Tutorial 2 – CNNs, RNNs, GANs et al.

Spandan Madan

Segment 1

Reflections on Assignment 1

(2 minutes)

Congratulations! Great job!

- Tough assignment, lots covered – python, git, jupyter, colab, pytorch.
- Thank you for your patience!
- Inputs, questions, discussions all highly appreciated!
- Please share your concluding thoughts on slack!

Segment 2

Quick Recap of Tutorial 1

(5 minutes)

Recap

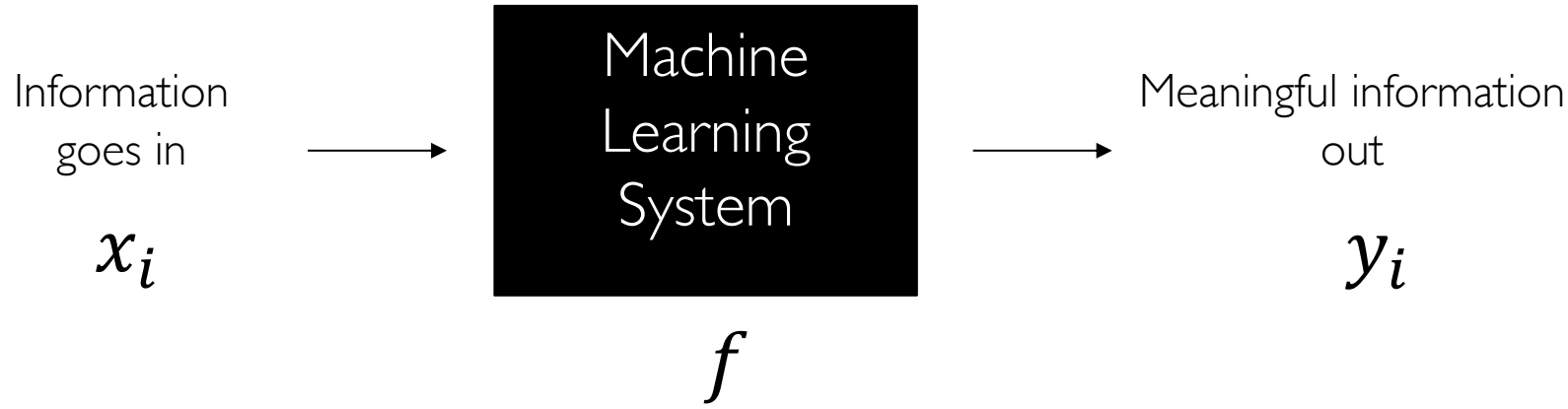
- Git – version control system
- Github = Git + hub

- Jupyter, colab

- Introduction to ML: the equations, the components of these equations and how to code them up.

The toolkit perspective

- Takes in **information**
- **Returns more useful information**.



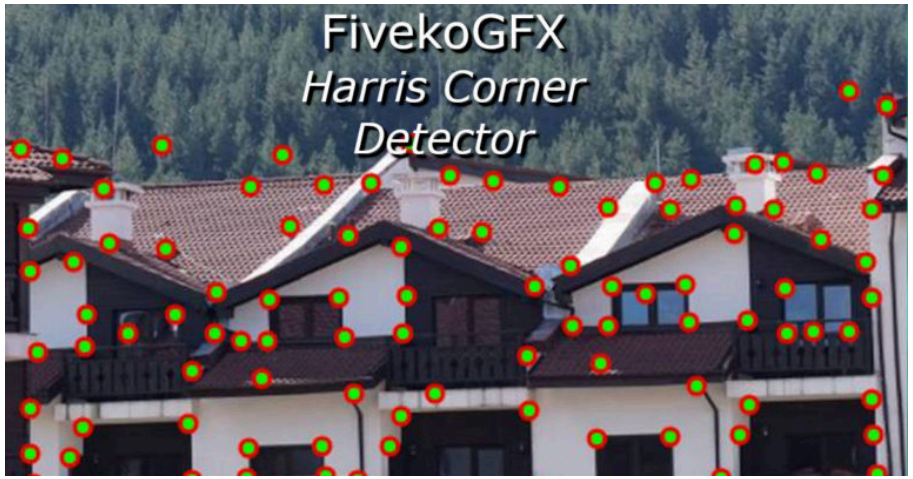
Features – aspects of input useful for the task

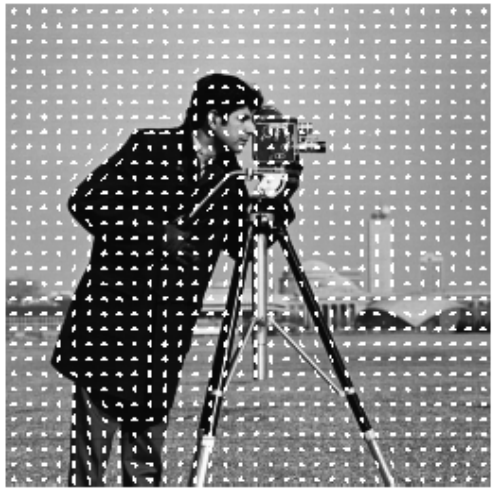
4 legs?

But then, how do you find legs in the image?

Pixabay

Researchers have spent their entire careers finding useful features





Deep Learning: Let's learn the right features

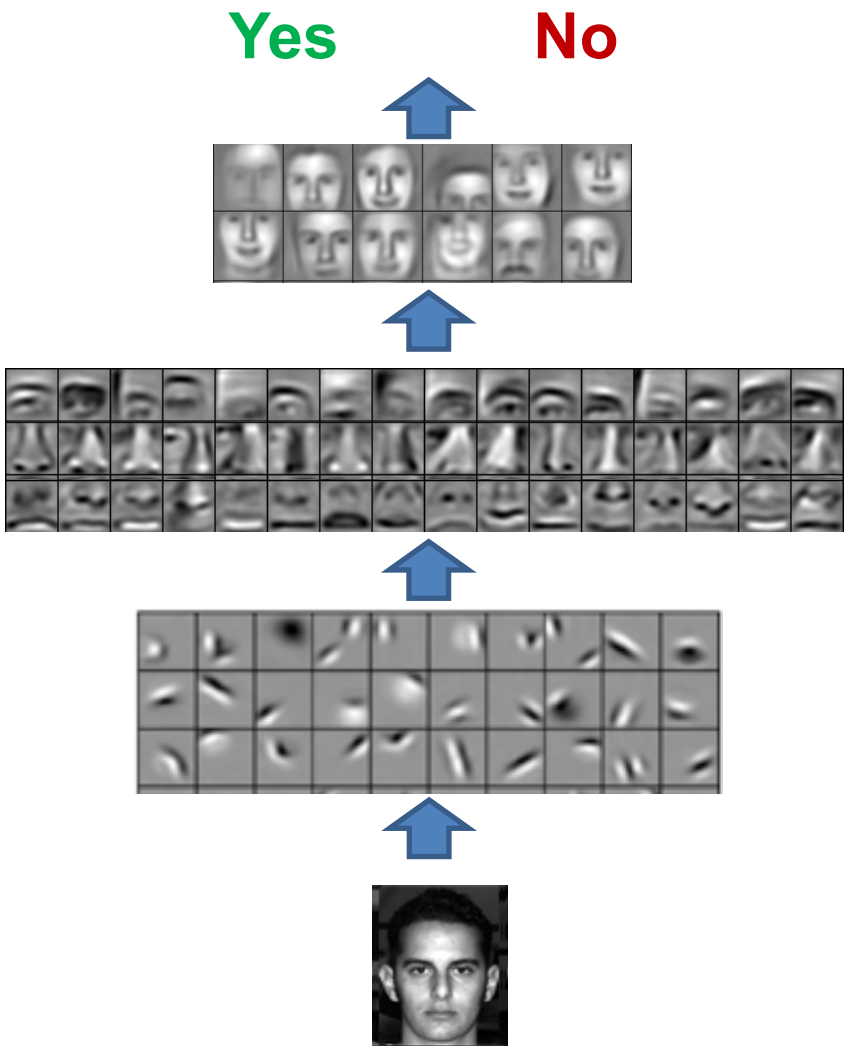
Learned features

Machine Learning System

chair

Deep Learning System

Learned Feature Hierarchy



[Honglak Lee]

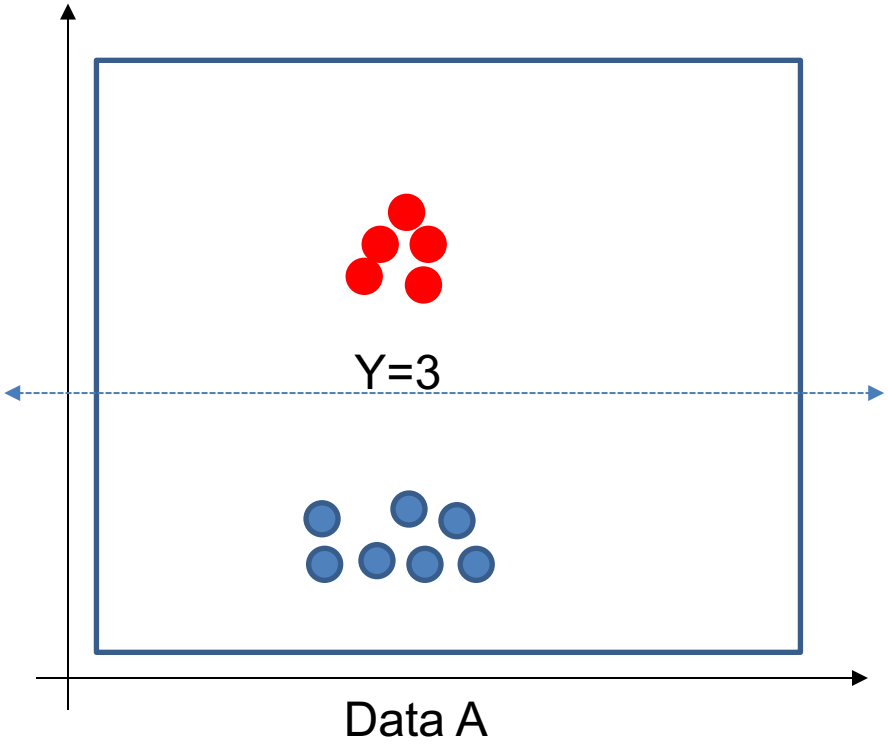
Segment 3

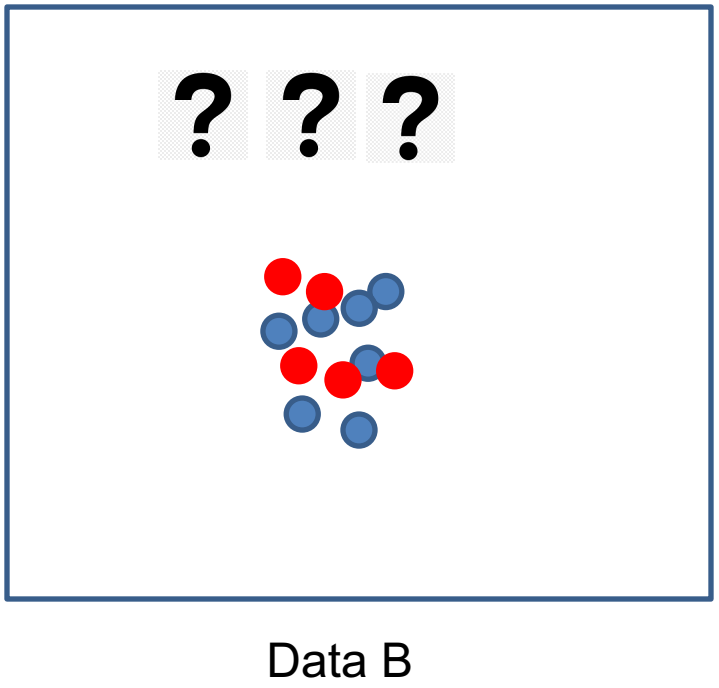
Feed-Forward Networks

(20 minutes)

Intuition for how this works

Which of these data sets would be hard to train a classifier for?



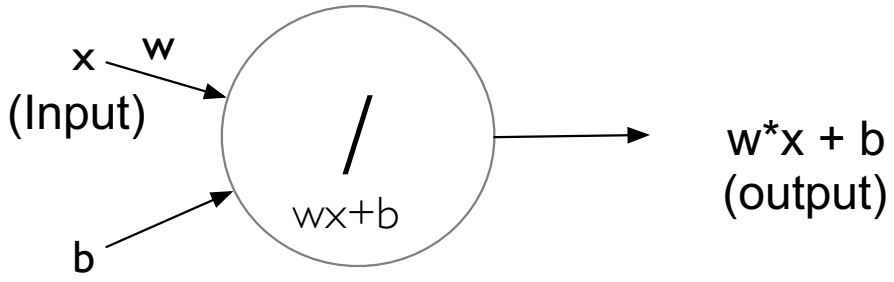


- Is there was a way to transform Data B into Data A so that classification becomes easy?
- That is precisely what deep learning does!

So, how does this really work?

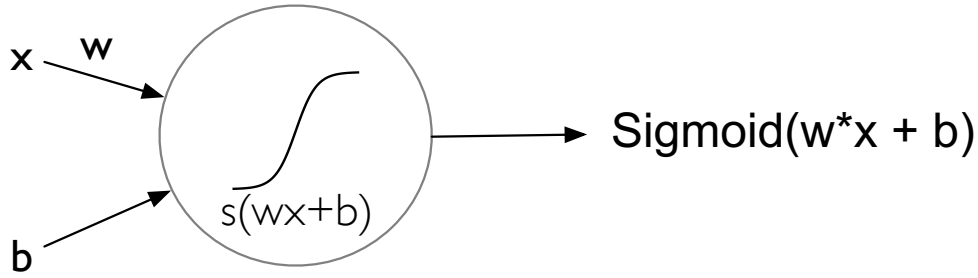
- Anatomy of a neural network:
  - Artificial Neuron
  - Multiple Layers of Neurons
  - Activation Functions
  - Architectures (Covered Later)

Linear neuron



- A neural network is composed of millions of neurons.
- A neuron is defined by:
  - Weight
  - Bias
  - Function inside: Usually a non-linear function like sigmoid, ReLU etc.

Non-linear neuron

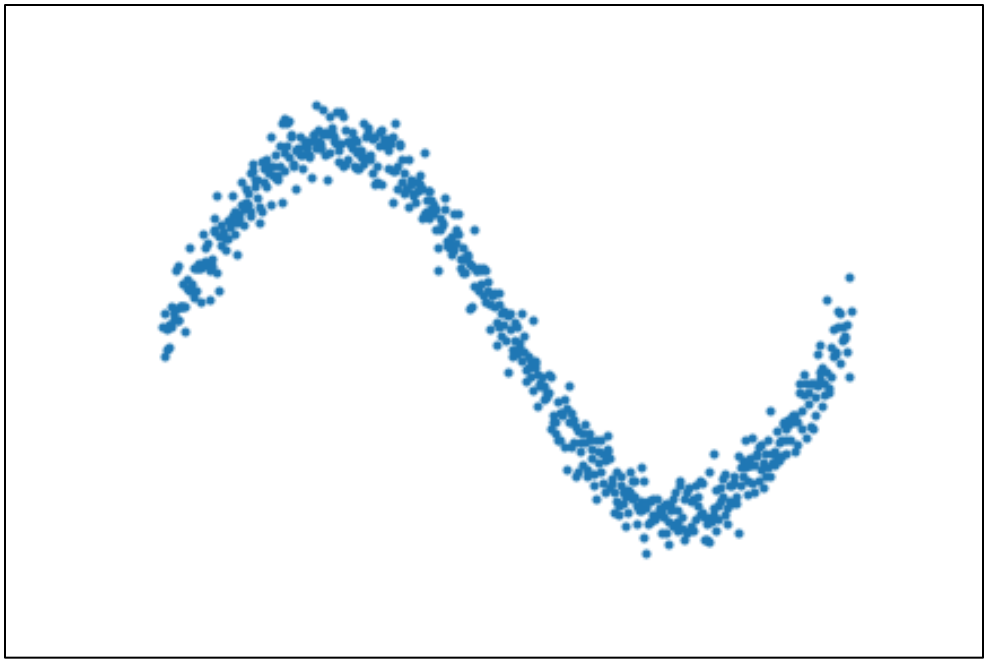


Other common kinds of non-linearity include Tanh, ReLU etc.

But, why do we need non-linear nodes? Let's compare them!

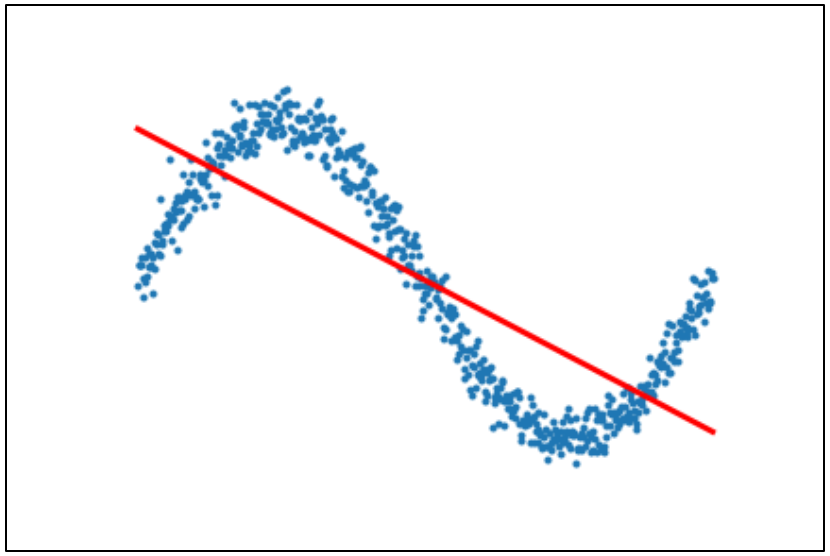
Linear vs Non-linear neuron

- Consider the data below. Let's try to fit the simplest neural net to it: 1 layer with just 1 neuron.

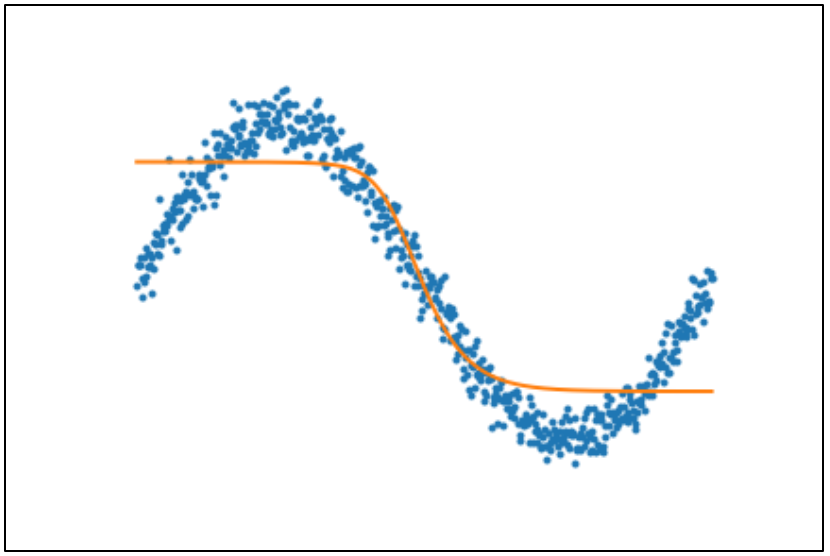


Fitting linear vs non-linear model

As expected, non-linearity helps us fit to more complicate data.



Linear neuron



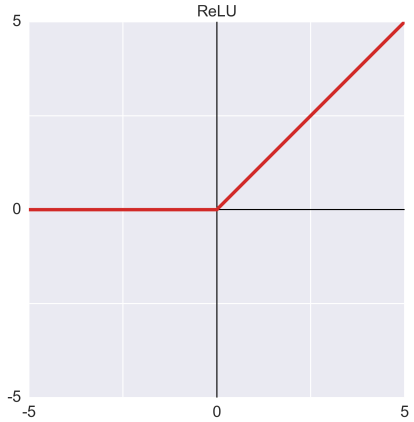
Non-linear neuron

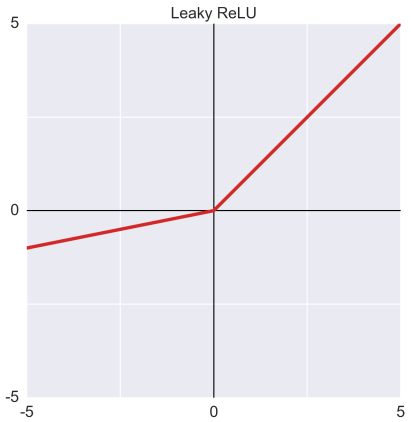
Simple intuition

- f(x) ~ g(x)
- If g(x) is non-linear, f must also be non-linear to approximate it.
- Two ways to do this:-
  - Take lines and combine non-linearly.
  - Take non-linear functions and combine linearly

Source of non-linearity: Activation Functions

- Without non-linearity at units, whole neural network acts like a linear function. So, it reduces to linear regression!
- Different kinds of non-linearities: Sigmoid, ReLU, leaky ReLU etc.

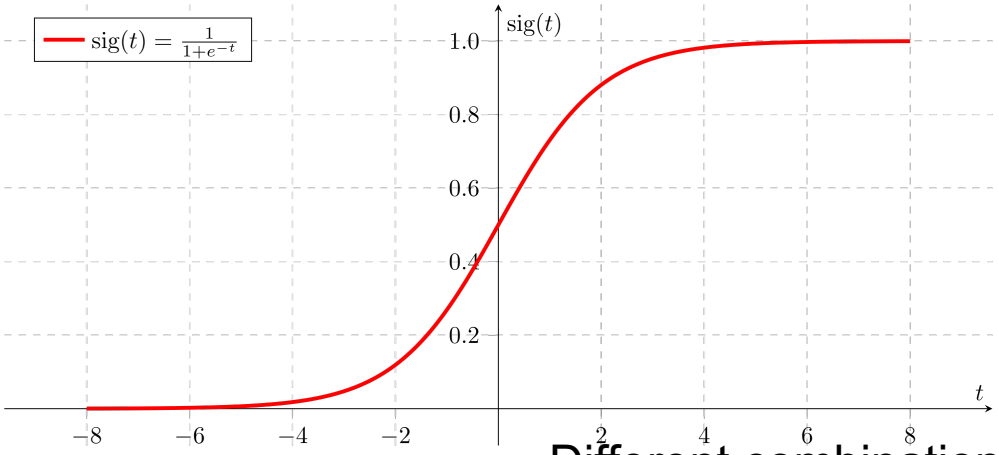


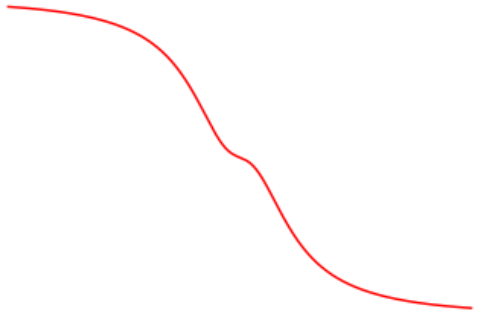


From 1 neuron to 1 layer

- Let's increase the network's complexity: only 1 layer, but multiple neurons.

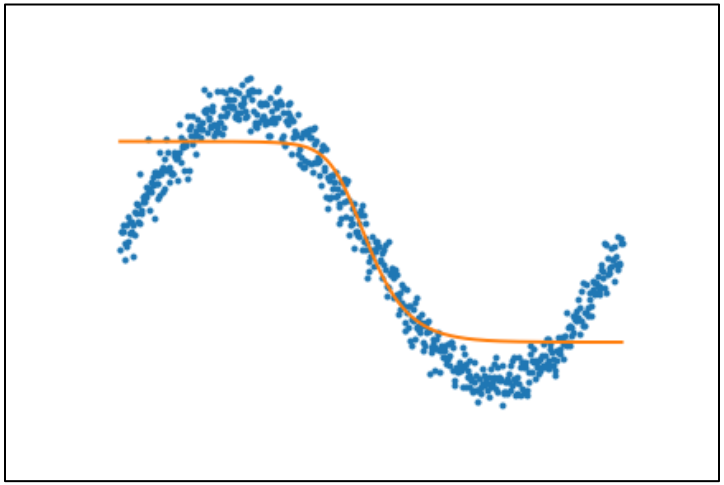
- Intuition:- Combining simpler functions to approximate more complicated ones.

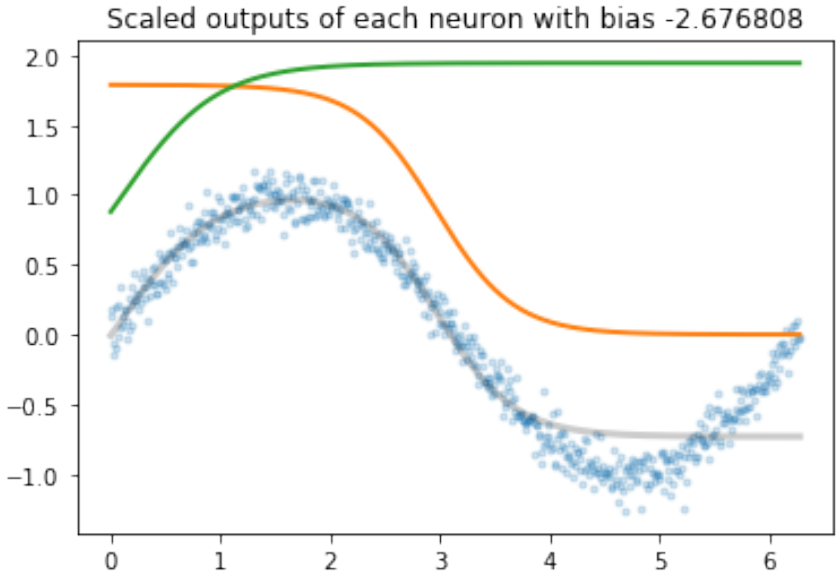


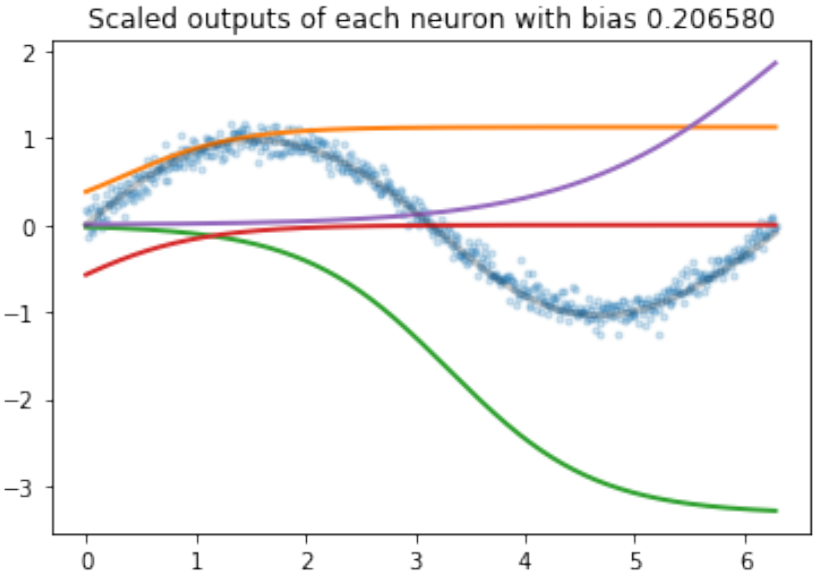


Different combinations of sigmoids

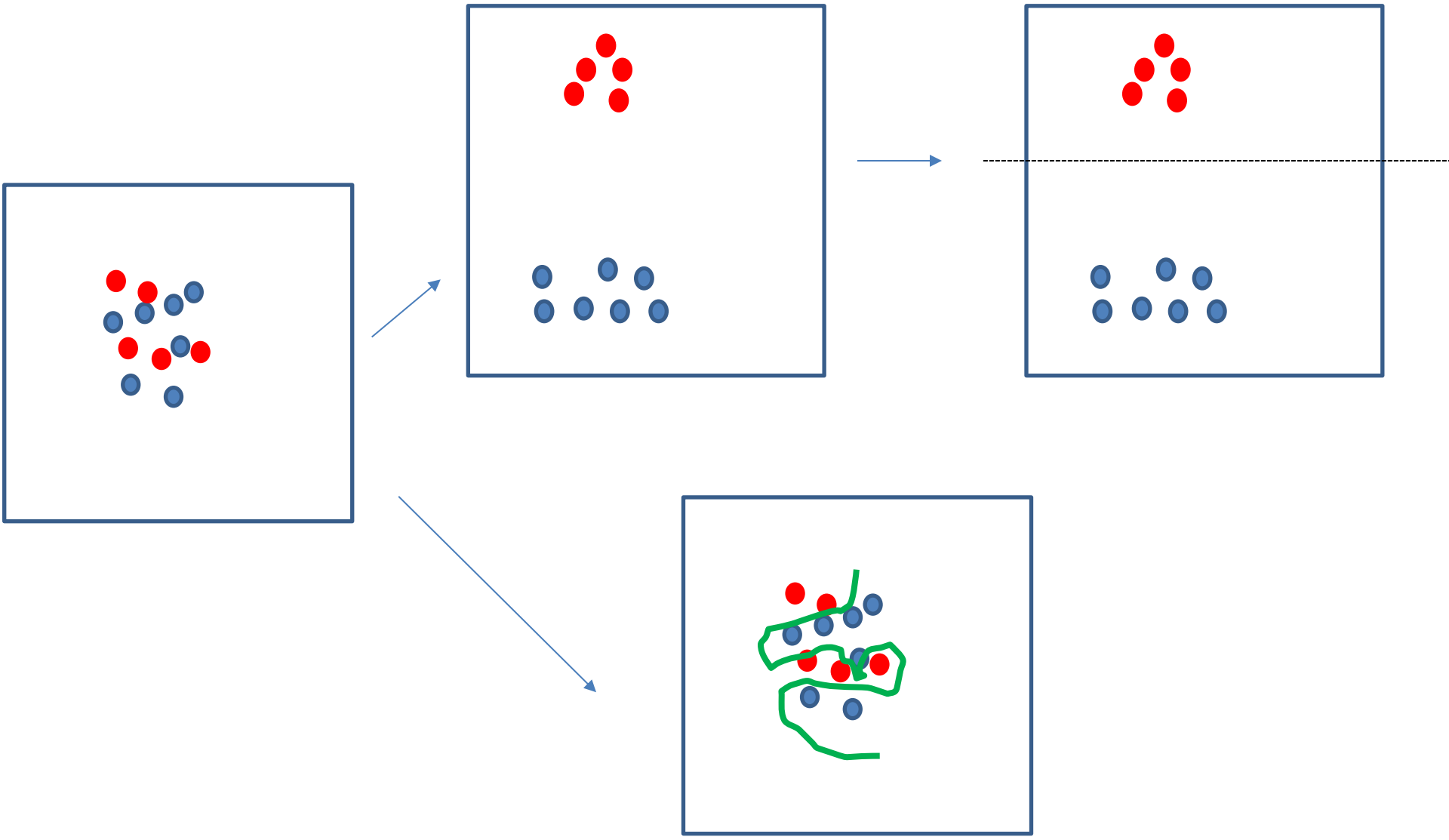
Multiple units, single layer







Two ways to look at it

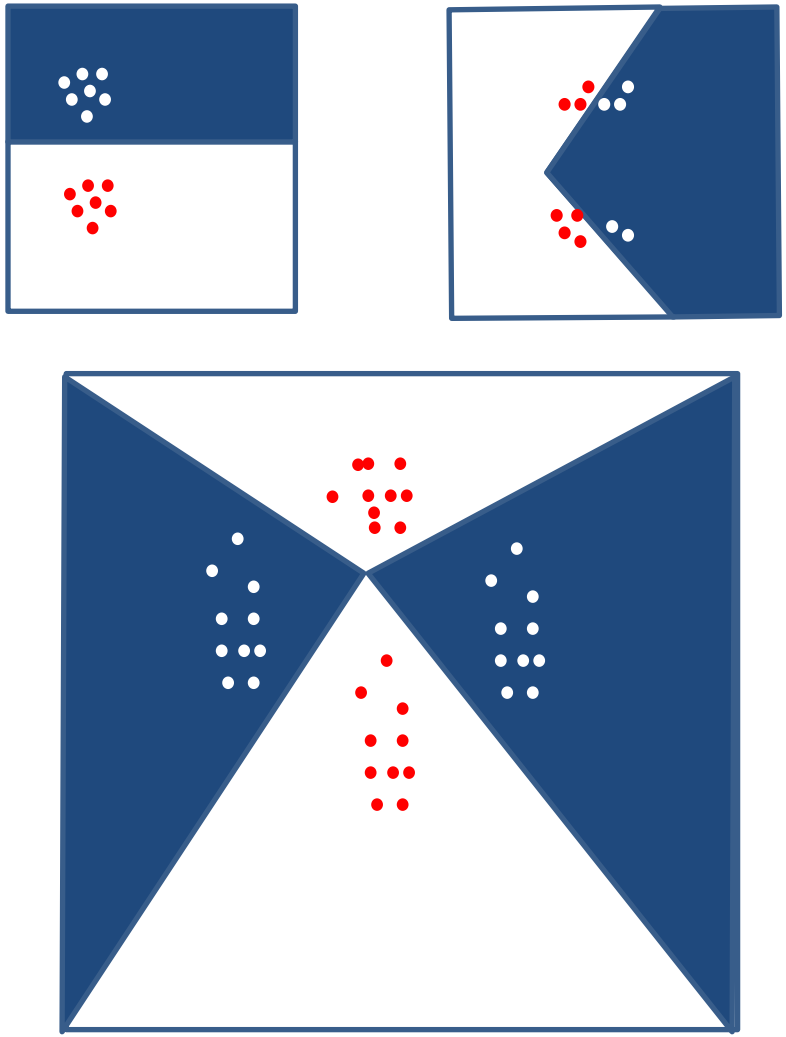


Why have multiple layers?

- What is the utility of multiple layers then?

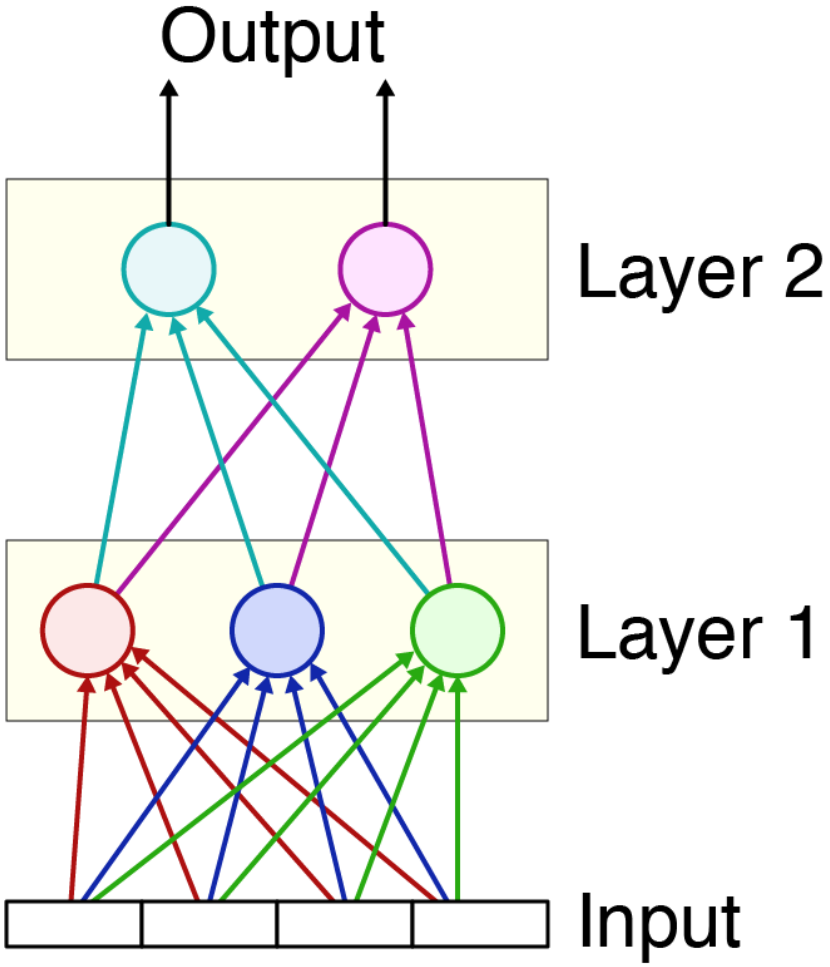
- Intuition: Groups of neurons in first layer can make different patterns. Subsequent layers ***combine*** these patterns to make even more complicated patterns.

Architectures vs classification power



A typical neural network

- 2 layers with artificial neurons
- Outputs of one layer are connected to inputs of next layer
- Four numbers as input, two numbers as output



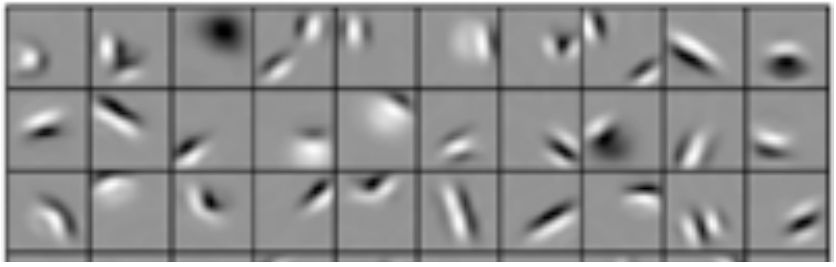
What do different layers do?

Yes **No**

Classification

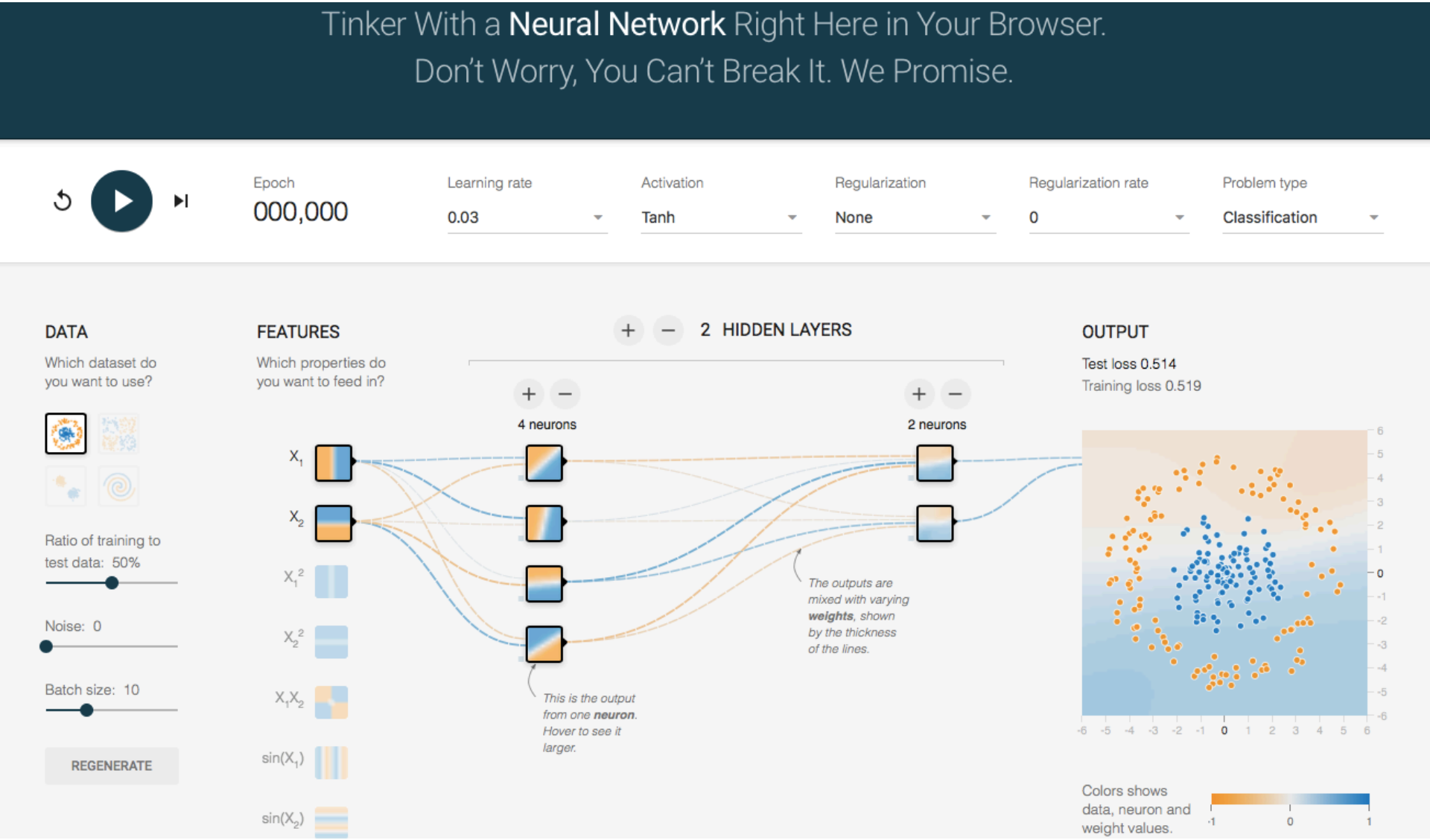
high level features

medium level features



low level features

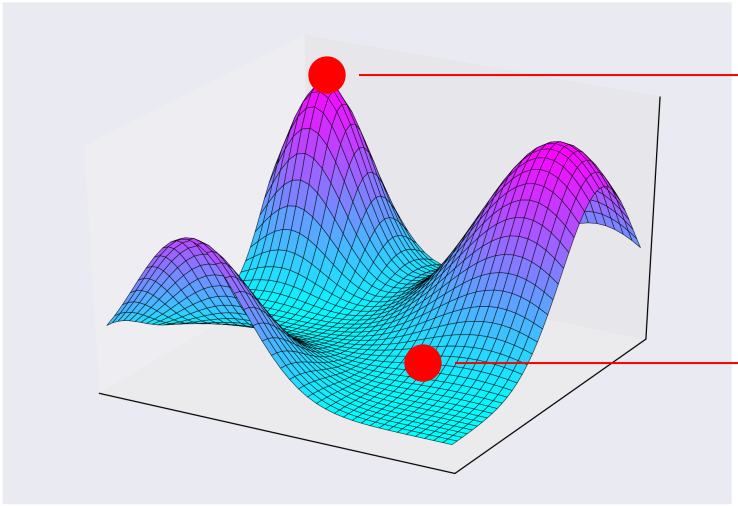
https://playground.tensorflow.org/



Parameters of Neural Network

- Feedforward neural network ~ Chain of computations.
- Output depends on the learned weights, or parameters.
- Training ~ Finding the right set of weights such that the output is desirable. For ex, the label chair, if input is a chair image.
- For the network to train, several factors must be hand designed.

Gradient Descent: Minimizing Error



- Moving along any axis corresponds to how loss will change as one particular weight is changed. We want to move in direction (i.e. update weights) that minimizes loss.
- Gradient: Quantifying change in loss as a particular weight is updated. Thus, one gradient per each weight.
- This is done by calculating the rate of change of loss, and the weight is updated according to this weight.

Stochastic Gradient Descent (SGD)

- Gradient descent looks at all training samples when deciding how to update weights. This is very slow.

- In practice, Stochastic gradient descent is used. At every training iteration, a random sample of training data is used to update weights.

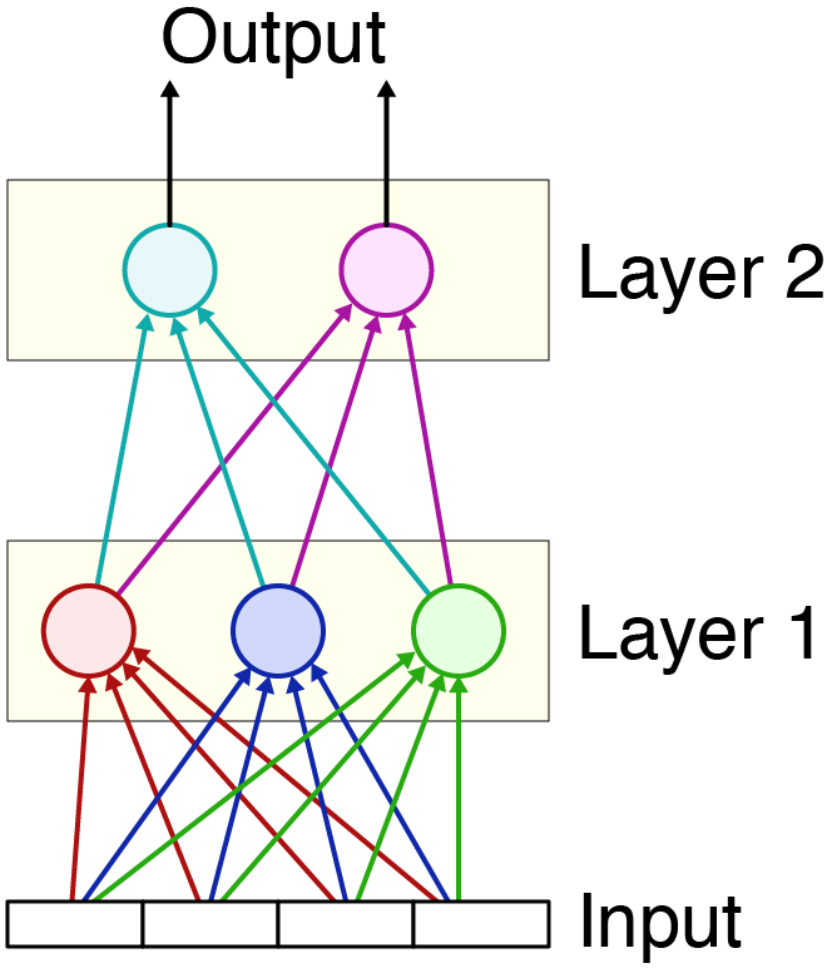
- Two benefits:
    1. Faster
    2. Based on randomness, so it can randomly “jump” out of a tough spot in the terrain.

Backpropagation

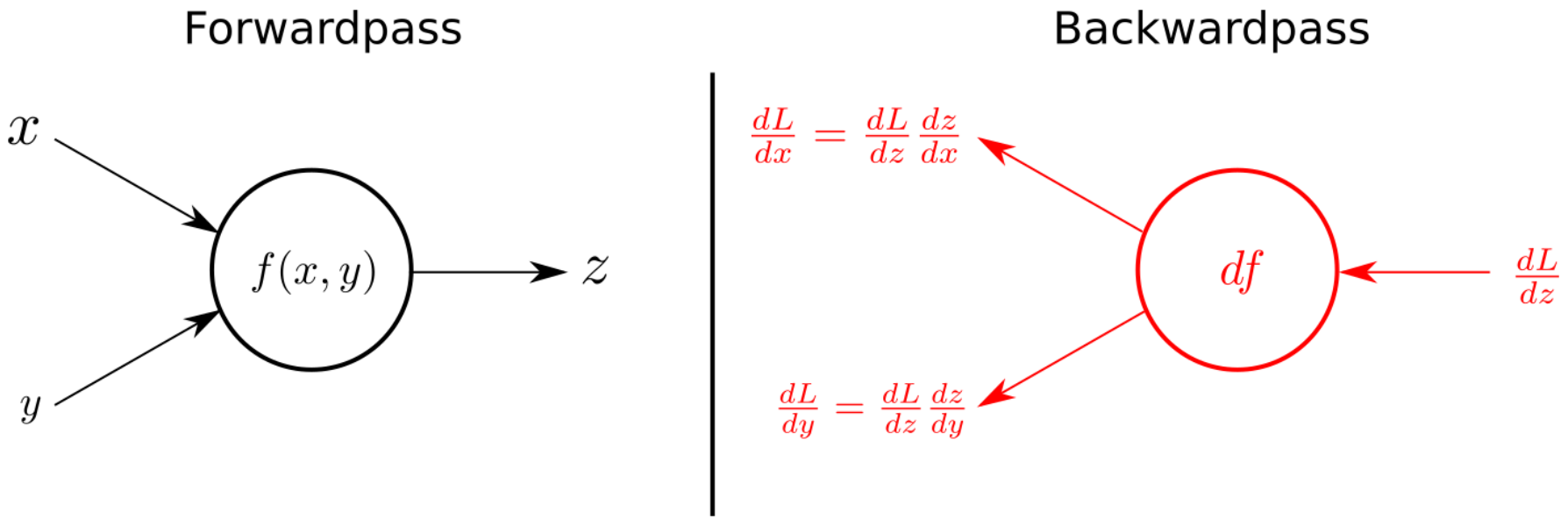
RECAP, Gradient: Quantifying change in loss as a particular weight is updated. Thus, one gradient per each weight.

Problem in calculating gradients: Gradient of weights in layer 1 depends on gradient of the weights of layer 2. Intractable for large network.

Solution: Re-use calculations made for one layer, when calculating gradients for another layer.



Chain Rule


Forwardpass
Backwardpass
x
y
$f(x, y)$
z
$\frac{dL}{dx} = \frac{dL}{dz}\frac{dz}{dx}$
$\frac{dL}{dy} = \frac{dL}{dz}\frac{dz}{dy}$
df
$\frac{dL}{dz}$

Regularization

- One way to prevent overfitting.

- Many, many ways to regularize. If you read it somewhere, it’s a way to prevent overfitting.

Recap

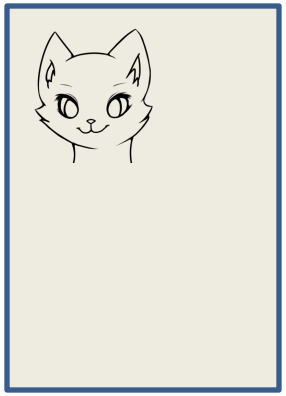
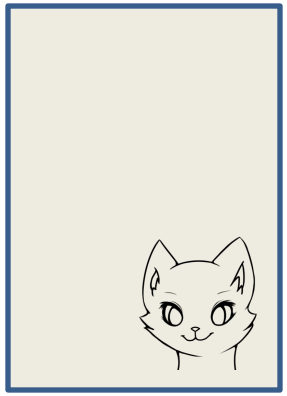
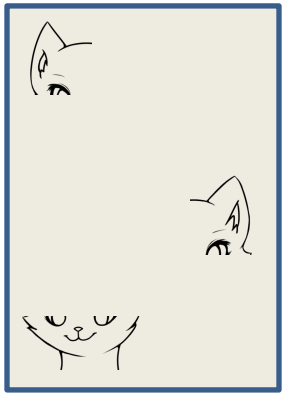
- Training -> Finding the right set of weights such that output of net is favourable.
- Loss -> Quantifying how well a particular set of weights does.
- Thus, training = minimizing loss.
- Loss is minimized by updating weights as dictated by gradient descent.
- To calculate gradient, we use backpropagation which re-uses calculations.
- One common problem is overfitting. It results in good test performance, but model doesn’t generalize to new samples.
- Regularization is one way to prevent overfitting.

Segment 1

Convolutional Neural Networks

(10 minutes)

Images Have Structure – Let's exploit it

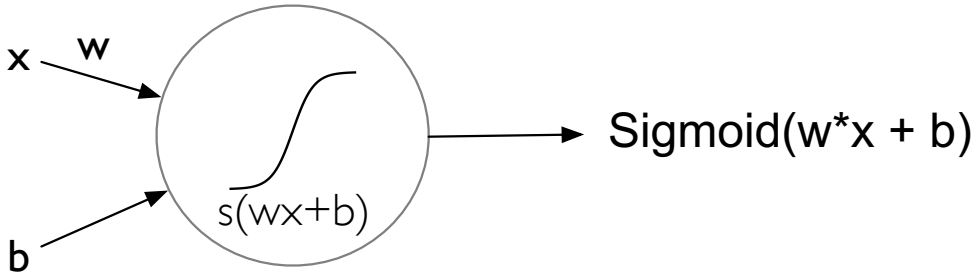




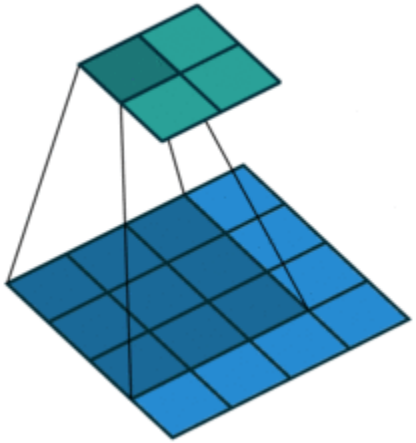
- Position of neighboring pixels should be correct for them to resemble an object.
- We need to look at chunks of local information to extract information from them. Convolutions help us do this.

Motivation for idea of CNN

- Distilling information in chunks of regions.
- Combining chunks into more complex chunks
- Parameter sharing: don’t learn the concept of a cat for every region of image from scratch.

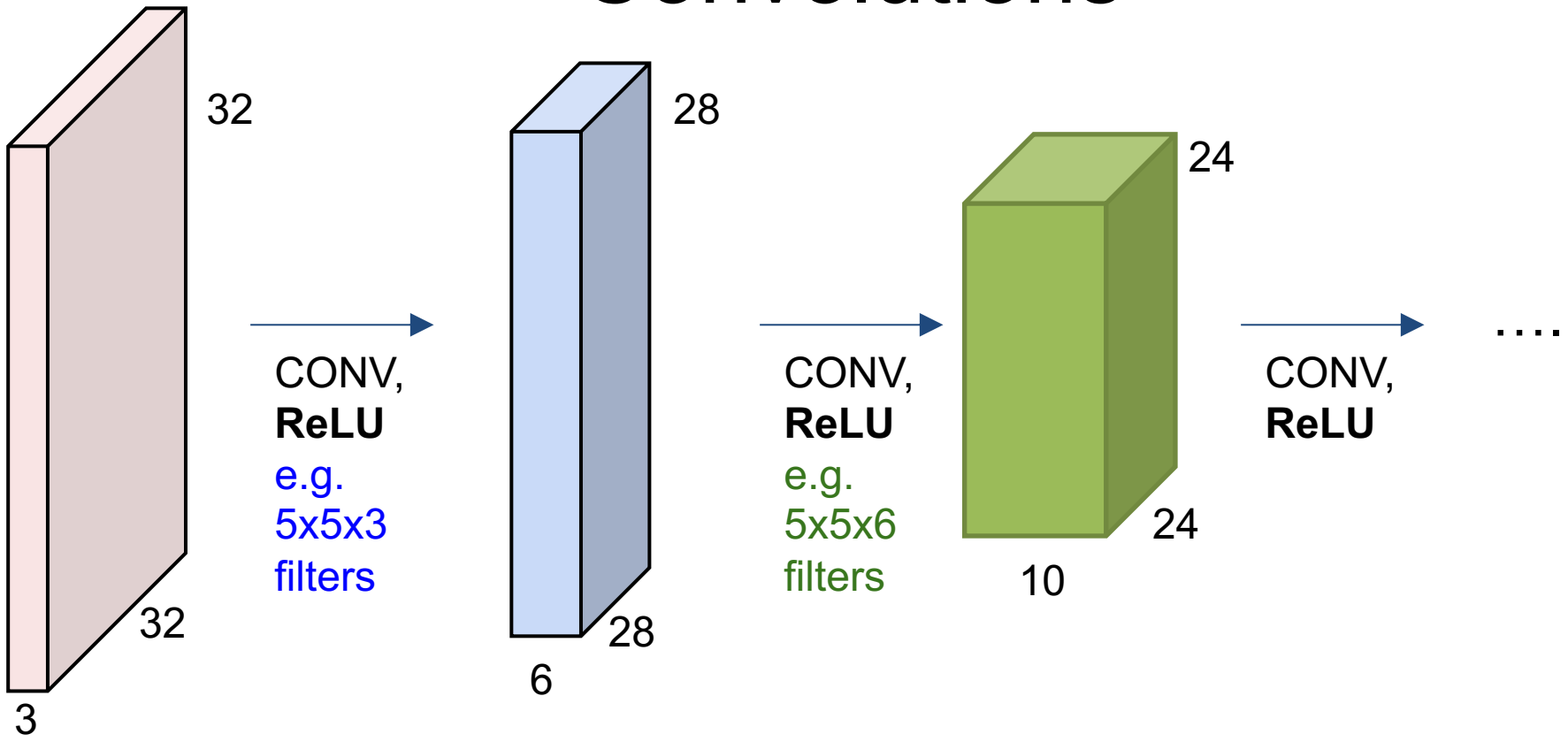
Tweaking neural net to have these properties: Convolution



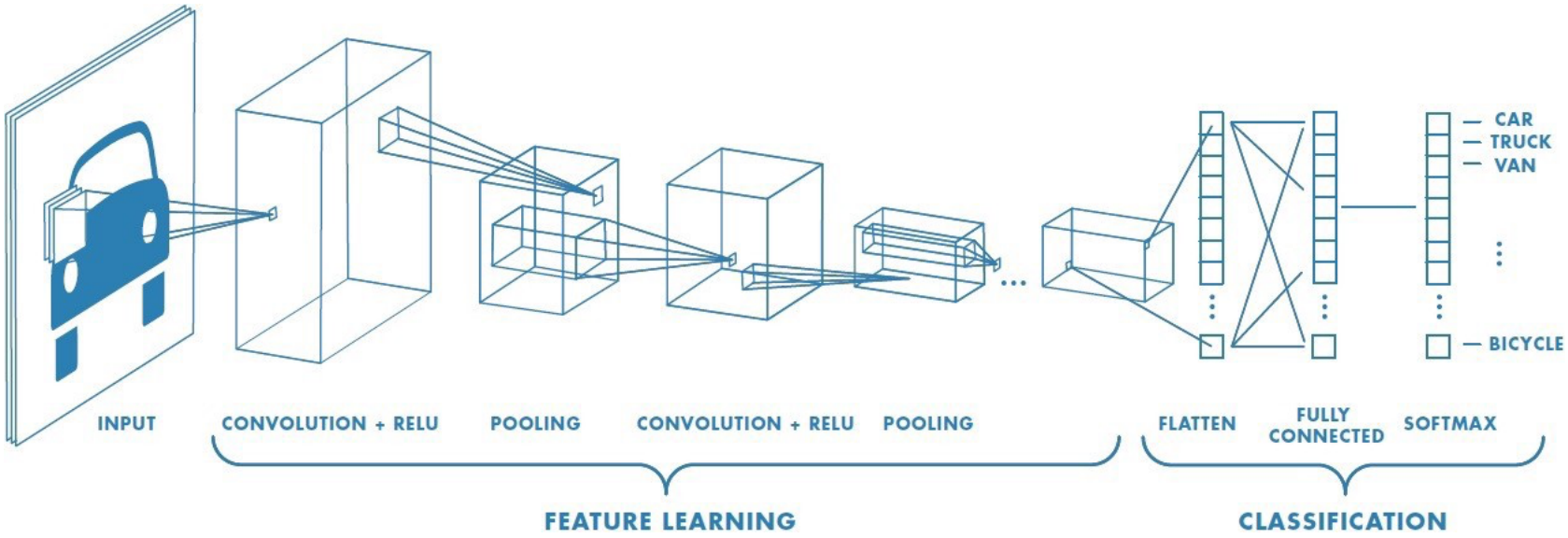


Blue maps are inputs images. The dark blue is a filter (another matrix) which operates on the image and gives the cyan maps as an output.

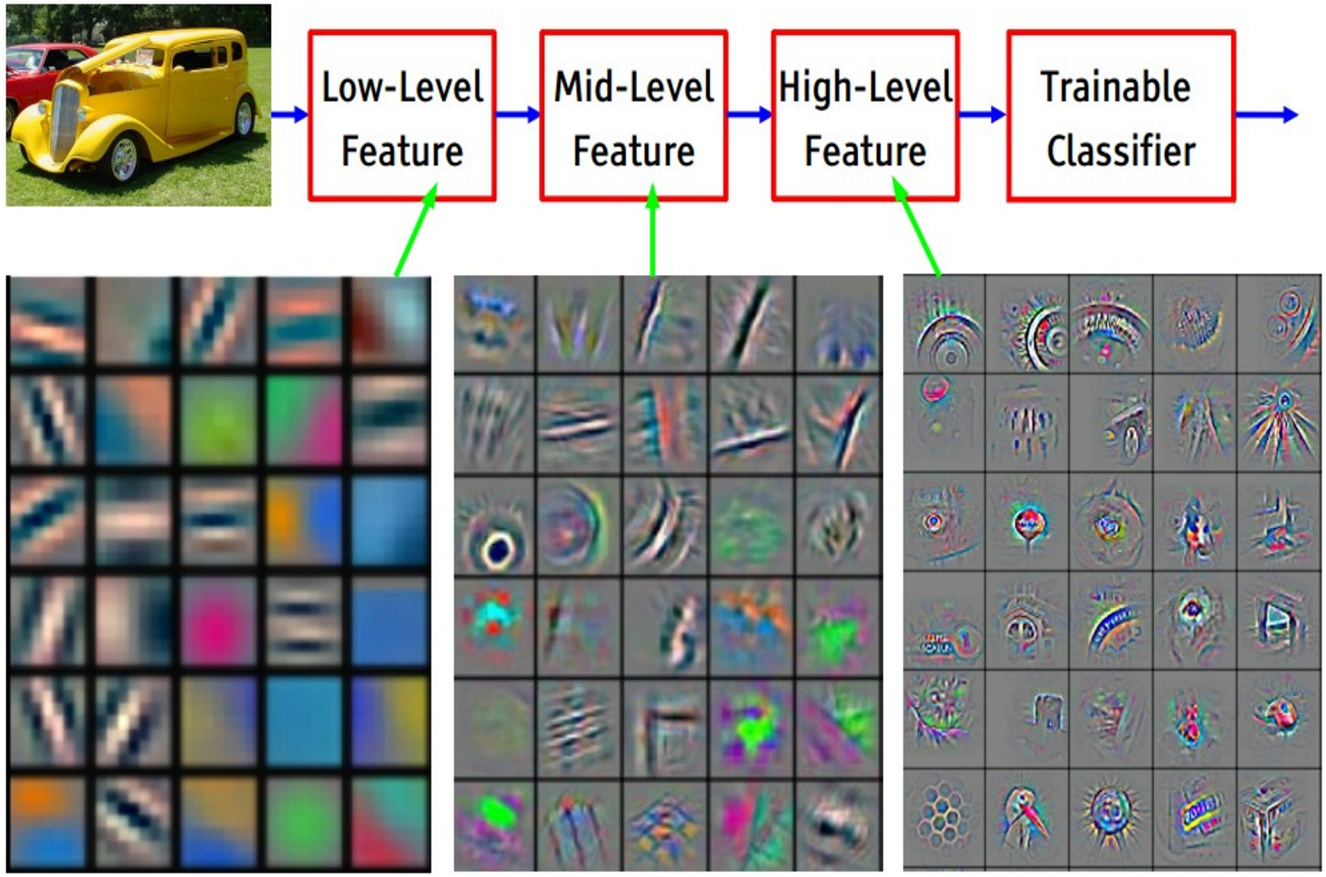
CNNs are composed of layers of Convolutions



Note that the activation function is still present.


CAR
TRUCK
VAN
BICYCLE
INPUT
CONVOLUTION + RELU
POOLING
CONVOLUTION + RELU
POOLING
FLATTEN
FULLY CONNECTED
SOFTMAX
FEATURE LEARNING
CLASSIFICATION

What has the network learned?



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

[Yann LeCun]

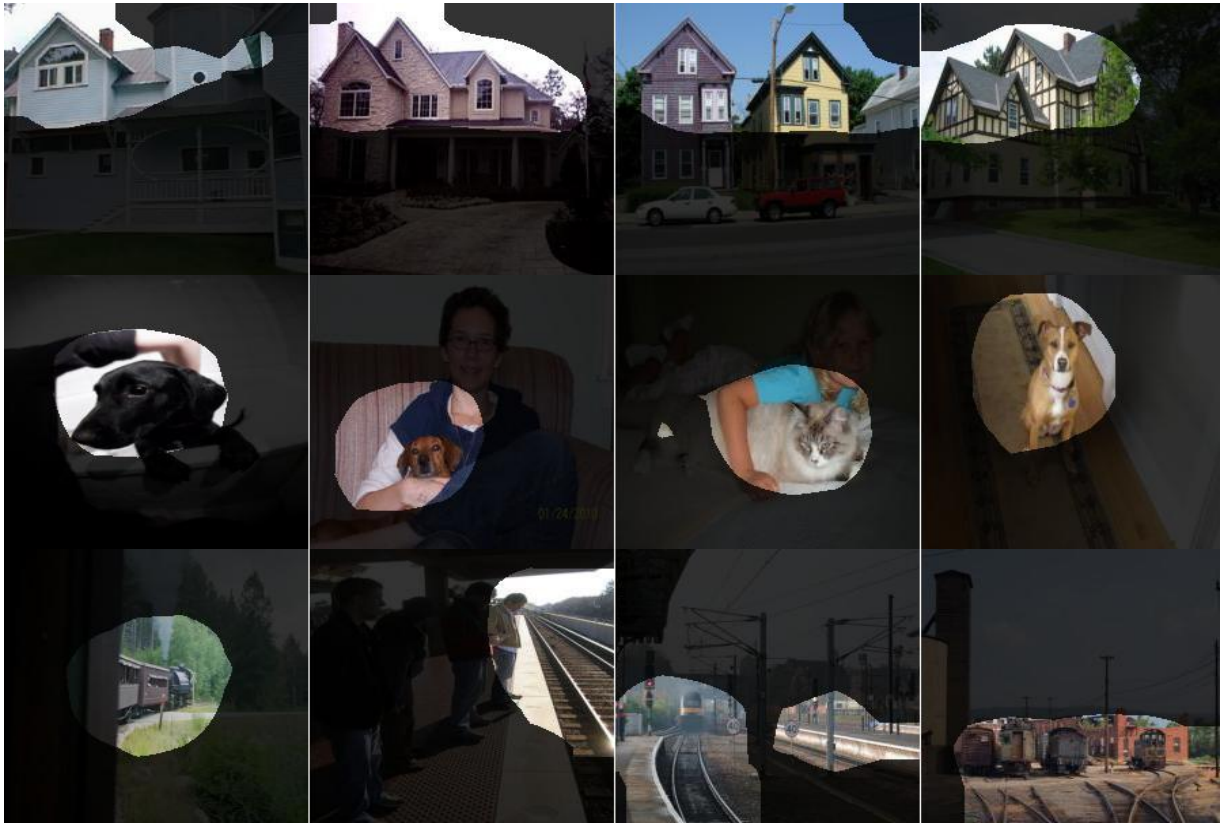
http://cs231n.github.io/

What has the network learned?

House

Dog

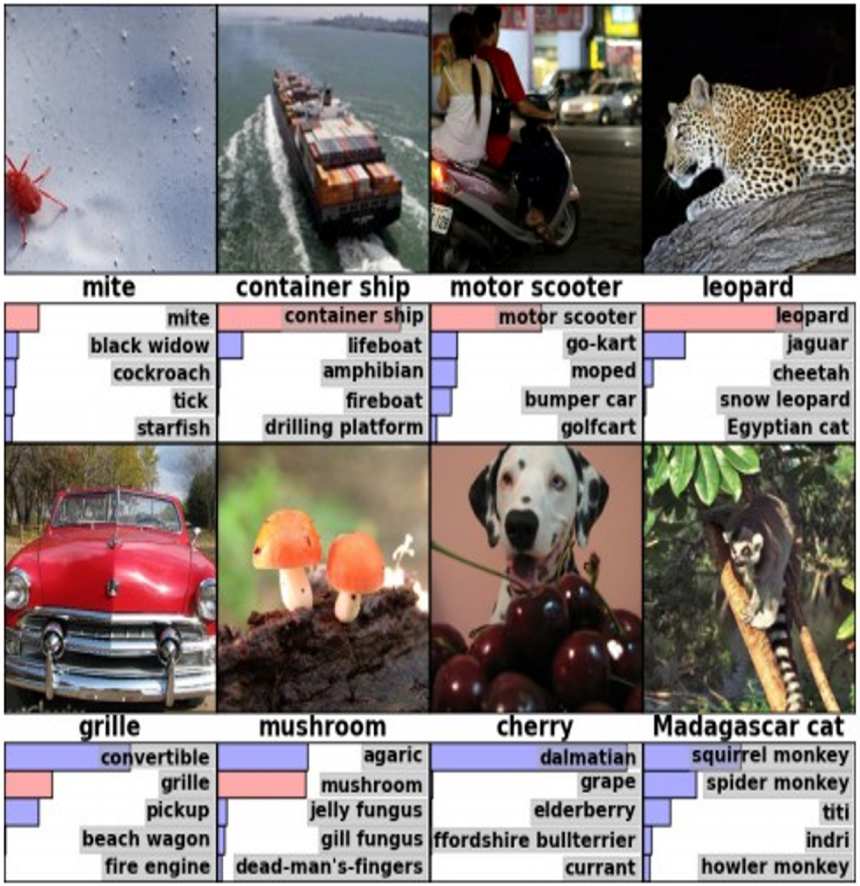
Train



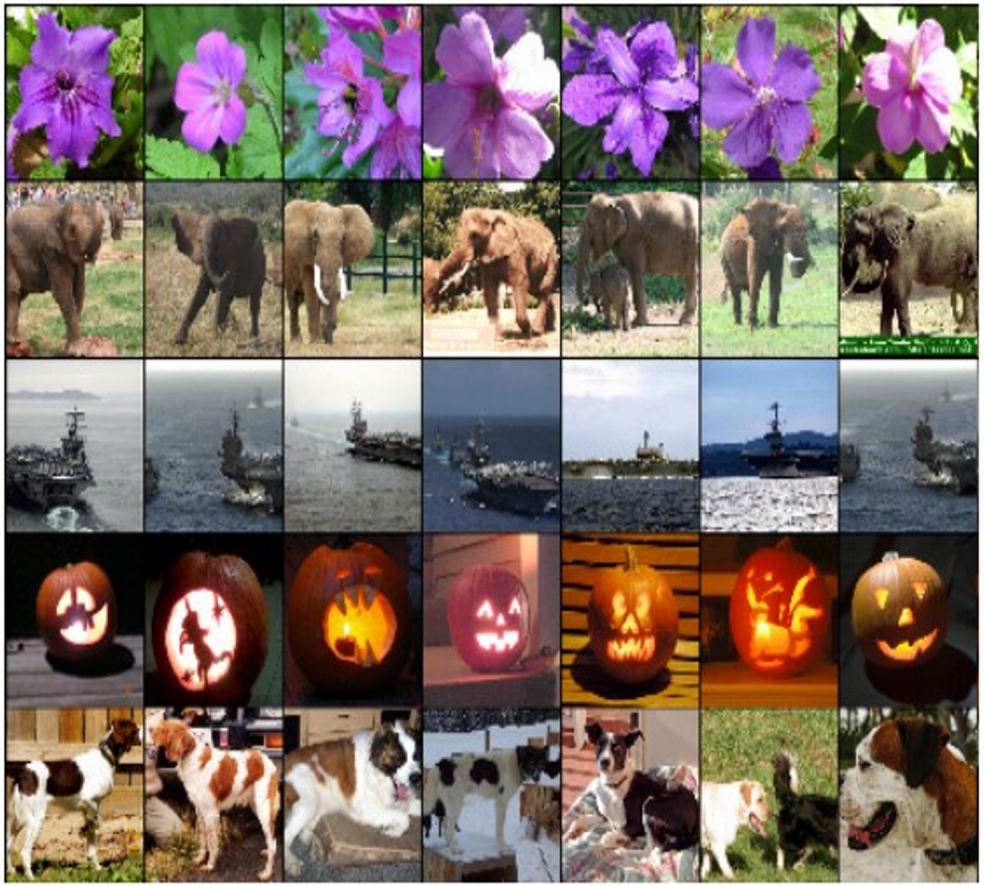
Parts of the image that made network think it's an image of a house/dog/train.

Applications

Classification



Retrieval



[Krizhevsky 2012]

Resources for building further

Video Series to watch (Videos 1-10):

https://www.youtube.com/watch?v=bXJx7y51cl0&list=PLkDaE6sCZn6Gl29AoE31iwdVwSG-KnDzF&index=10

Why?

https://www.youtube.com/watch?v=VOC3huqHrss

If you like the song:

https://www.youtube.com/watch?v=0jgrCKhxE1s

Segment 3

Recurrent Neural Networks

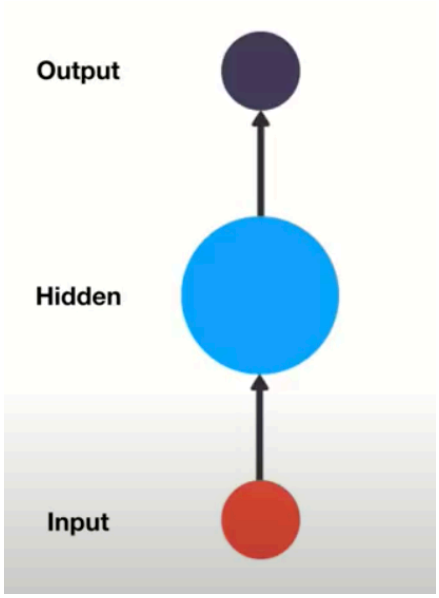
(10 minutes)

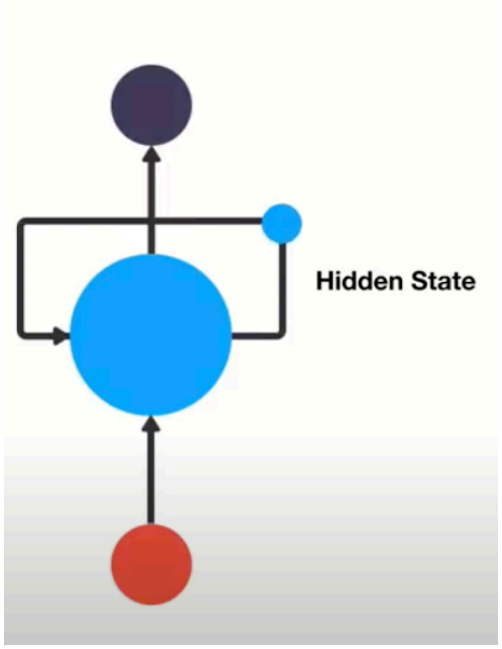
Intuition behind RNNs

- CNNs = patterns in grid of numbers, RNNs = patterns in a sequence
- Examples of sequences:- words in a sentence, characters in a word, sound pressure in air (speech) etc.
- How should we share parameters?
- In CNN- across region of image, in RNN - across regions of _____?

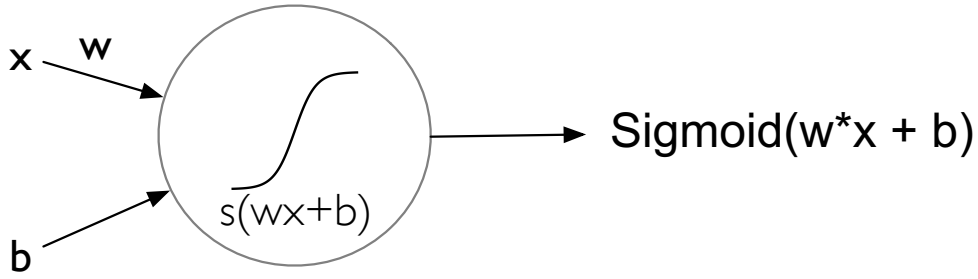
Patterns in different positions of a sequence

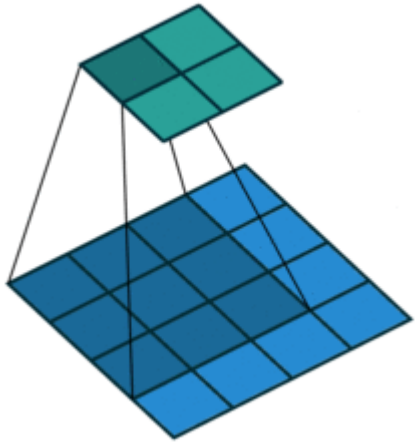
- I am hungry, so I will go to the kitchen.
- I will go to the kitchen, because I am hungry.
- Which word indicates blank would be kitchen?

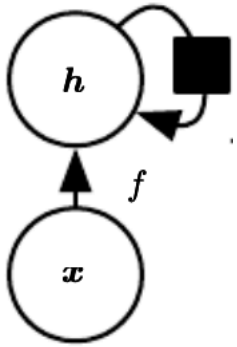




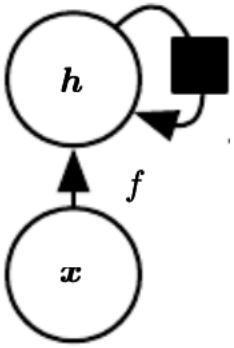
What's the Recurrence in RNNs?



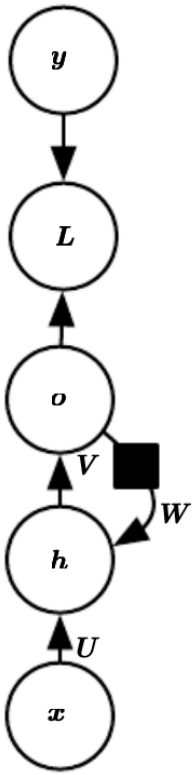




But, what *is* the recurrence?



How about a whole network?



- x, h, o are vectors (so, many units).
- Easy to add another layer – one more recurrent h unit!

Modern RNNs

- Most commonly used RNN types are LSTM (Long Short Term Memory), or GRU (Gated Recurrent Unit).
- We update the single unit to have additional properties.
- In a nutshell, can keep track of patterns across long distances in sequence.

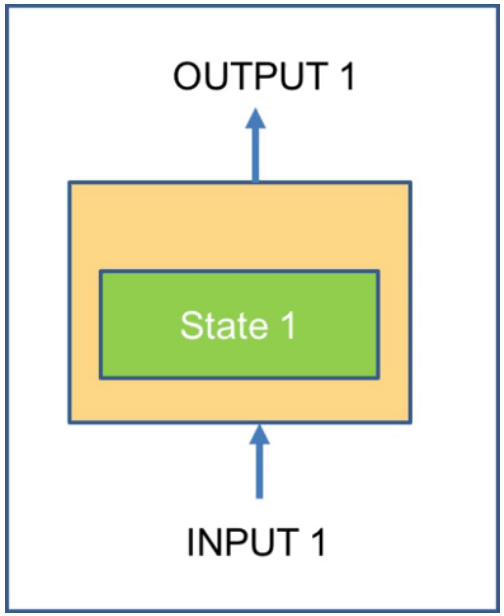
The need for memorizing and forgetting information

- Kilimanjaro is a snow-covered mountain 19,710 feet high, and is said to be the highest mountain in Africa. Close to the western summit there is the dried and frozen carcass of a leopard.

- It is important to remember Kilimanjaro from the past to make sense of word "summit" in the future of the sequence. Networks must have memory to retain such context.

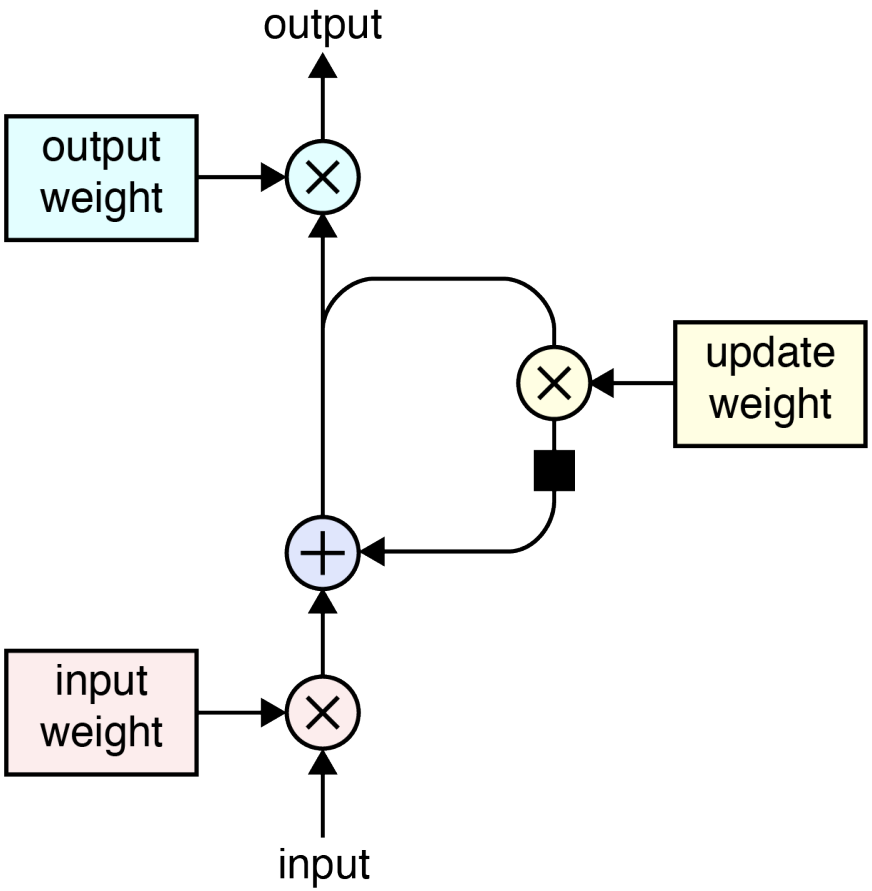
Simple RNN: only 1 RNN cell

- Two Step process:
  - For each new input, compute output using current input and state, which includes info from past inputs as well.
  - Update state to contain info about current input.



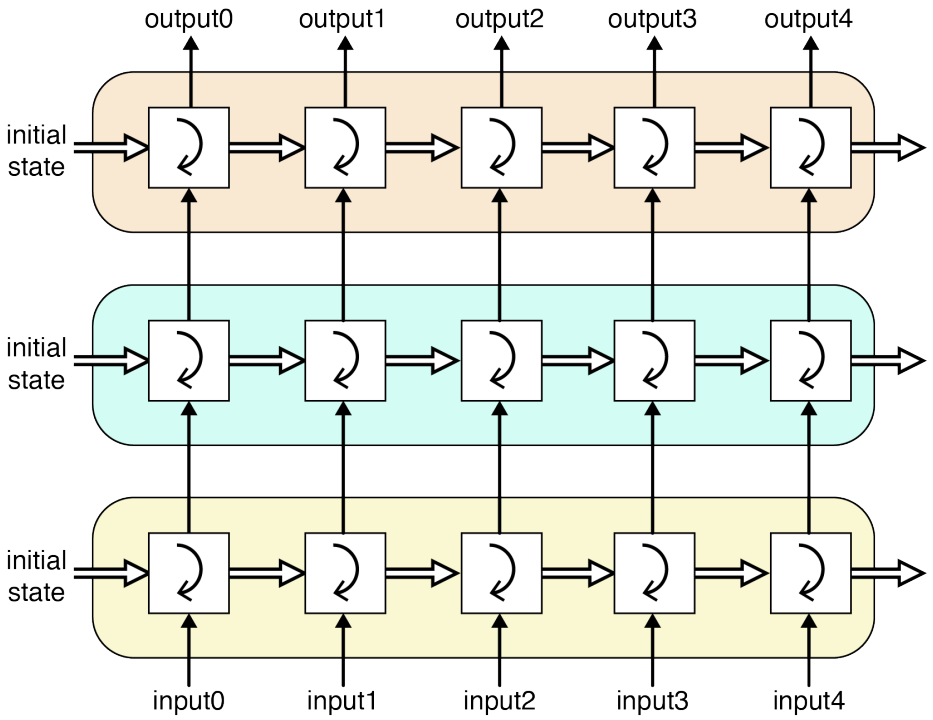
How does the state update?

- Scale values passing through by weights in three places
- Combine delayed state information by adding values
- Output has activation function as well



Deep RNNs

- Multiple RNN stages
- Each stage manages its own state
- Each stage has multiple LSTM units



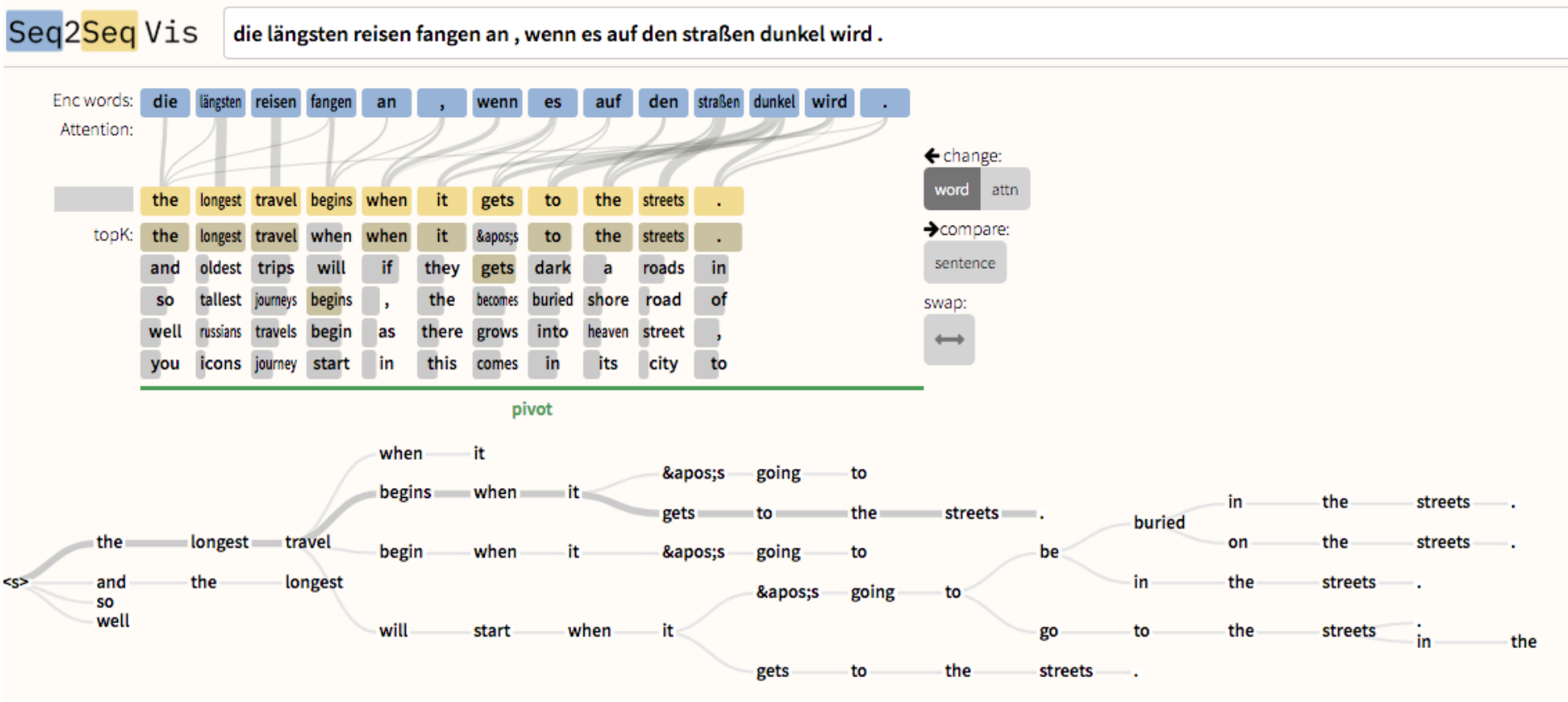
(b)

Further resources

- Illustrated RNN: https://www.youtube.com/watch?v=LHXXI4-IEns

- Illustrated LSTM/GRU: https://www.youtube.com/watch?v=8HyCNIVRbSU

What do RNNs learn?



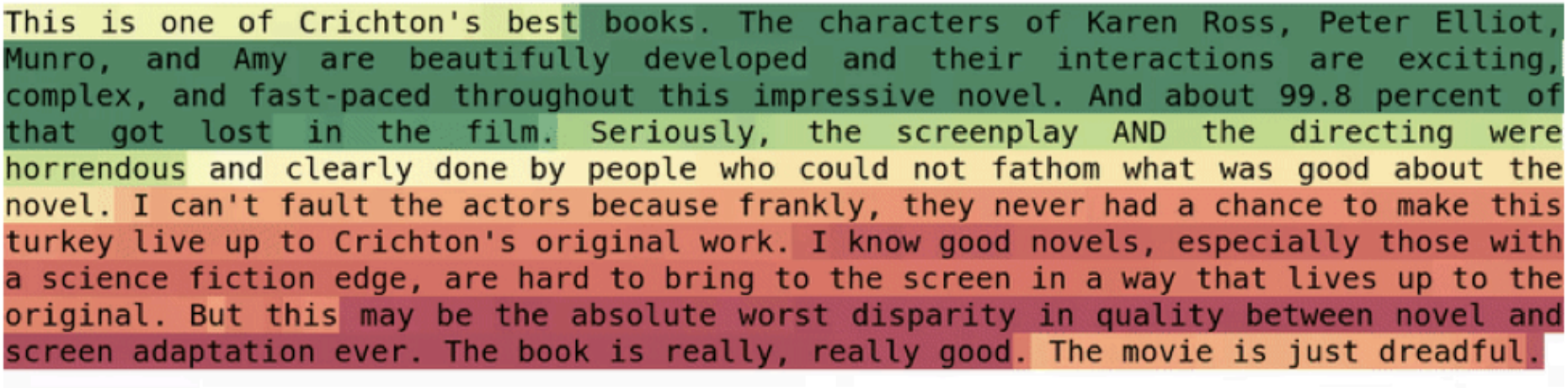
Applications

Translation

페루정부는유색인종과흑인의출입을달가와하지않는레스토랑에대한여러불만을들은뒤수도리마에있는유명레스토랑을닫았고, 추후통보가있을때까지임시휴업을명했다.
NMT: The government has closed a number of restaurants in the capital, Lima, after hearing complaints about a famous restaurant that does not welcome the entrance of people of color and blacks.
V8: The famous restaurant closed in Lima after hearing several complaints to the restaurant, which is not a happy access for people of color, black, Peru has ordered extra holiday until further notice.

Applications

Sentiment Prediction



Segment 4

GANs and other generative models

(10 minutes)

Discriminative vs Generative Models

- Consider two coins, one fair other unfair.
- One gives: H, H, T, T, H, T, H, T, T, H, H
- Other: H, H, H, H, H, H, H, H, T, H, H, H
- Which one is fair?

You just discriminated using a generative model

- Generative model for a coin toss:
- X = {0,1} with 50% probability.
- Similarly, can define generative model over images.
- If you have a model you can – (1) get new samples from it, (2) use model to discriminate.

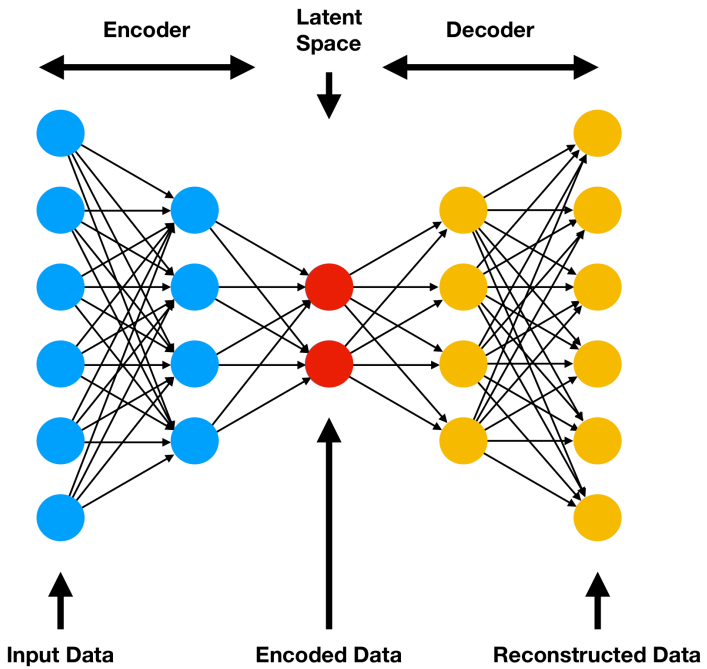
Some faces

Learning distributions can be very useful!

https://thispersondoesnotexist.com

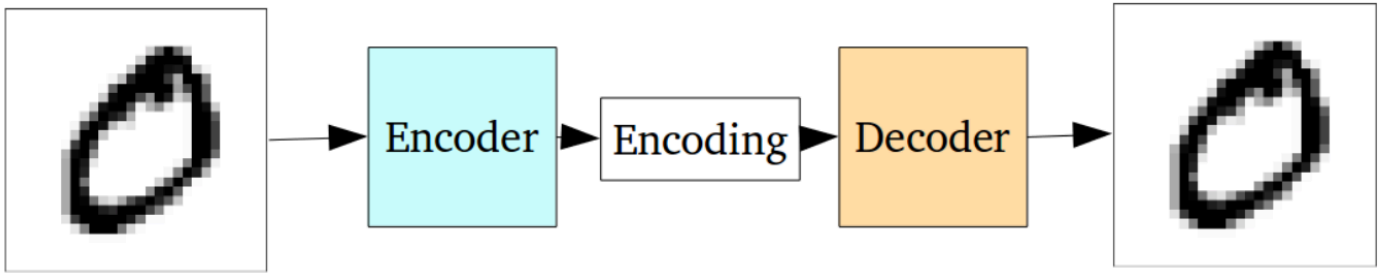
Auto-Encoders

- Dimensionality reduction: Summary of a vector.
- Stenographers of the deep learning world.

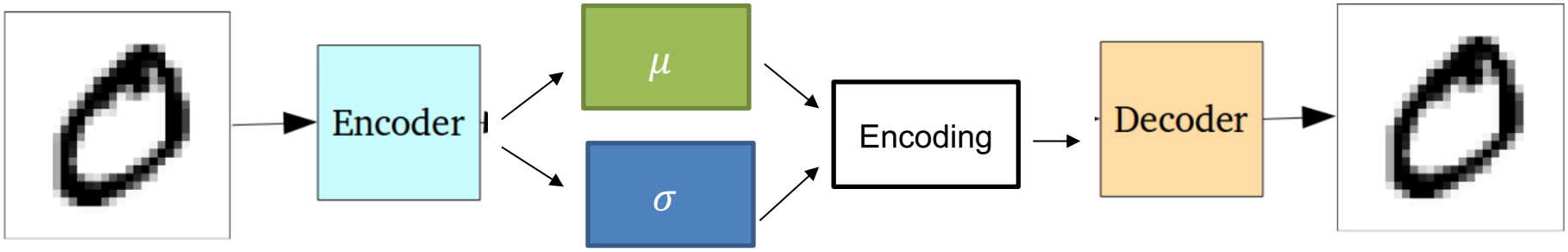


This is how google stores your images on the cloud

Variational Auto-encoders

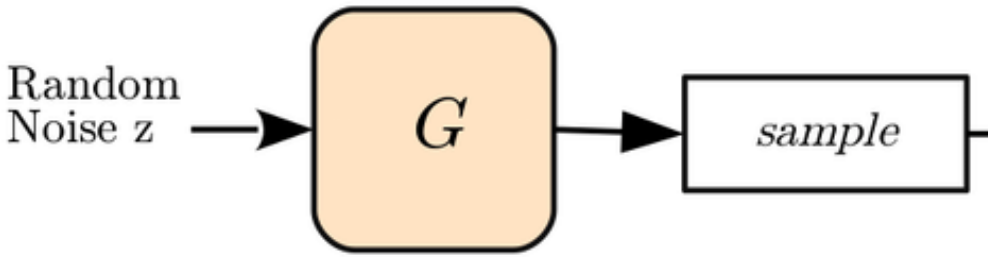


A standard Autoencoder



Generative *Adversarial* Networks

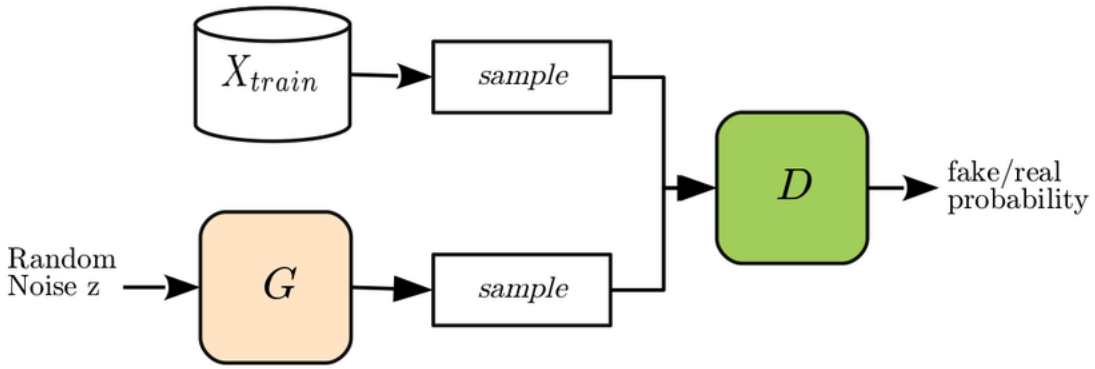
- Learning to sample from complex distribution like images of faces.



- Data available = Images of faces only!
- Trained by a “Discriminator Network”

Discriminator Network

- What can we do with face image?
- Train a classifier!
- Generating new image = fooling classifier?
- Classifier = Discriminator Network

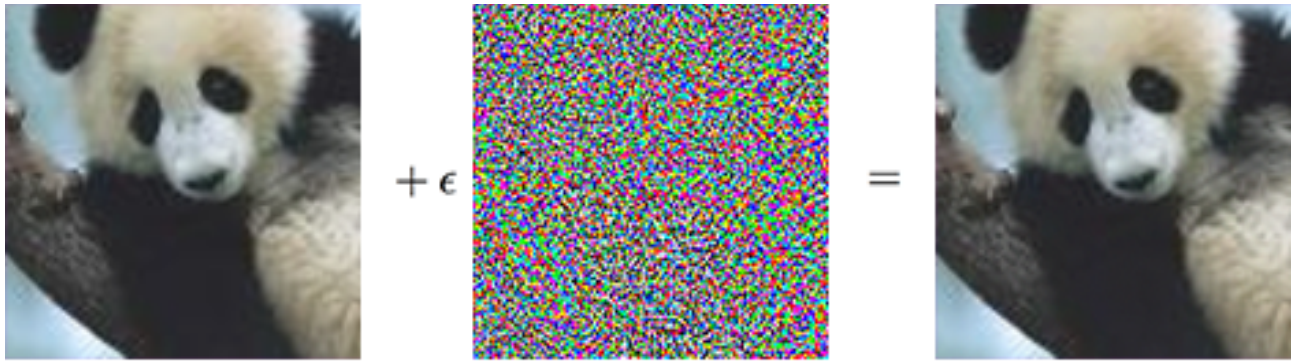


Segment 5

Adversarial Examples and Out of distribution Generalization

(10 minutes)

Adversarial Examples



"panda"
57.7% confidence

"gibbon"
99.3% confidence

Many ways to find adversarial examples

- Usually, loss = output value of correct category node.
- Adv loss = output value of wrong category node.
- Many ways to find them, mostly optimization based.
- Physical adversarial examples exist too!

Out of Distribution generalization

Do networks generalize across…

- Time i.e. video frames?
- Pose?
- Change in color/texture?
- Motion Blur?

Summary - 1

- Reflections + Recap
- Feed forward networks: non-linear neurons, many neurons in one layer, many layers.
- Gradient descent to learn it all.
- Let’s exploit problem structure – parameter sharing.

Summary - 2

- CNNs: same filter across image, capture neighboring features, then combine them across layers.
- RNNs: Same recurrent node across sequence elements, cell state stores information about past words, combines with current word.
- LSTMs/GRUs: For learning long-term relations in sequences

Summary - 3

- Generative models: learning the distribution from which data would have come. For ex: Faces
- Autoencoders = Stenographers
- VAE = Autoencoder beyond just your data.
- GAN = Generator vs discriminator: distribution learned in the process.

Summary - 4

- Adversarial examples: Visual illusions for neural networks.
- Out of distribution generalization: the real end game.

Many moving parts, very little control.

If you get frustrated…

Reminders

- Fill Track A/B sheet!
- Next assignment will be published this week, due Feb 23.
- Project declaration is due 23.
- Colin/Spandan/Gabriel's Office Hours for discussing projects!

Thank you!