without scene context. This results in a drop in performance for all semantic shifts, providing further evidence in support of the utility of scene context.

Test Dataset	ResNet [162]	ViT [99]	AND Mask [326]	CAD [33]	COR AL [358]	ERM [390]	IRM [19]	MTL [34]	Self Reg [195]	VREx [207]	HDNet (ours)
ScanNet	0.35	0.29	0.43	0.40	0.42	0.48	0.46	0.46	0.53	0.42	0.61

Table Supp..4: Human visual diet improves generalization to larger real world dataset as well. We curated a larger subset of ScanNet images, allowing more complex real world scenarios like blurry images, clutter and occlusions. We report the capability of models to generalize from synthetic HVD images to this more complex subset of ScanNet. HDNet leveraging human-like visual-diet outperforms all baselines on this more complex dataset as well.

on generalization. For this, we train HDNet such that both streams are trained with the target object. Thus, this modified version is forced to learn without scene context. These results are shown in Table. Supp..3. For all semantic shifts, forcing HDNet to learn with only the target results in a drop in accuracy. This provides further evidence supporting the utility of scene context in enabling generalization.

S28.1 Results with a larger, less controlled ScanNet test set.

We extend the generalization to real-world results presented in the main paper by reporting these numbers on a larger test set created by annotating additional images from ScanNet. As ScanNet was created by shooting video footage of 3D scenes, many frames can be blurry. In the original, smaller test-set such blurry frames were removed to ensure a higher quality test set. However, here we also include additional images with lower fidelity to report numbers on a larger test set. These numbers are reported in Table. Supp..4. The trend is consistent with results reported on a smaller, more controlled subset in the main paper—HDNet outperforms all other benchmarks by a large margin. As expected, including these images in the test set results in a drop in accuracy across all methods. All models were trained on synthetic images from HVD and were tested on a test set of natural images from ScanNet.

S29 Hyperparameters

HDNet: As our model builds on top of CRTNet[38] as backbone, we use the same hyperparameters for the backbone as reported in the original paper. All models were trained for 20 epochs with a learning rate of 0.0001, with a batch size of 15 on a Tesla V100 16Gb GPU.

Domain generalization: We used the code from Gulrajani et al.[137] to train and test domain generalization methods on our dataset. The code is available here: https://github.com/facebookresearch/DomainBed. To begin, we ran all available models and tried 10 random hyperparameter initializations. Of these, we picked the best performing hyperparameter seed—24596. We also picked the top performing algorithms as the baselines reported in the paper.

FasterRCNN: We used the code from Bomatter et al.[38] to train and test the modified FasterRCNN model for recognition. The code is available here: https://github.com/kreimanlab/WhenPigsFlyContext, and we used the exact hyperparameters mentioned in the repo.

List of semantic categories in MacaqueITBench

Table Supp..5 reports a list of all semantic categories in MacaqueITBench. As can be seen, the $8,233$ images correspond to over 300 categories.

bottleopener	cupsaucer	tractor	bowtie
recordplayer	corkscrew	calculator	shoe
duster	lock	scissors	gift
guitar	swissarmyknife	pda	servingpiece
ceilingfan	nunchaku	radio	socks
sushi	filingcabinet	Other	waterbottle
leaves	watergun	trumpet	Big Animate
stove	chocolate	greenplant	bones
necktie	grapes	cookpot	windchime
cassettetape	mattress	fishbowl	Non

bench	tongs	microphone	lunchbox
jacket	bonzai	bullet	fan
cheesegrater	watch	cake	stool
pasta	sword	shirt	orientalplatesetting
typewriter	backpack	babushkadolls	hat
headphone	fork	wallsconce	hookah
boot	toothpaste	Gabor	abacus
quilt	short	familiarObjects	feather
fireplace	beermug	balloon	crossbow
pen	razor	dollhouse	carabiners
lightbulb	keychain	lawnmower	Glove
broom	headband	golfbag	garbagetrash
babyplayard	manorha	skateboard	shovel
christmasstocking	cooler	exercise	wineglassfull
camera	cheese	makeupcompact	plate
gong	cellphone	showercurtain	birdcage
tricycle	carfront	sleepingbag	window
umbrella	coatrack	roadsign	breadloaf
waxseal	mathcompass	dvdplayer	Rodent
handgun	binoculars	hilighter	icecreamcones
jack-o-lantern	basket	spoon	shredder
camcorder	christmastreeornamantball	apple	Face
log	cookingpan	scrunchie	stapler
flashlight	muffler	candy	orifan
golfball	pokercard	Bird	collar
washer	baseballcards	perfumebottle	babywalker
axe	patioloungechair	banana	wig
cookie	fish hook	motorcycle	sewingmachine
toy	pizza	lamp	meat
tape	tire	decorativescreen	musicstand
crib	candleholderwithcandle	grill	battery
hammer	compass	lei	hairdryer
giftbow	wheelbarrow	keyboard	trunk
iceskates	hanger	bathsuit	pill
Hand	kettle	microscope	Big
fruitparfait	Symbol	eraser	baseballbat

cage	lightswitch	laptop	sodacan
beaker	PPE	extra	kayak
sofa	fishingpole	microwave	mailbox
snowglobe	carseat	Butterfly	corset
doll	rollerskates	Fish	frisbee
trophy	saltpeppershake	pacifier	pezdispenser
rosary	router	airplane	Cat
reportfile	soapdispenser	coffin	yarn
dumbbell	chessboard	computer	aircompressor
birdhouse	Print	pipe	hotairballoon
doorknocker	anchor	bed	cashregister
loom	lipstick	measuringtape	chair
train	remotecontrol	toaster	coffeemug
pants	pie	donut	powerstrip
seasponge	beanbagchair	bike	domino
glasses	nailpolish	cherubstatue	knife
coin	printer	mp3player	leatherman
Turtle	flag	Toy	hairbrush
ladder	bucket	bell	ringbinder
wineglass	robot	stamp	spraybottle
mushroom	dresser	peppersonplate	tent
bowlofchips	videoGameController	lantern	candybar
cracker	computermouse	cane	ambulance
toothbrush	goggle	scooter	doorknob
gamesboard	lighter	tray	backgammon
tv	sink	doorwayarch	gamehandheld
wheelchair	objects	barbiedoll	coffeemaker
bagel	juice	shotglass	Mask
tablesmall	highchair	spoolofstring	helmet
horseshoe	telescope	hourglass	tweezer
ring	Misc	spicerack	handmirror
cushion	phone	vase	woodboxsmall
bowlingpin	clock	handbag	globe
key	muffin	dynamite	strainer
checkbook	pillow	sandwich	scale
ball	sippycup	Starfish	bottle

tupperware	cigarettepack	seashell	handheldvacuum
tree	earings	vacuum	suit
bullhorn	ketchupbottle	babycarriage	necklace
fridge	nest	slinky	curlingiron
desk	suitcase	pencilsharpener	speakers
button	rug	bowl	scroll
flask	paintbrush	bill	tennisracquet
boppypillow	rollingpin	saddle	frame
handkerchief	toiletseat	slate	licenseplate
laudrybasket	easteregg	accordian	crown
circuitboard	Dog	bongo	barrel
rock	pitcher		

Table Supp..5: Images from MacaqueITBench.

Sample Images from MacaqueITBench

Fig. Supp.77 shows sample images which were presented to Macaques to collect responses from the IT Cortex.

Additional results with Mid hold-out strategy

In the main paper, we presented results with two hold out strategies—high and low. Here, we present results with the third hold-out strategy outlined in the paper. We refer to this as the Mid hold out strategy as samples between the 42.5 and the 67.5 percentile of every OOD attribute are held out as the test set. As shown in Fig. Supp.78, across all architectures and OOD attributes, models suffer to generalize to OOD samples for the Mid hold out strategy.

Additional results with intermediate layers

In the main paper we presented results for models trained with intermediate layers for the high hold out strategy. Here we provide additional results with models that use intermediate layers of DNNs as feature extractors. In Fig. Supp.79 and Fig. Supp.80 we report results for the low and mid hold-out strategies respectively.

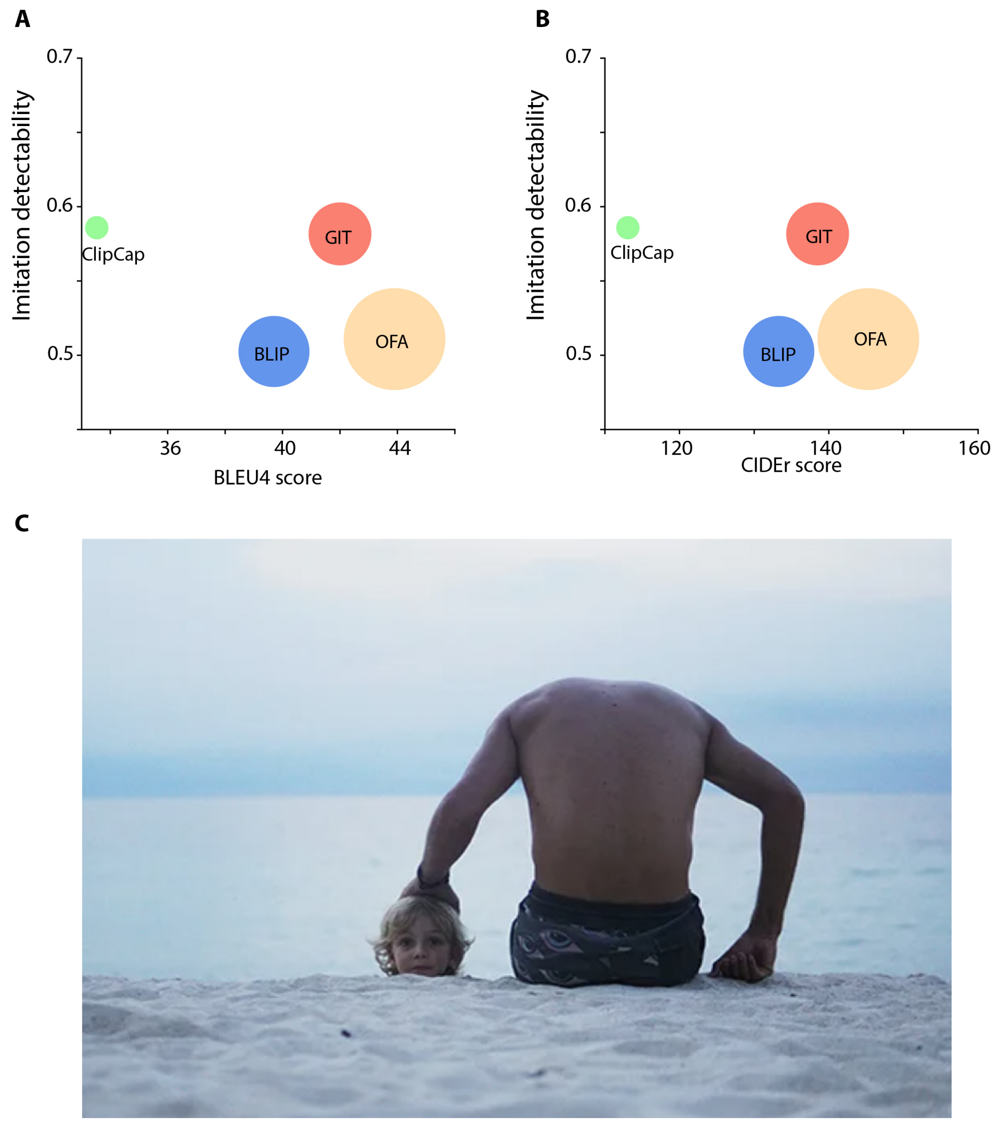


Figure Supp.69: Correlation between Imitation accuracy and standard performance metrics in the Image captioning task. Four state-of-the-art image captioning models, GIT[398] (red), OFA[399] (yellow), BLIP[227] (blue), and ClipCap[259] (green), are evaluated in terms of their Imitation accuracy (y-axis) as a function of their BLEU score[284] (A), and the CIDEr score[399] (B). The BLEU score and the CIDEr score are standard evaluation metrics commonly used in image captioning tasks to assess the quality of the generated captions given an image based on n-gram overlaps. The higher the values, the better. The radius of the circles is proportional to the AI model sizes in terms of the number of trainable network parameters. C. Illustration of correct answers that do not necessarily imitate humans. In response to the query "What do you see in this image?" GPT4 answered: "In this image, a man is sitting on a sandy beach with his back to the camera, facing the ocean. Next to him, a child's head is visible, appearing to be buried in the sand up to their neck. The man has one arm extended, resting his hand gently on the child's head. The scene is serene with the calm sea and sky in the background, creating a peaceful atmosphere. The combination of the man's pose and the child's position in the sand creates a unique and slightly humorous visual effect, suggesting a playful moment between the two." While the answer is correct and remarkably articulate, a human described the image differently: "In this curious and well-consrtucted image, we cannot see the man's head and it appears as if the boy's head belongs to him and the man took his own head off".

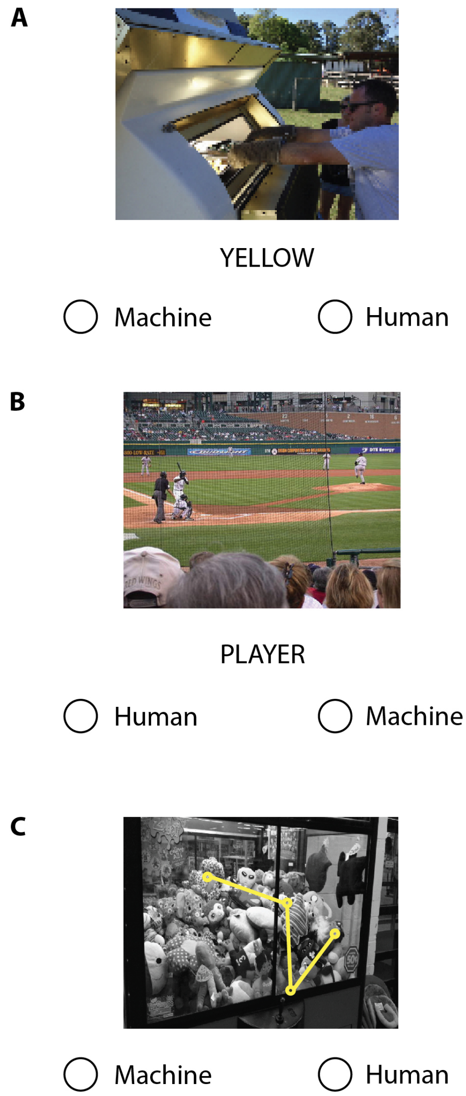


Figure Supp.70: Schematic illustration of Turing tests for three vision tasks. See also Fig. 9.1 for Turing tests for the three language tasks. A. Color estimation. Participants were presented with an image and a color and had to indicate whether the color selection was made by a human or a machine. B. Object detection. Participants were presented with an image and a noun and had to indicate whether the object description was made a human or a machine. C. Attention prediction. Participants were presented with an image and a sequence of positions (yellow circles) joined by lines and had to indicate whether those locations were the product of human eye movements or machine attention predictions.

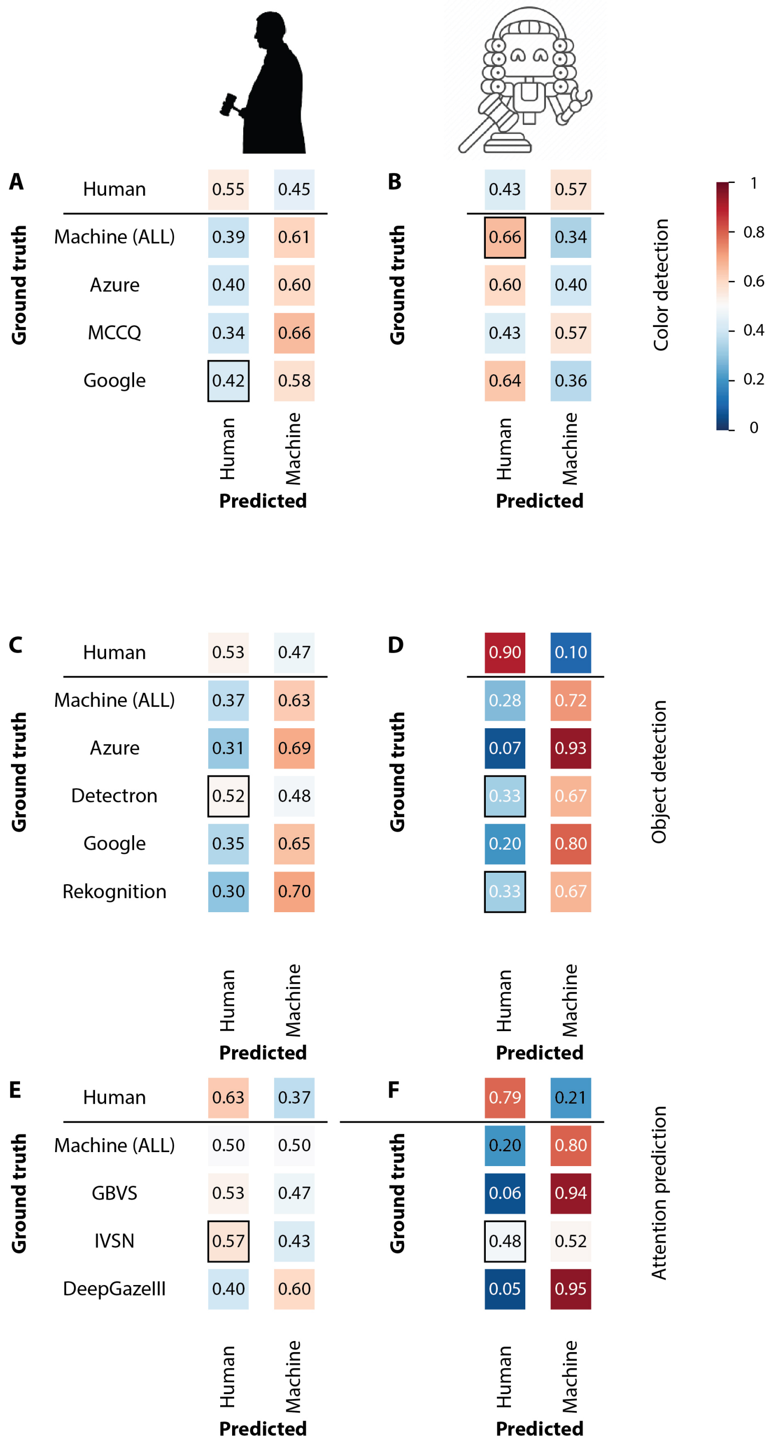


Figure Supp.71: Full results of the Turing test for each Vision task. Turing test results for human judges (left, A, C, E) and AI judges (right, B, D, F) in the three Vision tasks: Color detection (A,B), Object detection (C,D), and Attention prediction (E,F). The full confusion matrices follow the same conventions as Fig. 9.3. See the color bar in B which applies to all panels. The boxes with a black frame denote the best algorithm in terms of its ability to pass as human, i.e., highest $p(H|M)$.

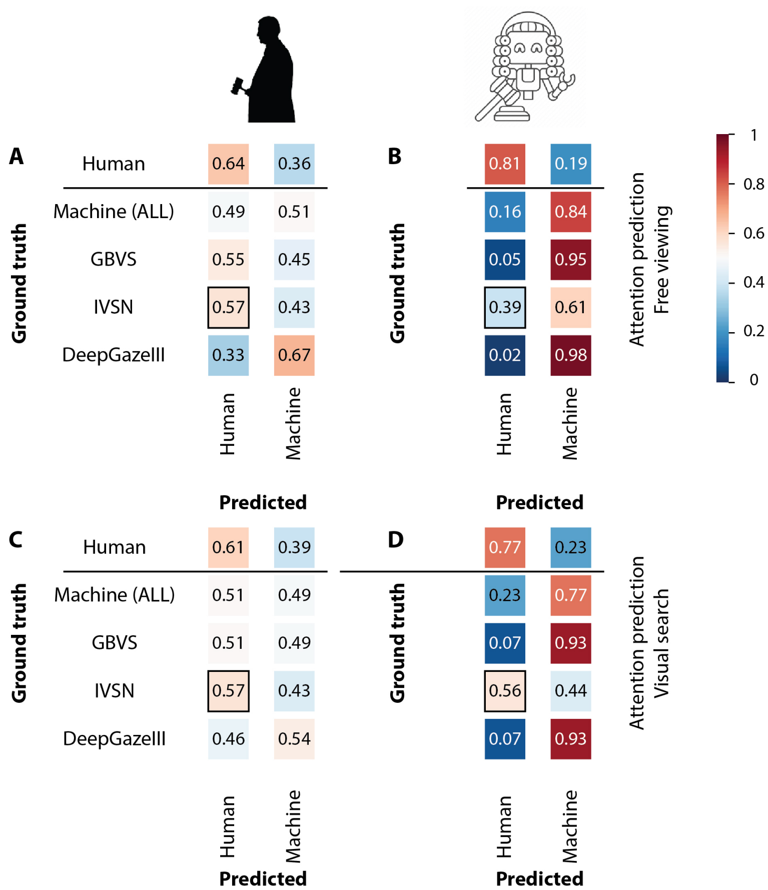


Figure Supp.72: Comparison between free viewing and visual search in the Attention task. Using the same format as in Fig. Supp.71, the results in Fig. Supp.71E are shown here separately for free viewing (A) and visual search (C) and the results in Fig. Supp.71F are shown here separately for free viewing (B) and visual search (D).

Figure Supp.73: Example images showing lighting tranformations. We show paired images from different lighting transformation domains between the right and left column in each row. All other parameters held constant.

Figure Supp.74: Example images showing material tranformations. We show paired images from different material transformation domains between the right and left column in each row. All other parameters held constant

Figure Supp.75: Example images showing viewpoint tranformations. We show paired images from different viewpoint transformation domains between the right and left column in each row. All other parameters held constant

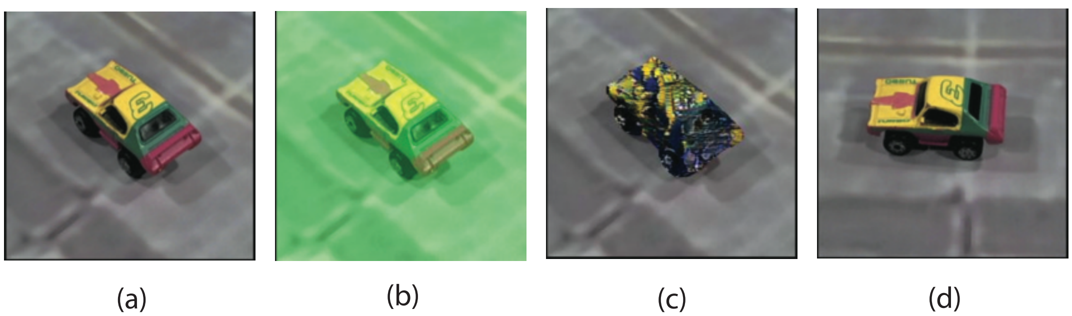


Figure Supp.76: Sample images from the Semantic-iLab dataset. (a) Original image of a toy car placed on a turntable from the original iLab[39] dataset. (b) Light shifted image from Semantic-iLab created by modifying the white balance of the original image. (c) Material shifted image from Semantic-iLab created by modifying the original image by first detecting the foreground object mask, and overlaying the style transferred image on this mask. (d) Viewpoint shifted image of the same object from the iLab dataset.

Figure Supp.77: Images from MacaqueITBench.

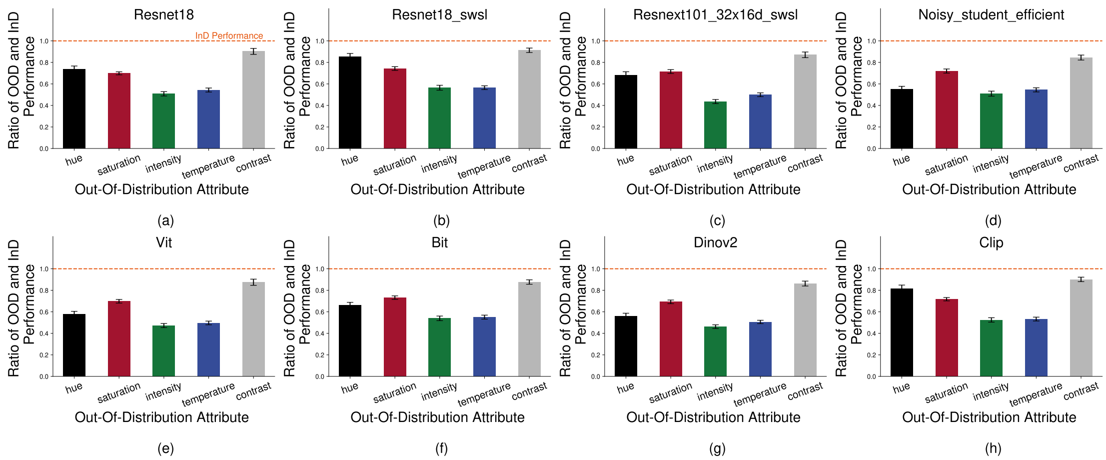


Figure Supp.78: Neural predictivity drops for Mid hold-out strategy as well. For all architectures, across multiple OOD shifts, performance on OOD is worse than in-distribution samples for the Mid hold-out strategy as well.

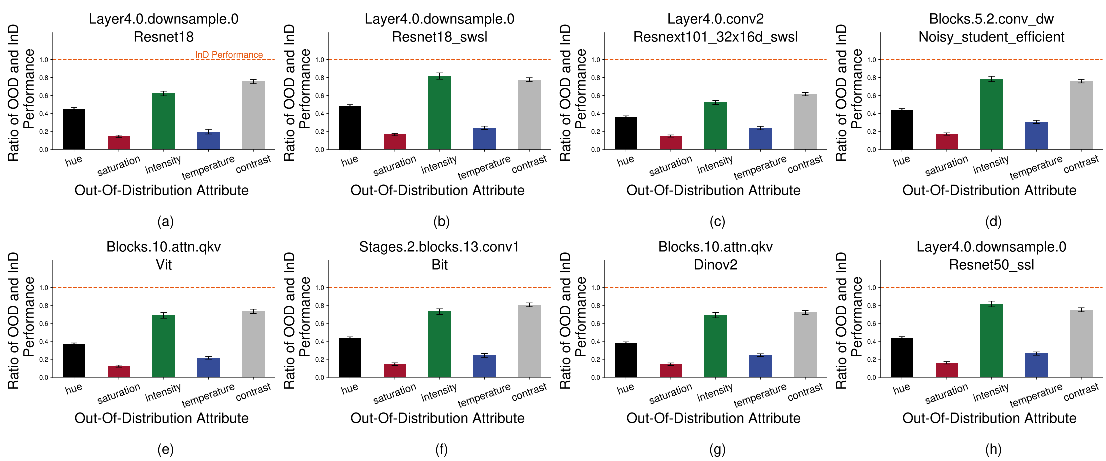


Figure Supp.79: Neural predictivity drops for low hold-out strategy for intermediate layer features as well. For all architectures, across multiple OOD shifts, performance on OOD is worse than in-distribution samples for the low hold-out strategy for image features extracted from intermediate DNN layers as well.

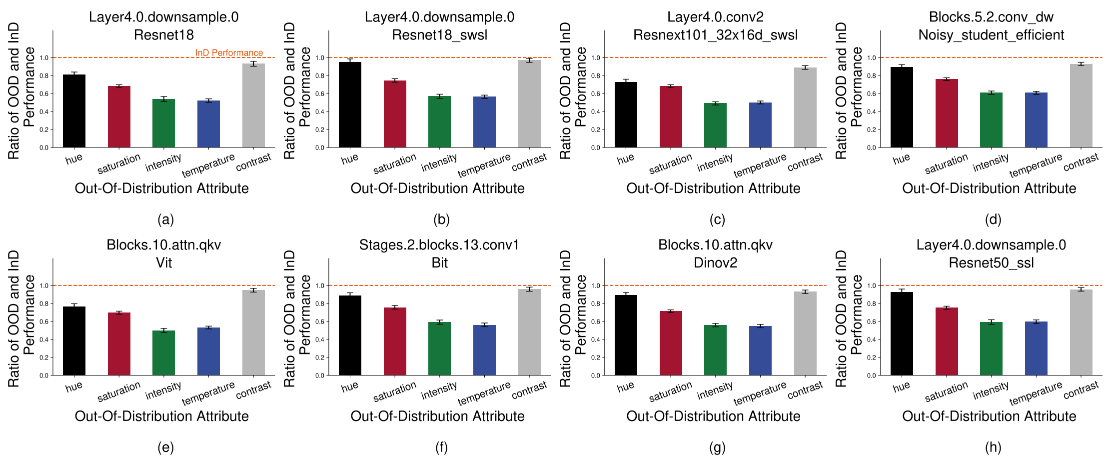


Figure Supp.80: Neural predictivity drops for mid hold-out strategy for intermediate layer features as well. For all architectures, across multiple OOD shifts, performance on OOD is worse than in-distribution samples for the mid hold-out strategy for image features extracted from intermediate DNN layers as well.

References

[ama] Amazon rekognition. https://aws.amazon.com/rekognition/. Accessed: 2022-10-30.

[vis] Google vision api. https://cloud.google.com/vision. Accessed: 2022-10-30.

[mic] Microsoft azure cognitive api. https://azure.microsoft.com/en-us/products/cognitive-services/. Accessed: 2022-10-30.

[ros] Robot Operating System (ROS). http://www.ros.org.

[5] (2023). The cnbc news: Ai tools such as chatgpt are generating a mammoth increase in malicious phishing emails. https://www.cnbc.com/2023/11/28/ai-like-chatgpt-is-creating-huge-increase-in-malicious-phishing-email.html.

[6] (2023). The straits times news: Stem the scams: Beware the bots to avoid being distraught. https://www.straitstimes.com/tech/stem-the-scams-beware-the-bots-to-avoid-being-distraught.

[7] Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. (2023). Gpt-4 technical report. arXiv preprint arXiv:2303.08774.

[8] Achille, A. & Soatto, S. (2018). Emergence of invariance and disentanglement in deep representations. Journal of Machine Learning Research, 19(1), 1947–1980.

[9] Afifi, A. J., Hellwich, O., & Soomro, T. A. (2018). Simultaneous object classification and viewpoint estimation using deep multi-task convolutional neural network. In Proceedings of the International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP) (pp. 177–184).

[10] Aggarwal, C. C. & Yu, P. S. (2001). Outlier detection for high dimensional data. In Proceedings of the 2001 ACM SIGMOD international conference on Management of data (pp. 37–46).

[11] Agrawal, H., Desai, K., Wang, Y., Chen, X., Jain, R., Johnson, M., Batra, D., Parikh, D., Lee, S., & Anderson, P. (2019). Nocaps: Novel object captioning at scale. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 8948–8957).

[12] Aher, G., Arriaga, R. I., & Kalai, A. T. (2023). Using large language models to simulate multiple humans and replicate human subject studies. arXiv.

[13] Ahmed, O., Träuble, F., Goyal, A., Neitz, A., Wüthrich, M., Bengio, Y., Schölkopf, B., & Bauer, S. (2020). Causalworld: A robotic manipulation benchmark for causal structure and transfer learning. CoRR, abs/2010.04296.

[14] Alcorn, M. A., Li, Q., Gong, Z., Wang, C., Mai, L., Ku, W.-S., & Nguyen, A. (2019). Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 4845–4854).

[15] Almazrouei, E., Alobeidli, H., Alshamsi, A., Cappelli, A., Cojocaru, R., Debbah, M., Goffinet, É., Hesslow, D., Launay, J., Malartic, Q., et al. (2023). The falcon series of open language models. arXiv preprint arXiv:2311.16867.

[16] Angtian, W., Kortylewski, A., & Yuille, A. (2021). Nemo: Neural mesh models of contrastive features for robust 3d pose estimation. In Proc of the Int Conf on Learning Representations.

[17] Anthropic (2024). Anthropic. claude 3 haiku: our fastest model yet.

[18] Arcaro, M. J., Schade, P. F., Vincent, J. L., Ponce, C. R., & Livingstone, M. S. (2017). Seeing faces is necessary for face-domain formation. Nature neuroscience, 20(10), 1404–1412.

[19] Arjovsky, M., Bottou, L., Gulrajani, I., & Lopez-Paz, D. (2019). Invariant risk minimization. arXiv preprint arXiv:1907.02893.

[20] Azulay, A. & Weiss, Y. (2019). Why do deep convolutional networks generalize so poorly to small image transformations? Journal of Machine Learning Research, 20(184), 1–25.

[21] Balaji, Y., Sankaranarayanan, S., & Chellappa, R. (2018). Metareg: Towards domain generalization using meta-regularization. Advances in neural information processing systems, 31.

[22] Bansal, Y., Ghorbani, B., Garg, A., Zhang, B., Cherry, C., Neyshabur, B., & Firat, O. (2022). Data scaling laws in NMT: The effect of noise and architecture. In Proceedings of the 39th International Conference on Machine Learning (ICML) (pp. 1466–1482).

[23] Bapst, V., Sanchez-Gonzalez, A., Doersch, C., Stachenfeld, K. L., Kohli, P., Battaglia, P. W., & Hamrick, J. B. (2019). Structured agents for physical construction. CoRR, abs/1904.03177.

[24] Barbu, A., Mayo, D., Alverio, J., Luo, W., Wang, C., Gutfreund, D., Tenenbaum, J., & Katz, B. (2019). ObjectNet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In Advances in Neural Information Processing Systems (pp. 9448–9458).

[25] Bardes, A., Ponce, J., & LeCun, Y. (2021). Vicreg: Variance-invariance-covariance regularization for self-supervised learning. arXiv preprint arXiv:2105.04906.

[26] Bashivan, P., Kar, K., & DiCarlo, J. J. (2019). Neural population control via deep image synthesis. Science, 364(6439), eaav9436.

[27] Battaglia, P., Pascanu, R., Lai, M., Rezende, D. J., et al. (2016). Interaction networks for learning about objects, relations and physics. In Advances in neural information processing systems (pp. 4502–4510).

[28] Bau, D., Zhou, B., Khosla, A., Oliva, A., & Torralba, A. (2017). Network dissection: Quantifying interpretability of deep visual representations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 6541–6549).

[29] Bäuerle, N. & Glauner, A. (2022). Distributionally robust markov decision processes and their connection to risk measures. Mathematics of Operations Research, 47(3), 1757–1780.

[30] Beery, S., Van Horn, G., & Perona, P. (2018). Recognition in terra incognita. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 456–473).

[31] Belder, A., Yefet, G., Ben-Itzhak, R., & Tal, A. (2022). Random walks for adversarial meshes. In ACM SIGGRAPH 2022 Conference Proceedings (pp. 1–9).

[32] Biedenkapp, A., Bozkurt, H. F., Eimer, T., Hutter, F., & Lindauer, M. (2020). Dynamic algorithm configuration: Foundation of a new meta-algorithmic framework. In Proceedings of the Twenty-fourth European Conference on Artificial Intelligence (ECAI'20).

[33] Blanchard, G., Deshmukh, A. A., Dogan, U., Lee, G., & Scott, C. (2017a). Domain generalization by marginal transfer learning. arXiv preprint arXiv:1711.07910.

[34] Blanchard, G., Deshmukh, A. A., Dogan, U., Lee, G., & Scott, C. (2017b). Domain generalization by marginal transfer learning. arXiv preprint arXiv:1711.07910.

[35] Blanz, V., Tarr, M. J., & Bülthoff, H. H. (1999). What object attributes determine canonical views? Perception, 28(5), 575–599.

[36] Block, N. (1981). Behaviourism and psychologism. Philosophical Review, 90(5), 43.

[37] Bloomberg, D. S. & Leptonica (2008). Color quantization using modified median cut.

[38] Bomatter, P., Zhang, M., Karev, D., Madan, S., Tseng, C., & Kreiman, G. (2021). When pigs fly: Contextual reasoning in synthetic and natural scenes.

[39] Borji, A., Izadi, S., & Itti, L. (2016). iLab-20M: A large-scale controlled object dataset to investigate deep learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 2221–2230). The dataset is available at https://bmobear.github.io/projects/viva/.

[40] Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., & Erhan, D. (2016). Domain separation networks. In NeurIPS.

[41] Boyd, A., Bowyer, K. W., & Czajka, A. (2022). Human-aided saliency maps improve generalization of deep learning. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (pp. 2735–2744).

[42] Bricolo, E., Poggio, T., & Logothetis, N. K. (1997). 3D object recognition: A model of view-tuned neurons. In Advances in Neural Information Processing Systems (pp. 41–47).

[43] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. Advances in neural information processing systems, 33, 1877–1901.

[44] Bubeck, S., Lee, Y. T., Price, E., & Razenshteyn, I. (2019). Adversarial examples from computational constraints. In Proceedings of the International Conference on Machine Learning (ICML) (pp. 831–840).

[45] Buzsáki, G., Anastassiou, C. A., & Koch, C. (2012). The origin of extracellular fields and currents—eeg, ecog, lfp and spikes. Nature reviews neuroscience, 13(6), 407–420.

[46] Caballero, E., Gupta, K., Rish, I., & Krueger, D. (2022). Broken neural scaling laws. arXiv preprint arXiv:2210.14891.

[47] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., & Beijbom, O. (2020). nuScenes: A multimodal dataset for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 11618–11628).

[48] Caesar, H., Uijlings, J., & Ferrari, V. (2018). Coco-stuff: Thing and stuff classes in context. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1209–1218).

[49] Canatar, A., Bordelon, B., & Pehlevan, C. (2021). Out-of-distribution generalization in kernel regression. Advances in Neural Information Processing Systems, 34, 12600–12612.

[50] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-end object detection with transformers. In European Conference on Computer Vision (pp. 213–229).: Springer.

[51] Carlucci, F. M., D'Innocente, A., Bucci, S., Caputo, B., & Tommasi, T. (2019). Domain generalization by solving jigsaw puzzles. In CVPR.

[52] Caruana, R. (1997). Multitask learning. Machine learning, 28(1), 41–75.

[53] Casper, S., Boix, X., D'Amario, V., Guo, L., Schrimpf, M., Vinken, K., & Kreiman, G. (2021). Frivolous units: Wider networks are not really that wide. In Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI).

[54] Cederborg, T., Grover, I., Isbell, C. L., & Thomaz, A. L. (2015). Policy shaping with human teachers. In International Joint Conference on Artificial Intelligence.

[55] Chaman, A. & Dokmanić, I. (2021). Truly shift-invariant convolutional neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 3773–3783).

[56] Chaman, A. & Dokmanic, I. (2021). Truly shift-invariant convolutional neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 3773–3783).

[57] Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., & Yu, F. (2015a). ShapeNet: An Information-Rich 3D Model Repository. Technical Report

arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago.

[58] Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., & Yu, F. (2015b). ShapeNet: An information-rich 3D model repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago.

[59] Chattopadhyay, P., Balaji, Y., & Hoffman, J. (2020a). Learning to balance specificity and invariance for in and out of domain generalization. In ECCV.

[60] Chattopadhyay, P., Balaji, Y., & Hoffman, J. (2020b). Learning to balance specificity and invariance for in and out of domain generalization. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16 (pp. 301–318).: Springer.

[61] Chen, D., Zhu, Y., Zhang, J., Du, Y., Li, Z., Liu, Q., Wu, S., & Wang, L. (2024). Uncovering neural scaling laws in molecular representation learning. Advances in Neural Information Processing Systems, 36.

[62] Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2018). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence, 40(4), 834–848.

[63] Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In International conference on machine learning (pp. 1597–1607).: PMLR.

[64] Chen, X., Fang, H., Lin, T.-Y., Vedantam, R., Gupta, S., Dollár, P., & Zitnick, C. L. (2015). Microsoft coco captions: Data collection and evaluation server. arXiv preprint arXiv:1504.00325.

[65] Chen, X. & He, K. (2021). Exploring simple siamese representation learning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 15750–15758).

[66] Choi, M. J., Torralba, A., & Willsky, A. S. (2012). Context models and out-of-context objects. Pattern Recognition Letters, 33(7), 853–862.

[67] Choi, W. & Savarese, S. (2012). A unified framework for multi-target tracking and collective activity recognition. In European Conference on Computer Vision (pp. 215–230).: Springer.

[68] Chollet, F. (2019). On the measure of intelligence. arXiv.

[69] Clerkin, E. M., Hart, E., Rehg, J. M., Yu, C., & Smith, L. B. (2017). Real-world visual statistics and infants' first-learned object names. Philosophical Transactions of the Royal Society B: Biological Sciences, 372(1711), 20160055.

[70] Cobbe, K., Hesse, C., Hilton, J., & Schulman, J. (2020). Leveraging procedural generation to benchmark reinforcement learning.

[71] Cobbe, K., Klimov, O., Hesse, C., Kim, T., & Schulman, J. (2019). Quantifying generalization in reinforcement learning.

[72] Cohen, T. S., Geiger, M., Köhler, J., & Welling, M. (2018). Spherical CNNs. In Proceedings of the International Conference on Learning Representations (ICLR).

[73] Cohen, T. S., Weiler, M., Kicanaoglu, B., & Welling, M. (2019). Gauge equivariant convolutional networks and the Icosahedral CNN. In Proceedings of the International Conference on Machine Learning (ICML) (pp. 1321–1330).

[74] Colby, K. M. (1981). Modeling a paranoid mind. Behavioral and Brain Sciences, 4(4), 515–534.

[75] Colby, K. M., Weber, S., & Hilf, F. D. (1971). Artificial paranoia. Artificial Intelligence, 2(1), 1–25.

[76] Collins, E. & Ghahramani, Z. (2021). Lamda: our breakthrough conversation technology. The Keyword, May, 18.

[77] Community, B. O. (Accessed 2020). Blender - a 3D modelling and rendering package. Blender Foundation, Stichting Blender Foundation, Amsterdam.

[78] Cooper, A., Boix, X., Harari, D., Madan, S., Pfister, H., Sasaki, T., & Sinha, P. (2021). To which out-of-distribution object orientations are dnns capable of generalizing? arXiv preprint arXiv:2109.13445.

[79] Cortes, C. & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273–297.

[80] Croce, F., Rebuffi, S.-A., Shelhamer, E., & Gowal, S. (2023). Seasoning model soups for robustness to adversarial and natural distribution shifts. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 12313–12323).

[81] Cui, Y., Yang, G., Veit, A., Huang, X., & Belongie, S. (2018). Learning to evaluate image captioning. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 5804–5812).

[82] Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T., & Nießner, M. (2017). Scannet: Richly-annotated 3d reconstructions of indoor scenes. In Proc. Computer Vision and Pattern Recognition (CVPR), IEEE.

[83] Dalal, N. & Triggs, B. (2005). Histograms of oriented gradients for human detection. In 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), volume 1 (pp. 886–893).: Ieee.

[84] Damassino, N. (2020). The questioning turing test. Minds and Machines, 30(4), 563–587.

[85] Daw, N. & Wyatt, H. (1976). Kittens reared in a unidirectional environment: evidence for a critical period. The Journal of physiology, 257(1), 155–170.

[86] De Winter, J. C. (2019). Using the student's t-test with extremely small sample sizes. Practical Assessment, Research, and Evaluation, 18(1), 10.

[87] DeNero, J., Klein, D., & Abbeel, P. (2014). Cs188: Berkeley pacman projects. http://ai.berkeley.edu/home.html (Spring 2014).

[88] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 248–255).

[89] Deng, Z., Vahdat, A., Hu, H., & Mori, G. (2016). Structure inference machines: Recurrent neural networks for analyzing relations in group activity recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 4772–4781).

[90] Denkowski, M. & Lavie, A. (2014). Meteor universal: Language specific translation evaluation for any target language. In Proceedings of the ninth workshop on statistical machine translation (pp. 376–380).

[DENSO Wave] DENSO Wave. COBOTTA. https://www.denso-wave.com/en/robot/product/collabo/cobotta.html.

[92] Desimone, R., Albright, T. D., Gross, C. G., & Bruce, C. (1984). Stimulus-selective properties of inferior temporal neurons in the macaque. Journal of Neuroscience, 4(8), 2051–2062.

[93] Devin, C., Abbeel, P., Darrell, T., & Levine, S. (2018). Deep object-centric representations for generalizable robot learning. In 2018 IEEE International Conference on Robotics and Automation (ICRA) (pp. 7111–7118).: IEEE.

[94] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[95] Diochnos, D., Mahloujifar, S., & Mahmoody, M. (2018). Adversarial risk and robustness: General definitions and implications for the uniform distribution. In Advances in Neural Information Processing Systems, volume 31.

[96] Divon, G. & Tal, A. (2018). Viewpoint estimation—insights & model. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 252–268).

[97] Divvala, S. K., Hoiem, D., Hays, J. H., Efros, A. A., & Hebert, M. (2009). An empirical study of context in object detection. In Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on (pp. 1271–1278).: IEEE.

[98] Dohmatob, E. (2019). Limitations of adversarial robustness: strong no free lunch theorem. In Proceedings of the 36th International Conference on Machine Learning (ICML) (pp. 1646–1654).

[99] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In Proceedings of the International Conference on Learning Representations (ICLR).

[100] Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., & Koltun, V. (2017). CARLA: An open urban driving simulator. In Proceedings of the Annual Conference on Robot Learning (CoRL) (pp. 1–16).

[101] Dou, Q., Castro, D. C., Kamnitsas, K., & Glocker, B. (2019). Domain generalization via model-agnostic learning of semantic features. In NeurIPS.

[102] Dulac-Arnold, G., Mankowitz, D. J., & Hester, T. (2019). Challenges of real-world reinforcement learning. CoRR, abs/1904.12901.

[103] Dvornik, N., Mairal, J., & Schmid, C. (2018). Modeling visual context is key to augmenting object detection datasets. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 364–380).

[104] Elhoseiny, M., El-Gaaly, T., Bakry, A., & Elgammal, A. (2016). A comparative analysis and study of multiview CNN models for joint object categorization and pose estimation. In Proceedings of the International Conference on Machine Learning (ICML) (pp. 888–897).

[105] Engstrom, L., Tran, B., Tsipras, D., Schmidt, L., & Madry, A. (2019). Exploring the landscape of spatial robustness. In Proceedings of the International Conference on Machine Learning (ICML) (pp. 1802–1811).

[106] Epic Games. Automotive Materials. https://www.unrealengine.com/marketplace/ja/product/automotive-material-pack.

[107] Epic Games. Unreal Engine. https://www.unrealengine.com.

[108] Erfani, S., Baktashmotlagh, M., Moshtaghi, M., Nguyen, X., Leckie, C., Bailey, J., & Kotagiri, R. (2016). Robust domain generalisation by enforcing distribution invariance. In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16) (pp. 1455–1461).: AAAI Press.

[Everett Gunther] Everett Gunther. Ultra Dynamic Sky. https://www.unrealengine.com/marketplace/product/ultra-dynamic-sky.

[110] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2012). The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[111] Fawzi, A., Fawzi, H., & Fawzi, O. (2018). Adversarial vulnerability for any classifier. In Advances in Neural Information Processing Systems, volume 31.

[112] Fawzi, A., Moosavi-Dezfooli, S.-M., & Frossard, P. (2016). Robustness of classifiers: from adversarial to random noise. In Advances in Neural Information Processing Systems, volume 29.

[113] Filos, A., Tigas, P., McAllister, R., Rhinehart, N., Levine, S., & Gal, Y. (2020). Can autonomous vehicles identify, recover from, and adapt to distribution shifts? CoRR, abs/2006.14911.

[114] Ford, N., Gilmer, J., Carlini, N., & Cubuk, E. D. (2019). Adversarial examples are a natural consequence of test error in noise. In Proceedings of the 36th International Conference on Machine Learning (ICML) (pp. 2280–2289).

[115] Fox News (2024). Captain sully didn't train for water landing before putting us airways jet down in hudson river 15 years ago. Accessed: 2024-08-31.

[116] Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., & Lempitsky, V. (2016). Domain-adversarial training of neural networks. JMLR, 17(1), 2096–2030.

[117] Gao, C. A., Howard, F. M., Markov, N. S., Dyer, E. C., Ramesh, S., Luo, Y., & Pearson, A. T. (2022). Comparing scientific abstracts generated by chatgpt to original abstracts using an artificial intelligence output detector, plagiarism detector, and blinded human reviewers. bioRxiv.

[118] Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A., & Brendel, W. (2018a). Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. arXiv preprint arXiv:1811.12231.

[119] Geirhos, R., Temme, C. R., Rauber, J., Schütt, H. H., Bethge, M., & Wichmann, F. A. (2018b). Generalisation in humans and deep neural networks. Advances in neural information processing systems, 31.

[120] Ghiasi, G., Cui, Y., Srinivas, A., Qian, R., Lin, T.-Y., Cubuk, E. D., Le, Q. V., & Zoph, B. (2020). Simple copy-paste is a strong data augmentation method for instance segmentation.

[121] Ghifary, M., Bastiaan Kleijn, W., Zhang, M., & Balduzzi, D. (2015). Domain generalization for object recognition with multi-task autoencoders. In ICCV.

[122] Ghodrati, A., Pedersoli, M., & Tuytelaars, T. (2014). Is 2D information enough for viewpoint estimation? In Proceedings of the British Machine Vision Conference (BMVC).

[123] Giles, C. L. & Maxwell, T. (1987). Learning, invariance, and generalization in high-order neural networks. Applied Optics, 26(23), 4972–4978.

[124] Gilmer, J., Metz, L., Faghri, F., Schoenholz, S. S., Raghu, M., Wattenberg, M., & Goodfellow, I. (2018). The relationship between high-dimensional geometry and adversarial examples. arXiv preprint, arXiv:1801.02774.

[125] Glorot, X. & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In AISTATS.

[126] Goldstein, H., Poole, C., & Safko, J. (2002). Classical mechanics. Addison-Wesley, 3rd edition.

[127] Gomez, P., Shutter, J., & Rouder, J. N. (2008). Memory for objects in canonical and noncanonical viewpoints. Psychonomic Bulletin & Review, 15(5), 940–944.

[128] Gonfaus, J. M., Boix, X., Van de Weijer, J., Bagdanov, A. D., Serrat, J., & Gonzalez, J. (2010). Harmony potentials for joint classification and segmentation. In Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on (pp. 3280–3287).: IEEE.

[129] Goodfellow, I., Lee, H., Le, Q. V., Saxe, A., & Ng, A. Y. (2009). Measuring invariances in deep networks. In Advances in Neural Information Processing Systems (pp. 646–654).

[130] Gopalakrishnan, K., Hedayatnia, B., Chen, Q., Gottardi, A., Kwatra, S., Venkatesh, A., Gabriel, R., & Hakkani-Tür, D. (2019). Topical-Chat: Towards Knowledge-Grounded Open-Domain Conversations. In Proc. Interspeech 2019 (pp. 1891–1895).

[131] Gopalan, N., Littman, M., MacGlashan, J., Squire, S., Tellex, S., Winder, J., Wong, L., et al. (2017). Planning with abstract markov decision processes. In Proceedings of the International Conference on Automated Planning and Scheduling, volume 27 (pp. 480–488).

[132] Goyal, V. & Grand-Clement, J. (2023). Robust markov decision processes: Beyond rectangularity. Mathematics of Operations Research, 48(1), 203–226.

[133] Grabner, A., Roth, P. M., & Lepetit, V. (2018). 3D pose estimation and 3D model retrieval for objects in the wild. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 3022–3031).

[134] Graef, P. D. (1992). Scene-context effects and models of real-world perception. In Eye movements and visual cognition (pp. 243–259). Springer.

[135] Greene, M. R. (2013). Statistics of high-level scene context. Frontiers in psychology, 4, 777.

[136] Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., & Smola, A. (2012). A kernel two-sample test. The Journal of Machine Learning Research, 13(1), 723–773.

[137] Gulrajani, I. & Lopez-Paz, D. (2020). In search of lost domain generalization. arXiv preprint arXiv:2007.01434.

[138] Gunderson, K. (1964). The imitation game. Mind, 73(290), 234–245.

[139] Gunderson, K. (1985). Mentality and machines. U of Minnesota Press.

[140] Guo, B., Zhang, X., Wang, Z., Jiang, M., Nie, J., Ding, Y., Yue, J., & Wu, Y. (2023). How close is chatgpt to human experts? comparison corpus, evaluation, and detection. arXiv.

[141] Guo, J., Zhu, X., Zhao, C., Cao, D., Lei, Z., & Li, S. Z. (2020). Learning meta face recognition in unseen domains. arXiv preprint arXiv:2003.07733.

[142] Gupta, S. K., Zhang, M., Wu, C.-C., Wolfe, J., & Kreiman, G. (2021). Visual search asymmetry: Deep nets and humans share similar inherent biases. Advances in Neural Information Processing Systems, 34, 6946–6959.

[143] Guss, W. H., Codel, C., Hofmann, K., Houghton, B., Kuno, N., Milani, S., Mohanty, S., Liebana, D. P., Salakhutdinov, R., Topin, N., et al. (2019). The minerl 2019 competition on sample efficient reinforcement learning using human priors. arXiv preprint arXiv:1904.10079.

[144] Guzik, E. E., Byrge, C., & Gilde, C. (2023). The originality of machines: Ai takes the torrance test. Journal of Creativity, 33(3), 100065.

[145] Hafner, D. (2021). Benchmarking the spectrum of agent capabilities. CoRR, abs/2109.06780.

[146] Halder, S. S., Lalonde, J.-F., & Charette, R. d. (2019). Physics-based rendering for improving robustness to rain. In Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR) (pp. 10203–10212).

[147] Hall, H. K. (2018). Deepfake videos: When seeing isn't believing. Cath. UJL & Tech, 27, 51.

[148] Hansen, N. (2016). The CMA evolution strategy: A tutorial. arXiv preprint, arXiv:1604.00772.

[149] Hansen, N., Akimoto, Y., & Baudis, P. (2019). CMA-ES/pycma on Github. Zenodo, DOI:10.5281/zenodo.2559634.

[150] Hansen, N. & Ostermeier, A. (1996). Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In Proceedings of IEEE International Conference on Evolutionary Computation (pp. 312–317).

[151] Hansen, N. & Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation, 9(2), 159–195.

[152] Harel, J., Koch, C., & Perona, P. (2006). Graph-based visual saliency. Advances in neural information processing systems, 19.

[153] Harman, R. & Lacko, V. (2010). On decompositional algorithms for uniform sampling from n-spheres and n-balls. Journal of Multivariate Analysis, 101(10), 2297–2304.

[154] Harnad, S. (1989). Minds, machines and searle. Journal of Experimental & Theoretical Artificial Intelligence, 1(1), 5–25.

[155] Harnad, S. (1991). Other bodies, other minds: A machine incarnation of an old philosophical problem. Minds and Machines, 1(1), 43–54.

[156] Harnad, S. (1994). Levels of functional equivalence in reverse bioengineering. Artificial life, 1(3), 293–301.

[157] Harnad, S. (1999). Turing on reverse-engineering the mind. Journal of Logic, Language, and Information.

[158] Hassabis, D., Kumaran, D., Summerfield, C., & Botvinick, M. (2017). Neuroscience-inspired artificial intelligence. Neuron, 95(2), 245–258.

[159] Hayes, P. & Ford, K. (1995). Turing test considered harmful. In IJCAI (1) (pp. 972–977).: Citeseer.

[160] He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 9729–9738).

[161] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (pp. 2961–2969).

[162] He, K., Zhang, X., Ren, S., & Sun, J. (2016a). Deep residual learning for image recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 770–778).

[163] He, K., Zhang, X., Ren, S., & Sun, J. (2016b). Deep residual learning for image recognition. In CVPR.

[164] Helm, H., Priebe, C. E., & Yang, W. (2023). A statistical turing test for generative models. arXiv.

[165] Henderson, J. M. (1992). Object identification in context: the visual processing of natural scenes. Canadian Journal of Psychology/Revue canadienne de psychologie, 46(3), 319.

[166] Hendrycks, D. & Dietterich, T. (2019). Benchmarking neural network robustness to common corruptions and perturbations. In Proceedings of the International Conference on Learning Representations (ICLR).

[167] Hendrycks, D., Mu, N., Cubuk, E. D., Zoph, B., Gilmer, J., & Lakshminarayanan, B. (2019). Augmix: A simple data processing method to improve robustness and uncertainty. arXiv preprint arXiv:1912.02781.

[168] Hodge, V. & Austin, J. (2004). A survey of outlier detection methodologies. Artificial intelligence review, 22, 85–126.

[169] Hold-Geoffroy, Y., Athawale, A., & Lalonde, J.-F. (2019). Deep sky modeling for single image outdoor lighting estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 6927–6935).

[170] Hu, H., Zhou, G.-T., Deng, Z., Liao, Z., & Mori, G. (2016). Learning structured inference neural networks with label relations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 2960–2968).

[171] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 4700–4708).

[172] Huang, X. & Belongie, S. (2017a). Arbitrary style transfer in real-time with adaptive instance normalization. In Proceedings of the IEEE international conference on computer vision (pp. 1501–1510).

[173] Huang, X. & Belongie, S. (2017b). Arbitrary style transfer in real-time with adaptive instance normalization. In Proceedings of the IEEE international conference on computer vision (pp. 1501–1510).

[174] Hubel, D. H. & Wiesel, T. N. (1964). Effects of monocular deprivation in kittens. Naunyn-Schmiedebergs Archiv für Experimentelle Pathologie und Pharmakologie, 248, 492–497.

[175] Iandola, F., Moskewicz, M., Karayev, S., Girshick, R., Darrell, T., & Keutzer, K. (2014). Densenet: Implementing efficient convnet descriptor pyramids. arXiv preprint arXiv:1404.1869.

[176] Ilse, M., Tomczak, J. M., Louizos, C., & Welling, M. (2020). DIVA: Domain invariant variational autoencoder. In MIDL.

[177] Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., & Madry, A. (2019). Adversarial examples are not bugs, they are features. In Advances in Neural Information Processing Systems.

[Intel] Intel. RealSense D435. https://www.intelrealsense.com/depth-camera-d435.

[179] Ito, P. (1980). 7 robustness of anova and manova test procedures. Handbook of statistics, 1, 199–236.

[180] Jain, L., Chen, S., Wu, W., Jang, U., Chandrasekaran, V., Seshia, S., & Jha, S. (2019). Generating semantic adversarial examples with differentiable rendering. https://openreview.net/forum?id=SJlRF04YwB.

[181] Jannai, D., Meron, A., Lenz, B., Levine, Y., & Shoham, Y. (2023). Human or not? a gamified approach to the turing test. arXiv.

[182] Jia, Y., Zhang, J., Shan, S., & Chen, X. (2020). Single-side domain generalization for face anti-spoofing. In CVPR.

[183] Johnson, S. P. & Aslin, R. N. (1996). Perception of object unity in young infants: The roles of motion, depth, and orientation. Cognitive Development, 11(2), 161–180.

[184] Jones, C. R. & Bergen, B. K. (2024). Does gpt-4 pass the turing test? arXiv.

[185] Joshi, A., Mukherjee, A., Sarkar, S., & Hegde, C. (2019). Semantic adversarial attacks: Parametric transformations that fool deep classifiers. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (pp. 4773–4783).

[186] Juliani, A., Berges, V.-P., Teng, E., Cohen, A., Harper, J., Elion, C., Goy, C., Gao, Y., Henry, H., Mattar, M., et al. (2018). Unity: A general platform for intelligent agents. arXiv preprint arXiv:1809.02627.

[187] Justesen, N., Torrado, R. R., Bontrager, P., Khalifa, A., Togelius, J., & Risi, S. (2018). Illuminating generalization in deep reinforcement learning through procedural level generation. arXiv preprint arXiv:1806.10729.

[188] Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. J. Artif. Intell. Res., 4, 237–285.

[189] Kandel, E. R., Schwartz, J. H., Jessell, T. M., Siegelbaum, S., Hudspeth, A. J., Mack, S., et al. (2000). Principles of neural science, volume 4. McGraw-hill New York.

[190] Kang, K., Belkhale, S., Kahn, G., Abbeel, P., & Levine, S. (2019). Generalization through simulation: Integrating simulated and real data into deep reinforcement learning for vision-based autonomous flight. In 2019 international conference on robotics and automation (ICRA) (pp. 6008–6014).: IEEE.

[191] Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., & Amodei, D. (2020). Scaling laws for neural language models. arXiv, preprint arXiv:2001.08361.

[192] Karpinska, M., Akoury, N., & Iyyer, M. (2021). The perils of using mechanical turk to evaluate open-ended text generation. arXiv preprint arXiv:2109.06835.

[193] Kasai, J., Sakaguchi, K., Dunagan, L., Morrison, J., Bras, R. L., Choi, Y., & Smith, N. A. (2021). Transparent human evaluation for image captioning. arXiv preprint arXiv:2111.08940.

[194] Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., & Krishnan, D. (2020). Supervised contrastive learning. Advances in Neural Information Processing Systems, 33, 18661–18673.

[195] Kim, D., Yoo, Y., Park, S., Kim, J., & Lee, J. (2021). Selfreg: Self-supervised contrastive regularization for domain generalization. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 9619–9628).

[196] Kingma, D. P. & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[197] Kingma, D. P. & Ba, J. (2015). Adam: A method for stochastic optimization. In Proc of the Int Conf on Learning Representations.

[198] Kingma, D. P. & Ba, J. (2017). Adam: A method for stochastic optimization. arXiv preprint, arXiv:1412.6980.

[199] Kirk, R., Zhang, A., Grefenstette, E., & Rocktäschel, T. (2021). A survey of generalisation in deep reinforcement learning. CoRR, abs/2111.09794.

[200] Kobatake, E. & Tanaka, K. (1994). Neuronal selectivities to complex object features in the ventral visual pathway of the macaque cerebral cortex. Journal of neurophysiology, 71(3), 856–867.

[201] Köhler, W. (1960). Dynamics in psychology. WW Norton & Company.

[202] Konkle, T., Brady, T. F., Alvarez, G. A., & Oliva, A. (2010). Conceptual distinctiveness supports detailed visual long-term memory for real-world objects. Journal of experimental Psychology: general, 139(3), 558.

[203] Krause, J., Stark, M., Deng, J., & Fei-Fei, L. (2013). 3D object representations for fine-grained categorization. In Proceedings of the 4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13).

[204] Kreiman, G. (2021). Biological and Computer Vision. Cambridge University Press.

[205] Kriegeskorte, N. (2015). Deep neural networks: a new framework for modeling biological vision and brain information processing. Annual review of vision science, 1, 417–446.

[206] Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. Technical report.

[207] Krueger, D., Caballero, E., Jacobsen, J.-H., Zhang, A., Binas, J., Zhang, D., Le Priol, R., & Courville, A. (2021). Out-of-distribution generalization via risk extrapolation (rex). In International Conference on Machine Learning (pp. 5815–5826).: PMLR.

[208] Kubilius, J., Schrimpf, M., Nayebi, A., Bear, D., Yamins, D. L. K., & DiCarlo, J. J. (2018). CORnet: Modeling the neural mechanisms of core object recognition. bioRxiv preprint, 408385. We used the following implementation in our experiments: https://github.com/dicarlolab/CORnet.

[209] Kümmerer, M., Bethge, M., & Wallis, T. S. (2022). Deepgaze iii: Modeling free-viewing human scanpaths with deep learning. Journal of Vision, 22(5), 7–7.

[210] LaCurts, K. (2011). Criticisms of the turing test and why you should ignore (most of) them. Official blog of MIT's course: Philosophy and theoretical computer science.

[211] Ladicky, L., Russell, C., Kohli, P., & Torr, P. H. (2010). Graph cut based inference with co-occurrence statistics. In European Conference on Computer Vision (pp. 239–253).: Springer.

[212] Larochelle, H., Erhan, D., Courville, A., Bergstra, J., & Bengio, Y. (2007). An empirical evaluation of deep architectures on problems with many factors of variation. In Proceedings of the 24th international conference on Machine learning (pp. 473–480).

[213] Leclerc, G., Salman, H., Ilyas, A., Vemprala, S., Kapoor, A., & Madry, A. (2021). 3DB: A framework for analyzing computer vision models with simulated data. https://github.com/3db/3db.

[214] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278–2324. The dataset is available at http://yann.lecun.com/exdb/mnist/.

[215] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278–2324. The dataset is available at http://yann.lecun.com/exdb/mnist/ (Last access: Oct. 1, 2020).

[216] Lewis, M. B. (2001). The lady's not for turning: Rotation of the Thatcher illusion. Perception, 30(6), 769–774.

[217] Li, C., Bai, J., & Hager, G. D. (2018a). A unified framework for multi-view multi-class object pose estimation. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 254–269).

[218] Li, C., Kreiman, G., & Ramanathan, S. (2024). Discovering neural policies to drive behavior by integrating deep reinforcement learning agents with biological neural networks. Nature Machine Intelligence, (pp. In Press).

[219] Li, C., Wang, J., Zhang, Y., Zhu, K., Hou, W., Lian, J., Luo, F., Yang, Q., & Xie, X. (2023). Large language models understand and can be enhanced by emotional stimuli. arXiv.

[220] Li, D., Yang, Y., Song, Y.-Z., & Hospedales, T. (2018b). Learning to generalize: Meta-learning for domain generalization. In Proceedings of the AAAI conference on artificial intelligence, volume 32.

[221] Li, D., Yang, Y., Song, Y.-Z., & Hospedales, T. M. (2017a). Deeper, broader and artier domain generalization. In Proceedings of the IEEE international conference on computer vision (pp. 5542–5550).

[222] Li, D., Yang, Y., Song, Y.-Z., & Hospedales, T. M. (2017b). Deeper, broader and artier domain generalization. In Proceedings of the IEEE international conference on computer vision (pp. 5542–5550).

[223] Li, D., Yang, Y., Song, Y.-Z., & Hospedales, T. M. (2018c). Learning to generalize: Meta-learning for domain generalization. In AAAI.

[224] Li, H., Pan, S. J., Wang, S., & Kot, A. C. (2018). Domain generalization with adversarial feature learning. In CVPR.

[225] Li, H., Pan, S. J., Wang, S., & Kot, A. C. (2018a). Domain generalization with adversarial feature learning. In CVPR.

[226] Li, H., Pan, S. J., Wang, S., & Kot, A. C. (2018b). Domain generalization with adversarial feature learning. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 5400–5409).

[227] Li, J., Li, D., Xiong, C., & Hoi, S. (2022). Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In ICML.

[228] Li, T.-M., Aittala, M., Durand, F., & Lehtinen, J. (2018c). Differentiable Monte Carlo ray tracing through edge sampling. ACM Transactions on Graphics (TOG), 37(6), 1–11.

[229] Li, Y., Tian, X., Gong, M., Liu, Y., Liu, T., Zhang, K., & Tao, D. (2018d). Deep domain generalization via conditional invariant adversarial networks. In Proceedings of the European conference on computer vision (ECCV) (pp. 624–639).

[230] Li, Z., Shafiei, M., Ramamoorthi, R., Sunkavalli, K., & Chandraker, M. (2020a). Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and svbrdf from a single image. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 2475–2484).

[231] Li, Z., Yu, T.-W., Sang, S., Wang, S., Song, M., Liu, Y., Yeh, Y.-Y., Zhu, R., Gundavarapu, N., Shi, J., et al. (2020b). Openrooms: An end-to-end open framework for photorealistic indoor scene datasets. arXiv preprint arXiv:2007.12868.

[232] Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In Proceedings of the IEEE international conference on computer vision (pp. 2980–2988).

[233] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In European conference on computer vision (pp. 740–755).: Springer.

[234] Liu, H.-T. D., Tao, M., Li, C.-L., Nowrouzezahrai, D., & Jacobson, A. (2019). Beyond pixel norm-balls: Parametric adversaries using an analytically differentiable renderer. In Proceedings of the International Conference on Learning Representations (ICLR).

[235] Logothetis, N. K. & Pauls, J. (1995). Psychophysical and physiological evidence for viewer-centered object representations in the primate. Cerebral Cortex, 5(3), 270–288.

[236] Logothetis, N. K. & Sheinberg, D. L. (1996). Visual object recognition. Annual review of neuroscience, 19(1), 577–621.

[237] Long, M., Cao, Y., Wang, J., & Jordan, M. (2015a). Learning transferable features with deep adaptation networks. In ICML.

[238] Long, M., Cao, Y., Wang, J., & Jordan, M. I. (2015b). Learning transferable features with deep adaptation networks. In ICML.

[239] Mack, S. C. & Eckstein, M. P. (2011). Object co-occurrence serves as a contextual cue to guide and facilitate visual search in a natural viewing environment. Journal of vision, 11(9), 9–9.

[240] Madan, S., Henry, T., Dozier, J., Ho, H., Bhandari, N., Sasaki, T., Durand, F., Pfister, H., & Boix, X. (2022a). When and how convolutional neural networks generalize to out-of-distribution category–viewpoint combinations. Nature Machine Intelligence, 4(2), 146–153.

[241] Madan, S., Henry, T., Dozier, J., Ho, H., Bhandari, N., Sasaki, T., Durand, F., Pfister, H., & Boix, X. (2022b). When and how convolutional neural networks generalize to out-of-distribution category–viewpoint combinations. Nature Machine Intelligence, 4(2), 146 – 153.

[242] Madan, S., Li, Y., Zhang, M., Pfister, H., & Kreiman, G. (2024). Improving generalization by mimicking the human visual diet.

[243] Madan, S., Sasaki, T., Li, T.-M., Boix, X., & Pfister, H. (2021). Small in-distribution changes in 3d perspective and lighting fool both cnns and transformers. arXiv preprint arXiv:2106.16198.

[244] Madan, S., Sasaki, T., Pfister, H., Li, T.-M., & Boix, X. (2023). Adversarial examples within the training distribution: A widespread challenge.

[245] Madan, S., You, L., Zhang, M., Pfister, H., & Kreiman, G. (2022c). What makes domain generalization hard?

[246] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2018). Towards deep learning models resistant to adversarial attacks. In Proceedings of the International Conference on Learning Representations (ICLR).

[247] Mahendran, S., Ali, H., & Vidal, R. (2018). Convolutional networks for object category and 3D pose estimation from 2D images. In Proceedings of the European Conference on Computer Vision Workshops (ECCVW) (pp. 698–715).

[248] Mahloujifar, S., Diochnos, D. I., & Mahmoody, M. (2019). The curse of concentration in robust learning: Evasion and poisoning attacks from concentration of measure. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33 (pp. 4536–4543).

[249] Manhardt, F., Wang, G., Busam, B., Nickel, M., Meier, S., Minciullo, L., Ji, X., & Navab, N. (2020). CPS++: Improving class-level 6D pose and shape estimation from monocular images with self-supervised learning. arXiv preprint arXiv:2003.05848.

[250] Maronga, B., Banzhaf, S., Burmeister, C., Esch, T., Forkel, R., Fröhlich, D., Fuka, V., Gehrke, K. F., Geletič, J., Giersch, S., et al. (2020). Overview of the palm model system 6.0. Geoscientific Model Development, 13(3), 1335–1372.

[Maruf] Maruf, R. Google fires engineer who contended its ai technology was sentient. CNN.

[252] Massa, F., Marlet, R., & Aubry, M. (2016). Crafting a multi-task CNN for viewpoint estimation. In Proceedings of the British Machine Vision Conference (BMVC) (pp. 91.1–91.12).

[253] Mathis, K. M. (1998). Does scene context automatically influence object recognition? Evidence from an interference task. State University of New York at Albany.

[254] McInnes, L., Healy, J., & Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426.

[255] Mei, Q., Xie, Y., Yuan, W., & Jackson, M. O. (2024). A turing test of whether ai chatbots are behaviorally similar to humans. PNAS.

[256] Michaelis, C., Mitzkus, B., Geirhos, R., Rusak, E., Bringmann, O., Ecker, A. S., Bethge, M., & Brendel, W. (2019). Benchmarking robustness in object detection: Autonomous driving when winter is coming. arXiv preprint arXiv:1907.07484.

[257] Min, J., Lee, J., Ponce, J., & Cho, M. (2019). Spair-71k: A large-scale benchmark for semantic correspondence. arXiv preprint arXiv:1908.10543.

[258] Mintun, E., Kirillov, A., & Xie, S. (2021). On interaction between augmentations and corruptions in natural corruption robustness. Advances in Neural Information Processing Systems, 34.

[259] Mokady, R., Hertz, A., & Bermano, A. H. (2021). Clipcap: Clip prefix for image captioning. arXiv preprint arXiv:2111.09734.

[260] Mondal, S. S., Dulberg, Z., & Cohen, J. (2022). Generalization to out-of-distribution transformations.

[261] Moos, J., Hansel, K., Abdulsamad, H., Stark, S., Clever, D., & Peters, J. (2022a). Robust reinforcement learning: A review of foundations and recent advances. Machine Learning and Knowledge Extraction, 4(1), 276–315.

[262] Moos, J., Hansel, K., Abdulsamad, H., Stark, S., Clever, D., & Peters, J. (2022b). Robust reinforcement learning: A review of foundations and recent advances. Machine Learning and Knowledge Extraction, 4(1), 276–315.

[263] Morcos, A. S., Barrett, D. G., Rabinowitz, N. C., & Botvinick, M. (2018). On the importance of single directions for generalization. In Proceedings of the International Conference on Learning Representations (ICLR).

[264] Motiian, S., Piccirilli, M., Adjeroh, D. A., & Doretto, G. (2017). Unified deep supervised domain adaptation and generalization. In Proceedings of the IEEE international conference on computer vision (pp. 5715–5725).

[265] Mu, J., Qiu, W., Hager, G. D., & Yuille, A. L. (2020). Learning from synthetic animals. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 12386–12395).

[266] Muandet, K., Balduzzi, D., & Schölkopf, B. (2013). Domain generalization via invariant feature representation. In International conference on machine learning (pp. 10–18).: PMLR.

[267] Mueller, P., Haegler, S., Ulmer, A., Schubiger, M., Müller Arisona, S., & Weber, B. (Accessed 2020). Esri CityEngine - a 3D city modeling software for urban design, visual effects, and VR/AR. Esri R&D Center Zurich.

[268] Naganuma, H. & Hataya, R. (2023). An empirical investigation of pre-trained model selection for out-of-distribution generalization and calibration. arXiv preprint arXiv:2307.08187.

[269] Nakkiran, P. (2019). Adversarial robustness may be at odds with simplicity. arXiv preprint, arXiv:1901.00532.

[270] Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., & Sutskever, I. (2020). Deep double descent: Where bigger models and more data hurt. In Proceedings of the International Conference on Learning Representations (ICLR).

[271] Neelakantan, A., Xu, T., Puri, R., Radford, A., Han, J. M., Tworek, J., Yuan, Q., Tezak, N., Kim, J. W., Hallacy, C., et al. (2022). Text and code embeddings by contrastive pre-training. arXiv preprint arXiv:2201.10005.

[272] Neil, H. & Dirk, W. (2020). Transformers for image recognition at scale. Online: https://ai. googleblog. com/2020/12/transformers-for-image-recognitionat. html.

[273] Nilim, A. & El Ghaoui, L. (2005). Robust control of markov decision processes with uncertain transition matrices. Operations Research, 53(5), 780–798.

[274] Oliva, A. & Torralba, A. (2006). Building the gist of a scene: The role of global image features in recognition. Progress in brain research, 155, 23–36.

[275] Oliva, A. & Torralba, A. (2007). The role of context in object recognition. Trends in cognitive sciences, 11(12), 520–527.

[276] Olshausen, B. A., Anderson, C. H., & Van Essen, D. C. (1993). A neurobiological model of visual attention and invariant pattern recognition based on dynamic routing of information. Journal of Neuroscience, 13(11), 4700–4719.

[277] OpenAI, Akkaya, I., Andrychowicz, M., Chociej, M., Litwin, M., McGrew, B., Petron, A., Paino, A., Plappert, M., Powell, G., Ribas, R., Schneider, J., Tezak, N., Tworek, J., Welinder, P., Weng, L., Yuan, Q., Zaremba, W., & Zhang, L. (2019). Solving rubik's cube with a robot hand. CoRR, abs/1910.07113.

[278] Oquab, M., Bottou, L., Laptev, I., & Sivic, J. (2015). Is object localization for free?-weakly-supervised learning with convolutional neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 685–694).

[279] Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., et al. (2023). Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193.

[280] Orhan, E., Gupta, V., & Lake, B. M. (2020). Self-supervised learning through the eyes of a child. Advances in Neural Information Processing Systems, 33, 9960–9971.

[281] Osiński, B., Jakubowski, A., Zięcina, P., Miłoś, P., Galias, C., Homoceanu, S., & Michalewski, H. (2020). Simulation-based reinforcement learning for real-world autonomous driving. In 2020 IEEE international conference on robotics and automation (ICRA) (pp. 6411–6418).: IEEE.

[282] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., & Lowe, R. (2022). Training language models to follow instructions with human feedback.

[283] Ozuysal, M., Lepetit, V., & Fua, P. (2009). Pose estimation for category specific multiview object localization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 778–785).

[284] Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting of the Association for Computational Linguistics (pp. 311–318).

[285] Patil, P., Du, J.-H., & Tibshirani, R. J. (2024). Optimal ridge regularization for out-of-distribution prediction. arXiv preprint arXiv:2404.01233.

[286] Penedones, H., Collobert, R., Fleuret, F., & Grangier, D. (2012). Improving object classification using pose information. Technical Report Idiap-RR-30-2012, Idiap Research Institute.

[287] Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., & Wang, B. (2019). Moment matching for multi-source domain adaptation. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 1406–1415).

[288] Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532–1543).

[289] Peyre, J., Laptev, I., Schmid, C., & Sivic, J. (2017). Weakly-supervised learning of visual relations. In ICCV.

[290] Pharr, M., Jakob, W., & Humphreys, G. (2016). Physically based rendering: From theory to implementation. Morgan Kaufmann.

[291] Poggio, T. & Anselmi, F. (2016). Visual cortex and deep networks: learning invariant representations. MIT Press.

[292] Ponce, C. R., Xiao, W., Schade, P. F., Hartmann, T. S., Kreiman, G., & Livingstone, M. S. (2019). Evolving images for visual neurons using a deep generative network reveals coding principles and neuronal preferences. Cell, 177(4), 999–1009.

[293] Posten, H. O. (1982). Two-sample wilcoxon power over the pearson system and comparison with the t-test. Journal of Statistical Computation and Simulation, 16(1), 1–18.

[294] Prato, G., Guiroy, S., Caballero, E., Rish, I., & Chandar, S. (2021). Scaling laws for the out-of-distribution generalization of image classifiers. In ICML 2021 Workshop on Uncertainty and Robustness in Deep Learning.

[295] Puig, X., Ra, K., Boben, M., Li, J., Wang, T., Fidler, S., & Torralba, A. (2018). Virtualhome: Simulating household activities via programs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 8494–8502).

[296] Purtill, R. L. (1971). Beating the imitation game. Mind, 80(318), 290–294.

[297] Qian, R., Meng, T., Gong, B., Yang, M.-H., Wang, H., Belongie, S., & Cui, Y. (2021). Spatiotemporal contrastive video representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 6964–6974).

[298] Qiao, F., Zhao, L., & Peng, X. (2020). Learning to learn single domain generalization. In CVPR.

[299] Qiu, W. & Yuille, A. (2016). UnrealCV: Connecting computer vision to Unreal Engine. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 909–916).

[300] Quiroga, R. Q., Reddy, L., Kreiman, G., Koch, C., & Fried, I. (2005). Invariant visual representation by single neurons in the human brain. Nature, 435(7045), 1102–1107.

[301] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. In Proceedings of the 38th International Conference on Machine Learning (ICML) (pp. 8748–8763).

[302] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. OpenAI blog, 1(8), 9.

[303] Recht, B., Roelofs, R., Schmidt, L., & Shankar, V. (2018). Do CIFAR-10 classifiers generalize to CIFAR-10? arXiv preprint arXiv:1806.00451.

[304] Recht, B., Roelofs, R., Schmidt, L., & Shankar, V. (2019). Do ImageNet classifiers generalize to ImageNet? In Proceedings of the International Conference on Machine Learning (ICML).

[305] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779–788).

[306] Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., & Bowman, S. R. (2023). Gpqa: A graduate-level google-proof q&a benchmark. arXiv.

[307] Ren, J., Liu, P. J., Fertig, E., Snoek, J., Poplin, R., Depristo, M., Dillon, J., & Lakshminarayanan, B. (2019). Likelihood ratios for out-of-distribution detection. Advances in neural information processing systems, 32.

[308] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. arXiv preprint arXiv:1506.01497.

[309] Ren, Y. & Bashivan, P. (2023). How well do models of visual cortex generalize to out of distribution samples? bioRxiv, (pp. 2023–05).

[310] Riesenhuber, M. & Poggio, T. (1998). Just one view: Invariances in inferotemporal cell tuning. In Advances in Neural Information Processing Systems (pp. 215–221).

[311] Rosenfeld, A., Zemel, R., & Tsotsos, J. K. (2018). The elephant in the room. arXiv preprint arXiv:1808.03305.

[312] Ruff, H. A. (1982). Effect of object movement on infants' decision of object structure. Developmental Psychology, 18(3), 462.

[313] Rummery, G. A. & Niranjan, M. (1994). On-line Q-learning using connectionist systems, volume 37. University of Cambridge, Department of Engineering Cambridge, UK.

[314] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision, 115(3), 211–252.

[315] Rust, N. C. & DiCarlo, J. J. (2010). Selectivity and tolerance ("invariance") both increase as visual information propagates from cortical area V4 to IT. Journal of Neuroscience, 30(39), 12978–12995.

[316] Saito, K., Ushiku, Y., & Harada, T. (2017). Asymmetric tri-training for unsupervised domain adaptation. In ICML.

[317] Saito, K., Watanabe, K., Ushiku, Y., & Harada, T. (2018). Maximum classifier discrepancy for unsupervised domain adaptation. In CVPR.

[318] Sakai, A., Sunagawa, T., Madan, S., Suzuki, K., Katoh, T., Kobashi, H., Pfister, H., Sinha, P., Boix, X., & Sasaki, T. (2021). Three approaches to facilitate dnn generalization to objects in out-of-distribution orientations and illuminations: late-stopping, tuning batch normalization and invariance loss. arXiv preprint arXiv:2111.00131.

[319] Sakai, A., Sunagawa, T., Madan, S., Suzuki, K., Katoh, T., Kobashi, H., Pfister, H., Sinha, P., Boix, X., & Sasaki, T. (2022). Three approaches to facilitate invariant neurons and generalization to out-of-distribution orientations and illuminations. Neural Networks, 155, 119–143.

[320] Savarese, S. & Fei-Fei, L. (2007). 3d generic object categorization, localization and pose estimation. In 2007 IEEE 11th International Conference on Computer Vision (pp. 1–8).: IEEE.

[321] Schmidt, L., Santurkar, S., Tsipras, D., Talwar, K., & Madry, A. (2018). Adversarially robust generalization requires more data. In Advances in Neural Information Processing Systems, volume 31.

[322] Schrimpf, M., Kubilius, J., Hong, H., Majaj, N. J., Rajalingham, R., Issa, E. B., Kar, K., Bashivan, P., Prescott-Roy, J., Geiger, F., et al. (2018). Brain-score: Which artificial neural network for object recognition is most brain-like? BioRxiv, (pp. 407007).

[323] Searle, J. R. (1980). Minds, brains, and programs. Behavioral and brain sciences, 3(3), 417–424.

[324] Sejnowski, T. (2022). Large language models and the reverse turing test. arXiv preprint arXiv:2207.14382.

[325] Shafahi, A., Huang, W. R., Studer, C., Feizi, S., & Goldstein, T. (2018). Are adversarial examples inevitable? arXiv preprint, arXiv:1809.02104.

[326] Shahtalebi, S., Gagnon-Audet, J.-C., Laleh, T., Faramarzi, M., Ahuja, K., & Rish, I. (2021). Sand-mask: An enhanced gradient masking strategy for the discovery of invariances in domain generalization. arXiv preprint arXiv:2106.02266.

[327] Shama Sastry, C. & Oore, S. (2019). Detecting out-of-distribution examples with in-distribution examples and gram matrices. arXiv e-prints, (pp. arXiv–1912).

[328] Shamir, A., Melamed, O., & BenShmuel, O. (2021). The dimpled manifold model of adversarial examples in machine learning. arXiv preprint, arXiv:2106.10151.

[329] Shamir, A., Safran, I., Ronen, E., & Dunkelman, O. (2019). A simple explanation for the existence of adversarial examples with small hamming distance. arXiv preprint, arXiv:1901.10861.

[330] Shamsabadi, A. S., Sanchez-Matilla, R., & Cavallaro, A. (2020). Colorfool: Semantic adversarial colorization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 1151–1160).

[331] Shankar, S., Piratla, V., Chakrabarti, S., Chaudhuri, S., Jyothi, P., & Sarawagi, S. (2018). Generalizing across domains via cross-gradient training. arXiv preprint arXiv:1804.10745.

[332] Shao, R., Lan, X., Li, J., & Yuen, P. C. (2019). Multi-adversarial discriminative deep domain generalization for face presentation attack detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 10023–10031).

[333] Sharan, L., Rosenholtz, R., & Adelson, E. (2009). Material perception: What can you see in a brief glance? Journal of Vision, 9(8), 784–784.

[334] Shepard, R. N. & Metzler, J. (1971). Mental rotation of three-dimensional objects. Science, 171(3972), 701–703.

[335] Shibli, A. M., Pritom, M. M. A., & Gupta, M. (2024). Abusegpt: Abuse of generative ai chatbots to create smishing campaigns.

[336] Shin, D., Fowlkes, C. C., & Hoiem, D. (2018). Pixels, voxels, and views: A study of shape representations for single view 3D object shape prediction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 3061–3069).

[337] Shiohara, K. & Yamasaki, T. (2022). Detecting deepfakes with self-blended images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 18720–18729).

[338] Shuster, K., Xu, J., Komeili, M., Ju, D., Smith, E. M., Roller, S., Ung, M., Chen, M., Arora, K., Lane, J., Behrooz, M., Ngan, W., Poff, S., Goyal, N., Szlam, A., Boureau, Y.-L., Kambadur, M., & Weston, J. (2022a). Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage.

[339] Shuster, K., Xu, J., Komeili, M., Ju, D., Smith, E. M., Roller, S., Ung, M., Chen, M., Arora, K., Lane, J., et al. (2022b). Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage. arXiv preprint arXiv:2208.03188.

[340] Siddiqui, S. A., Krueger, D., & Breuel, T. (2023). Investigating the nature of 3d generalization in deep neural networks. arXiv preprint arXiv:2304.09358.

[SilverTm] SilverTm. City Park Environment Collection. https://www.unrealengine.com/marketplace/ja/product/city-park-environment-collection.

[342] Simonyan, K., Vedaldi, A., & Zisserman, A. (2014). Deep inside convolutional networks: Visualising image classification models and saliency maps. In Proceedings of the International Conference on Learning Representations Workshop (ICLRW).

[343] Sinha, A., Namkoong, H., Volpi, R., & Duchi, J. (2017). Certifying some distributional robustness with principled adversarial training. arXiv preprint arXiv:1710.10571.

[344] Sinha, P. & Poggio, T. (1996). Role of learning in three-dimensional form perception. Nature, 384(6608), 460–463.

[345] Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical networks for few-shot learning. Advances in neural information processing systems, 30.

[346] Sohn, K., Shang, W., Yu, X., & Chandraker, M. (2019). Unsupervised domain adaptation for distance metric learning. In ICLR.

[347] Sokolic, J., Giryes, R., Sapiro, G., & Rodrigues, M. (2017). Generalization error of invariant classifiers. In AISTATS.

[348] Spaak, E., Peelen, M. V., & de Lange, F. P. (2022). Scene context impairs perception of semantically congruent objects. Psychological Science, 33(2), 299–313. PMID: 35020519.

[349] Speer, R., Chin, J., Lin, A., Jewett, S., & Nathan, L. (2018). Luminosoinsight/wordfreq: v2.2.

[350] Squire, S., Tellex, S., Arumugam, D., & Yang, L. (2015). Grounding english commands to reward functions. In Robotics: Science and Systems.

[351] Srivastava, A., Rastogi, A., Rao, A., Shoeb, A. A. M., Abid, A., Fisch, A., Brown, A. R., Santoro, A., Gupta, A., Garriga-Alonso, A., Kluska, A., Lewkowycz, A., Agarwal, A., Power, A., Ray, A., Warstadt, A., Kocurek, A. W., Safaya, A., Tazarv, A., Xiang, A., Parrish, A., Nie, A., Hussain, A., Askell, A., Dsouza, A., Slone, A., Rahane, A., Iyer, A. S., Andreassen, A., Madotto, A., Santilli, A., Stuhlmüller, A., Dai, A., La, A., Lampinen, A., Zou, A., Jiang, A., Chen, A., Vuong, A., Gupta, A., Gottardi, A., Norelli, A., Venkatesh, A., Gholamidavoodi, A., Tabassum, A., Menezes, A., Kirubarajan, A., Mullokandov, A., Sabharwal, A., Herrick, A., Efrat, A., Erdem, A., Karakaş, A., Roberts, B. R., Loe, B. S., Zoph, B., Bojanowski, B., Özyurt, B., Hedayatnia, B., Neyshabur, B., Inden, B., Stein, B., Ekmekci, B., Lin, B. Y., Howald, B., Orinion, B., Diao, C., Dour, C., Stinson, C., Argueta, C., Ramírez, C. F., Singh, C., Rathkopf, C., Meng, C., Baral, C., Wu, C., Callison-Burch, C., Waites, C., Voigt, C., Manning, C. D., Potts, C., Ramirez, C., Rivera, C. E., Siro, C., Raffel, C., Ashcraft, C., Garbacea, C., Sileo, D., Garrette, D., Hendrycks, D., Kilman, D., Roth, D., Freeman, D., Khashabi, D., Levy, D., González, D. M., Perszyk, D., Hernandez, D., Chen, D., Ippolito, D., Gilboa, D., Dohan, D., Drakard, D., Jurgens, D., Datta, D., Ganguli, D., Emelin, D., Kleyko, D., Yuret, D., Chen, D., Tam, D., Hupkes, D., Misra, D., Buzan, D., Mollo, D. C., Yang, D., Lee, D.-H., Schrader, D., Shutova, E., Cubuk, E. D., Segal, E., Hagerman, E., Barnes, E., Donoway, E., Pavlick, E., Rodola, E., Lam, E., Chu, E., Tang, E., Erdem, E., Chang, E., Chi, E. A., Dyer, E., Jerzak, E., Kim, E., Manyasi, E. E., Zheltonozhskii, E., Xia, F., Siar, F., Martínez-Plumed, F., Happé, F., Chollet, F., Rong, F., Mishra, G., Winata, G. I., de Melo, G.,

Kruszewski, G., Parascandolo, G., Mariani, G., Wang, G., Jaimovitch-López, G., Betz, G., Gur-Ari, G., Galijasevic, H., Kim, H., Rashkin, H., Hajishirzi, H., Mehta, H., Bogar, H., Shevlin, H., Schütze, H., Yakura, H., Zhang, H., Wong, H. M., Ng, I., Noble, I., Jumelet, J., Geissinger, J., Kernion, J., Hilton, J., Lee, J., Fisac, J. F., Simon, J. B., Koppel, J., Zheng, J., Zou, J., Kocoń, J., Thompson, J., Wingfield, J., Kaplan, J., Radom, J., Sohl-Dickstein, J., Phang, J., Wei, J., Yosinski, J., Novikova, J., Bosscher, J., Marsh, J., Kim, J., Taal, J., Engel, J., Alabi, J., Xu, J., Song, J., Tang, J., Waweru, J., Burden, J., Miller, J., Balis, J. U., Batchelder, J., Berant, J., Frohberg, J., Rozen, J., Hernandez-Orallo, J., Boudeman, J., Guerr, J., Jones, J., Tenenbaum, J. B., Rule, J. S., Chua, J., Kanclerz, K., Livescu, K., Krauth, K., Gopalakrishnan, K., Ignatyeva, K., Markert, K., Dhole, K. D., Gimpel, K., Omondi, K., Mathewson, K., Chiafullo, K., Shkaruta, K., Shridhar, K., McDonell, K., Richardson, K., Reynolds, L., Gao, L., Zhang, L., Dugan, L., Qin, L., Contreras-Ochando, L., Morency, L.-P., Moschella, L., Lam, L., Noble, L., Schmidt, L., He, L., Colón, L. O., Metz, L., Şenel, L. K., Bosma, M., Sap, M., ter Hoeve, M., Farooqi, M., Faruqui, M., Mazeika, M., Baturan, M., Marelli, M., Maru, M., Quintana, M. J. R., Tolkiehn, M., Giulianelli, M., Lewis, M., Potthast, M., Leavitt, M. L., Hagen, M., Schubert, M., Baitemirova, M. O., Arnaud, M., McElrath, M., Yee, M. A., Cohen, M., Gu, M., Ivanitskiy, M., Starritt, M., Strube, M., Swędrowski, M., Bevilacqua, M., Yasunaga, M., Kale, M., Cain, M., Xu, M., Suzgun, M., Walker, M., Tiwari, M., Bansal, M., Aminnaseri, M., Geva, M., Gheini, M., T, M. V., Peng, N., Chi, N. A., Lee, N., Krakover, N. G.-A., Cameron, N., Roberts, N., Doiron, N., Martinez, N., Nangia, N., Deckers, N., Muennighoff, N., Keskar, N. S., Iyer, N. S., Constant, N., Fiedel, N., Wen, N., Zhang, O., Agha, O., Elbaghdadi, O., Levy, O., Evans, O., Casares, P. A. M., Doshi, P., Fung, P., Liang, P. P., Vicol, P., Alipoormolabashi, P., Liao, P., Liang, P., Chang, P., Eckersley, P., Htut, P. M., Hwang, P., Miłkowski, P., Patil, P., Pezeshkpour, P., Oli, P., Mei, Q., Lyu, Q., Chen, Q., Banjade, R., Rudolph, R. E., Gabriel, R., Habacker, R., Risco, R., Millière, R., Garg, R., Barnes, R., Saurous, R. A., Arakawa, R., Raymaekers, R., Frank, R., Sikand, R., Novak, R., Sitelew, R., LeBras, R., Liu, R., Jacobs, R., Zhang, R., Salakhutdinov, R., Chi, R., Lee, R., Stovall, R., Teehan, R., Yang, R., Singh, S., Mohammad, S. M., Anand, S., Dillavou, S., Shleifer, S., Wiseman, S., Gruetter, S., Bowman, S. R., Schoenholz, S. S., Han, S., Kwatra, S., Rous, S. A., Ghazarian, S., Ghosh, S., Casey, S., Bischoff, S., Gehrmann, S., Schuster, S., Sadeghi, S., Hamdan, S., Zhou, S., Srivastava, S., Shi, S., Singh, S., Asaadi, S., Gu, S. S., Pachchigar, S., Toshniwal, S., Upadhyay, S., Shyamolima, Debnath, Shakeri, S., Thormeyer, S., Melzi, S., Reddy, S., Makini, S. P., Lee, S.-H., Torene, S., Hatwar, S., Dehaene, S., Divic, S., Ermon, S., Biderman, S., Lin, S., Prasad, S., Piantadosi, S. T., Shieber, S. M., Misherghi, S., Kiritchenko, S., Mishra, S., Linzen, T., Schuster, T., Li, T., Yu, T., Ali, T., Hashimoto, T., Wu, T.-L., Desbordes, T., Rothschild, T., Phan, T., Wang, T., Nkinyili, T., Schick, T., Kornev, T., Tunduny, T., Gerstenberg, T., Chang, T., Neeraj, T., Khot, T., Shultz, T., Shaham, U., Misra, V., Demberg, V., Nyamai, V., Raunak, V., Ramasesh, V., Prabhu, V. U., Padmakumar, V., Srikumar, V., Fedus, W., Saunders, W., Zhang, W., Vossen, W., Ren, X., Tong, X., Zhao, X., Wu, X., Shen, X., Yaghoobzadeh, Y., Lakretz, Y., Song, Y., Bahri, Y., Choi, Y., Yang, Y., Hao, Y., Chen, Y., Belinkov, Y., Hou, Y., Hou, Y., Bai, Y., Seid, Z., Zhao, Z., Wang, Z., Wang, Z. J., Wang, Z., & Wu, Z. (2023). Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. arXiv.

[352] Srivastava, S., Ben-Yosef, G., & Boix, X. (2019). Minimal images in deep neural networks: Fragile object recognition in natural images. In Proceedings of the International Conference on Learning Representations (ICLR).

[353] Standley, T., Zamir, A. R., Chen, D., Guibas, L., Malik, J., & Savarese, S. (2020). Which tasks should be learned together in multi-task learning? In Proceedings of the International Conference on Machine Learning (ICML).

[354] Stone, A., Ramirez, O., Konolige, K., & Jonschkowski, R. (2021). The distracting control suite – a challenging benchmark for reinforcement learning from pixels.

[355] Stutz, D., Hein, M., & Schiele, B. (2019). Disentangling adversarial robustness and generalization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 6976–6987).

[356] Su, H., Qi, C. R., Li, Y., & Guibas, L. J. (2015). Render for CNN: Viewpoint estimation in images using CNNs trained with rendered 3D model views. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (pp. 2686–2694).

[357] Sugiyama, M. & Kawanabe, M. (2012). Machine learning in non-stationary environments: Introduction to covariate shift adaptation. MIT press.

[358] Sun, B. & Saenko, K. (2016). Deep CORAL: correlation alignment for deep domain adaptation. CoRR, abs/1607.01719.

[359] Sun, J. & Jacobs, D. W. (2017). Seeing what is not there: Learning context to determine where objects are missing. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5716–5724).

[360] Super, H. & Roelfsema, P. R. (2005). Chronic multiunit recordings in behaving animals: advantages and limitations. Progress in brain research, 147, 263–282.

[361] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 2818–2826).

[362] Tanay, T. & Griffin, L. (2016). A boundary tilting persepective on the phenomenon of adversarial examples. arXiv preprint, arXiv:1608.07690.

[363] Tang, H., Schrimpf, M., Lotter, W., Moerman, C., Paredes, A., Caro, J. O., Hardesty, W., Cox, D., & Kreiman, G. (2018). Recurrent computations for visual pattern completion. Proceedings of the National Academy of Sciences, 115(35), 8835–8840.

[364] Tanmay, K., Khandelwal, A., Agarwal, U., & Choudhury, M. (2023). Probing the moral development of large language models through defining issues test. arXiv.

[365] Taylor, J. A. & Ivry, R. B. (2013). Context-dependent generalization. Frontiers in Human Neuroscience, 7, 171.

[366] Team, G., Anil, R., Borgeaud, S., Wu, Y., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., et al. (2023). Gemini: a family of highly capable multimodal models. arXiv preprint arXiv:2312.11805.

[367] Tellex, S., Kollar, T., Dickerson, S., Walter, M., Banerjee, A., Teller, S., & Roy, N. (2011). Understanding natural language commands for robotic navigation and mobile manipulation. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 25 (pp. 1507–1514).

[368] Terhune, K. P., Liu, G. T., Modestino, E. J., Miki, A., Sheth, K. N., Liu, C.-S. J., Bonhomme, G. R., & Haselgrove, J. C. (2005). Recognition of objects in non-canonical views: A functional MRI study. Journal of Neuro-Ophthalmology, 25(4), 273–279.

[369] Thompson, P. (1980). Margaret Thatcher: a new illusion. Perception.

[370] Tibshirani, R. (1996). Regression shrinkage and selection via the Lasso. Journal of the Royal Statistical Society. Series B (Methodological), (pp. 267–288).

[Tiku] Tiku, N. The google engineer who thinks the company's ai has come to life. WashingtonPost.

[372] Torralba, A. (2003). Contextual priming for object detection. International journal of computer vision, 53(2), 169–191.

[373] Torralba, A. & Efros, A. A. (2011). Unbiased look at dataset bias. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 1521–1528).

[374] Torralba, A., Murphy, K. P., & Freeman, W. T. (2005). Contextual models for object detection using boosted random fields. In Advances in neural information processing systems (pp. 1401–1408).

[375] Torralba, A., Murphy, K. P., Freeman, W. T., & Rubin, M. A. (2003). Context-based vision system for place and object recognition. In Computer Vision, IEEE International Conference on, volume 2 (pp. 273–273).: IEEE Computer Society.

[376] Torralba, A., Oliva, A., Castelhano, M. S., & Henderson, J. M. (2006). Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search. Psychological review, 113(4), 766.

[377] Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., & Jégou, H. (2021). Training data-efficient image transformers & distillation through attention. In Proceedings of the 38th International Conference on Machine Learning (ICML) (pp. 10347–10357).

[378] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. (2023). Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971.

[379] Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., & Madry, A. (2019). Robustness may be at odds with accuracy. In Proceedings of the International Conference on Learning Representations (ICLR).

[380] Tucker, R. & Snavely, N. (2020). Single-view view synthesis with multiplane images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 551–560).

[381] Tulsiani, S., Carreira, J., & Malik, J. (2015). Pose induction for novel object categories. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (pp. 64–72).

[382] Tulsiani, S. & Malik, J. (2015). Viewpoints and keypoints. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 1510–1519).

[383] Turing, A. M. (1950). : (pp. 23–65). Springer.

[384] Turk, A. M. (2012). Amazon mechanical turk. Retrieved August, 17, 2012.

[385] Ullman, S. (1996). High-level vision: Object recognition and visual cognition. MIT Press.

[386] Ullman, S. (2019). Using neuroscience to develop artificial intelligence. Science, 363(6428), 692–693.

[387] Varol, G., Romero, J., Martin, X., Mahmood, N., Black, M. J., Laptev, I., & Schmid, C. (2017). Learning from synthetic humans. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[388] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. arXiv preprint arXiv:1706.03762.

[389] Vedantam, R., Lawrence Zitnick, C., & Parikh, D. (2015). Cider: Consensus-based image description evaluation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4566–4575).

[390] Vedantam, R., Lopez-Paz, D., & Schwab, D. J. (2021). An empirical investigation of domain generalization with empirical risk minimizers. Advances in Neural Information Processing Systems, 34.

[391] Venkateswara, H., Eusebio, J., Chakraborty, S., & Panchanathan, S. (2017). Deep hashing network for unsupervised domain adaptation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5018–5027).

[392] Volpi, R., Namkoong, H., Sener, O., Duchi, J. C., Murino, V., & Savarese, S. (2018a). Generalizing to unseen domains via adversarial data augmentation. In NeurIPS.

[393] Volpi, R., Namkoong, H., Sener, O., Duchi, J. C., Murino, V., & Savarese, S. (2018b). Generalizing to unseen domains via adversarial data augmentation. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), Advances in Neural Information Processing Systems, volume 31: Curran Associates, Inc.

[394] Wada, K. (2018). labelme: Image polygonal annotation with python. https://github.com/wkentaro/labelme.

[395] Walter, M. R., Hemachandra, S. M., Homberg, B. S., Tellex, S., & Teller, S. (2013). : Robotics: Science and Systems.

[396] Wang, G., Han, H., Shan, S., & Chen, X. (2020). Cross-domain face presentation attack detection via multi-domain disentangled representation learning.

In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 6678–6687).

[397] Wang, J., Pun, A., Tu, J., Manivasagam, S., Sadat, A., Casas, S., Ren, M., & Urtasun, R. (2021a). AdvSim: Generating safety-critical scenarios for self-driving vehicles. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 9909–9918).

[398] Wang, J., Yang, Z., Hu, X., Li, L., Lin, K., Gan, Z., Liu, Z., Liu, C., & Wang, L. (2022a). Git: A generative image-to-text transformer for vision and language. arXiv preprint arXiv:2205.14100.

[399] Wang, P., Yang, A., Men, R., Lin, J., Bai, S., Li, Z., Ma, J., Zhou, C., Zhou, J., & Yang, H. (2022b). Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In International Conference on Machine Learning (pp. 23318–23340).: PMLR.

[400] Wang, Z., Loog, M., & Van Gemert, J. (2021b). Respecting domain relations: Hypothesis invariance for domain generalization. In 2020 25th International Conference on Pattern Recognition (ICPR) (pp. 9756–9763).: IEEE.

[401] Warde-Farley, D. & Goodfellow, I. (2016). Adversarial perturbations of deep neural networks. In T. Hazan, G. Papandreou, & D. Tarlow (Eds.), Perturbations, Optimization, and Statistics (pp. 311–342). Cambridge, MA, USA: MIT Press.

[402] Watkins, C. J. C. H. & Dayan, P. (1992). Q-learning. Machine Learning, 8(3), 279–292.

[403] Watt, S. (1996). Naive psychology and the inverted turing test. Psycoloquy, 7(14), 463–518.

[404] Weizenbaum, J. (1966). Eliza—a computer program for the study of natural language communication between man and machine. Communications of the ACM, 9(1), 36–45.

[Wertheimer] Wertheimer, T. Blake lemoine: Google fires engineer who said ai tech has feelings. BBC news.

[406] Westerlund, M. (2019). The emergence of deepfake technology: A review. Technology Innovation Management Review, 9(11).

[407] Wiles, O., Gkioxari, G., Szeliski, R., & Johnson, J. (2020). Synsin: End-to-end view synthesis from a single image. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 7467–7477).

[408] Wood, J. N. & Wood, S. M. (2018). The development of invariant object recognition requires visual experience with temporally smooth objects. Cognitive Science, 42(4), 1391–1406.

[409] Wood, J. N. & Wood, S. M. W. (2022). The development of object recognition requires experience with the surface features of objects. bioRxiv, (pp. 2022–12).

[410] Wu, K., Wu, E., & Kreiman, G. (2018). Learning scene gist with convolutional neural networks to improve object recognition. In Information Sciences and Systems (CISS), 2018 52nd Annual Conference on (pp. 1–6).: IEEE.

[411] Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., & Girshick, R. (2019). Detectron2. https://github.com/facebookresearch/detectron2.

[412] Xiang, Y., Mottaghi, R., & Savarese, S. (2014). Beyond pascal: A benchmark for 3D object detection in the wild. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision (pp. 75–82).

[413] Xiang, Y., Schmidt, T., Narayanan, V., & Fox, D. (2018). PoseCNN: A convolutional neural network for 6d object pose estimation in cluttered scenes. In Proceedings of the Robotics: Science and Systems (RSS).

[414] Xiao, C., Yang, D., Li, B., Deng, J., & Liu, M. (2019). Meshadv: Adversarial meshes for visual recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 6898–6907).

[415] Xiao, W., Sharma, S., Kreiman, G., & Livingstone, M. S. (2024). Feature-selective responses in macaque visual cortex follow eye movements during natural vision. Nature Neuroscience, (pp. 1–10).

[416] Xie, C., Tan, M., Gong, B., Wang, J., Yuille, A. L., & Le, Q. V. (2020a). Adversarial examples improve image recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 819–828).

[417] Xie, Q., Luong, M.-T., Hovy, E., & Le, Q. V. (2020b). Self-training with noisy student improves imagenet classification. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 10687–10698).

[418] Xie, Q., Luong, M.-T., Hovy, E., & Le, Q. V. (2020c). Self-training with noisy student improves imagenet classification.

[419] Xie, S., Girshick, R., Dollár, P., Tu, Z., & He, K. (2017). Aggregated residual transformations for deep neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 1492–1500).

[420] Xu, Z., Liu, D., Yang, J., Raffel, C., & Niethammer, M. (2020). Robust and generalizable visual representation learning via random convolutions. arXiv preprint arXiv:2007.13003.

[421] Yalniz, I. Z., Jégou, H., Chen, K., Paluri, M., & Mahajan, D. (2019). Billion-scale semi-supervised learning for image classification. CoRR, abs/1905.00546.

[422] Yamins, D. L. & DiCarlo, J. J. (2016). Using goal-driven deep learning models to understand sensory cortex. Nature neuroscience.

[423] Yamins, D. L., Hong, H., Cadieu, C. F., Solomon, E. A., Seibert, D., & DiCarlo, J. J. (2014). Performance-optimized hierarchical models predict neural responses in higher visual cortex. In Proceedings of the National Academy of Sciences.

[424] Yang, D., Xiao, C., Li, B., Deng, J., & Liu, M. (2018). Realistic adversarial examples in 3d meshes. arXiv preprint arXiv:1810.05206, 2, 2.

[425] Yang, G. R., Joglekar, M. R., Song, H. F., Newsome, W. T., & Wang, X.-J. (2019). Task representations in neural networks trained to perform many cognitive tasks. Nature Neuroscience, 22(2), 297–306.

[426] Yao, J., Fidler, S., & Urtasun, R. (2012). Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation. In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on (pp. 702–709).: IEEE.

[427] Yoon, J. S., Kim, K., Gallo, O., Park, H. S., & Kautz, J. (2020). Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 5336–5345).

[428] Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., & Yoo, Y. (2019). Cutmix: Regularization strategy to train strong classifiers with localizable features. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 6023–6032).

[429] Zagoruyko, S. & Komodakis, N. (2016). Wide residual networks. In Proceedings of the British Machine Vision Conference (BMVC) (pp. 87.1–87.12).

[430] Zaidi, S. S. A., Boix, X., Prasad, N., Gilad-Gutnick, S., Ben-Ami, S., & Sinha, P. (2020). Is robustness to transformations driven by invariant neural representations? arXiv preprint, arXiv:2007.00112.

[431] Zeiler, M. D. & Fergus, R. (2014). Visualizing and understanding convolutional networks. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 818–833).

[432] Zellers, R., Holtzman, A., Rashkin, H., Bisk, Y., Farhadi, A., Roesner, F., & Choi, Y. (2019). Defending against neural fake news. Advances in neural information processing systems, 32.

[433] Zeng, X., Liu, C., Wang, Y.-S., Qiu, W., Xie, L., Tai, Y.-W., Tang, C.-K., & Yuille, A. L. (2019). Adversarial attacks beyond the image space. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 4302–4311).

[434] Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2017a). mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.09412.

[435] Zhang, M., Armendariz, M., Xiao, W., Rose, O., Bendtz, K., Livingstone, M., Ponce, C., & Kreiman, G. (2022). Look twice: A generalist model predicts return fixations across tasks and species. PLoS Computational Biology, (pp. In Press).

[436] Zhang, M., Feng, J., Ma, K. T., Lim, J. H., Zhao, Q., & Kreiman, G. (2018). Finding any waldo with zero-shot invariant and efficient visual search. Nature communications, 9(1), 3730.

[437] Zhang, M., Tseng, C., & Kreiman, G. (2020). Putting visual object recognition in context. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 12985–12994).

[438] Zhang, M., Wang, T., Lim, J. H., Kreiman, G., & Feng, J. (2019a). Variational prototype replays for continual learning. arXiv preprint arXiv:1905.09447.

[439] Zhang, Q., Guo, Q., Gao, R., Juefei-Xu, F., Yu, H., & Feng, W. (2021). Adversarial relighting against face recognition. arXiv preprint arXiv:2108.07920.

[440] Zhang, R. (2019). Making convolutional networks shift-invariant again. In Proceedings of the International Conference on Machine Learning (ICML) (pp. 7324–7334).

[441] Zhang, Y., Song, S., Yumer, E., Savva, M., Lee, J.-Y., Jin, H., & Funkhouser, T. (2017b). Physically-based rendering for indoor scene understanding using convolutional neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 5287–5295).

[442] Zhang, Y., Sun, S., Galley, M., Chen, Y.-C., Brockett, C., Gao, X., Gao, J., Liu, J., & Dolan, B. (2019b). Dialogpt: Large-scale generative pre-training for conversational response generation. arXiv preprint arXiv:1911.00536.

[443] Zhao, J. & Itti, L. (2017). Improved deep learning of object category using pose information. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV) (pp. 550–559).

[444] Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2015). Object detectors emerge in deep scene CNNs. In Proceedings of the International Conference on Learning Representations (ICLR).

[445] Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., & Torralba, A. (2017). Places: A 10 million image database for scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence.

[446] Zhou, B., Sun, Y., Bau, D., & Torralba, A. (2018). Revisiting the importance of individual units in cnns via ablation. arXiv preprint arXiv:1806.02891.

[447] Zhou, K., Liu, Z., Qiao, Y., Xiang, T., & Loy, C. C. (2021). Domain generalization in vision: A survey. arXiv preprint arXiv:2103.02503.

[448] Zhou, T., Tulsiani, S., Sun, W., Malik, J., & Efros, A. A. (2016). View synthesis by appearance flow. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 286–301).

[449] Zhu, Y., Wong, J., Mandlekar, A., & Martín-Martín, R. (2020). robosuite: A modular simulation framework and benchmark for robot learning. CoRR, abs/2009.12293.