





#### See my screen? Hear the bell?

## A/V Test







## Coding Machine Learning





#### Spandan Madan Neuro 140/240





Main Goal: Get you set up and running for coding ML projects.

- •How to set up your code?
- •What tools/languages/frameworks can help make our life easier?
- •How does it connect with the Theory?

## Learning Outcomes









T ### SECTION	I = FIISL, LEUS IMPON
1 ### Secti	on 1 – First, let's imp
⊥ ### Sec	tion I – First, let's : ection 1 – First let's
· · · · · · · · · · · · · · · · · · ·	Section 1 – First, le
. · · · ; 1 #	## Section 1 – First,
	### Section 1 - First
	<pre>fromfuture impor</pre>
	<pre>import torch</pre>
1, 1, 1, 5	<pre>import torch.nn as nn</pre>
1 1 6	<pre>import torch.optim as</pre>
	<pre>from torch.autograd i</pre>
	<pre>import numpy as np</pre>
1, 1; 1; <sup>1; 1</sup> , 1, 9	<pre>import torchvision</pre>
1, 1, 1; <sup>1; 1</sup> , 1; <sup>10</sup>	from torchvision impo
$\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ $\frac{11}{1}$	<pre>import matplotlib.pyp</pre>
11 - 11 + 11 + 11 + 12 = 12	import time
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	import copy
$11^{-1}$ $1^{-1}$ $1^{-1}$ $1^{-1}$ $1^{-1}$	import os
11 - 11 + 11 + 11 + 11 + 15	from PIL import Image
$21^{-1}$ $11^{-1}$ $11^{-1}$ $16^{-1}$	ImageFile.LOAD_TRUNCA
$2^{1}_{2}^{2}^{2}_{2}^{2}^{1}_{2}^{1}_{1}^{1$	<pre>from fine_tuning_conf</pre>
2, 2, 2, 2, 1, 18	
2: 2: 2: 2: 2: 19	## If you want to kee
2, 2, 2, 2, 2, 2, 20	
$2! \frac{2!}{2} \frac{2!}{2} \frac{2!}{2} \frac{2!}{2} \frac{2!}{2!} 21$	if USE_TENSORBOARD:
$21^{21}$ $21^{21}$ $21^{21}$ $21^{21}$ $21^{21}$ $21^{21}$ $21^{21}$ $22^{21}$	from pycrayon imp
$2^{2} 2^{1} 2^{2} 2^{2} 2^{2} 2^{2} 2^{2}$	<pre>cc = CrayonClient</pre>
21 <sup>2</sup> 21 <sup>2</sup> 21 <sup>2</sup> 21 <sup>2</sup>	try:
$21^{21}$ $21^{21}$ $21^{21}$ $21^{21}$ $21^{21}$	cc.remove_exp
31 <sup>2</sup> 21 <sup>2</sup> 2 <sup>2</sup> 2 <sup>3</sup> 2 <sup>6</sup>	except
3: <sup>3</sup> , 3: <sup>2: 2: 2</sup>	pass
3; 3; <sup>3</sup> ; 3; <sup>2; 2;</sup> 3; 3; <sup>3</sup> ; <sup>2</sup> ; <sup>2</sup> ; 28	foo = cc.create e
3: 3, 3, 3, 3, 2 3: 3: 3, 3, 29	
3, 5, 5, 3, 3, 5, 5, 30	
3, 3; 3; 3, 3	## If you want to use
3, 3, 3, 3, 32	
3, 3,	use $apu = GPU MODE$
3, 33	if use apu:
37	



## Learning Outcomes





moort everything we will be needir

print\_function, division

optim ort Variable

datasets, models, transforms lot <mark>as</mark> plt

D\_IMAGES = True \_file import \*

track of your network on tensorboard, set USE

CrayonClient ostname=TENSORBOARD\_SERVER)

iment(EXP\_NAME)

periment(EXP\_NAME)

he GPU, set GPU\_MODE TO 1 in config file

#### Files in that project

import torch from torchvision import datasets

def train\_model():

def evaluate\_model():

#### **Code in these files**





### Section 1 – First. let's import everything we will be needing ### Section 1 – First, let's import everything we will be need from \_\_future\_\_ import print\_function, division import torch nport torch.nn as nn import torch.optim as optim from torch.autograd import Variable oort torchvisior from torchvision import datasets, models, transforms import matplotlib.pyplot as plt import time import copy import os from PIL import ImageFile ImageFile.LOAD\_TRUNCATED\_IMAGES = True from fine\_tuning\_config\_file import \* ## If you want to keep a track of your network on tensorboard, set USE if USE TENSORBOARD from pycrayon import CrayonClient cc = CrayonClient(hostname=TENSORBOARD\_SERVER) cc.remove\_experiment(EXP\_NAME) except: pass foo = cc.create\_experiment(EXP\_NAME) ## If you want to use the GPU, set GPU\_MODE TO 1 in config file

 $use_gpu = GPU_MODE$ 

How is this box organized? Where is it stored? How do I track changes? How do I share it with someone else? How would these be opened?

What kind of files would I have? Which ones need to be shared and how?

## Learning Outcomes





Section 1 – First. let's import everything we will be needin ### Section 1 – First. let's import everything we will be needing



from torchvision import datasets

def train\_model():

def evaluate\_model():

#### **Code in these files**

#### Files in that project

What language/frameworks should I use?









- Where is it stored?
- How do you track changes?
- How do you share it with someone else?



# Part 1: The Box of Code



### How is your project organized?





- Git: A system for tracking versions of files (Version Control)
- GitHub: Store your code + it tracks changes using git

## GitHub = Git + Hub







## Why do we need Git?











# Why do we need Git?





- Managing versions is important.
- Especially when multiple people are working on the same codebase.
- Combined with GitHub Great way to store/manage/share your projects.





Workspace	
Content	
Directory1/ File1 File2	
Directory2/ File3 Directory3/ File4	









## How git works



Local







Remote



<u>Group 1: Getting stuff from github</u> git clone - Copy from GitHub to local. git fetch - get changes from remote git merge - merge changes into local git pull = git fetch + git merge

**Group 2: Reflecting your local changes on github** git add - add a changed file to staging area git commit - commit the change i.e. create snapshot git push - upload new snapshot to github

**Group 3: Branches etc git checkout -** switch or create new branch (Assignment 2)

https://github.com/Spandan-Madan/Harvard\_BAI

## Few useful commands







## More advanced git

- How to resolve merge conflicts.
- Pushing too large a file undoing added files.
- Undoing a commit i.e. moving to previous commit.







## Summary Part 1

- Git = Management system for your files





GitHub = a place where code is stored and managed using git snapshots





1 ### Section	1 – First, let's impo
1 ### Secti	on 1 – First, let's im
1 <b>###</b> Sec	tion 1 - First, let's
· · · · · · · · · · · · · · · · · · ·	ection 1 - First, let E Section 1 - First le
· · · · · 1 #	### Section 1 - First,
	### Section 1 - First
	fromfuture impo
	import torch
	<pre>import torch.nn as ni</pre>
1, 1, 1, 6	<pre>import torch.optim as</pre>
1, 1; 1, 1, <b>1</b> , <b>7</b>	<pre>from torch.autograd :</pre>
$\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ 8	import numpy as np
1, 1; 1; 1; 1; 1; 1; 9	<pre>import torchvision</pre>
1, 1, 1, 1, 1, 1, 1, 10	from torchvision impo
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1	<pre>import matplotlib.pyp</pre>
$1^{1}$ $1^{1}$ $1^{1}$ $1^{1}$ $1^{1}$ $1^{1}$ $1^{1}$ $1^{2}$	<pre>import time</pre>
111111111111111111111111111111111111	import copy
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	import os
21 1! <sup>1</sup> 1 <sup>1</sup> 1 <sup>1</sup> 1 <sup>1</sup> 1 <sup>1</sup>	from PIL import Image
$2^{1} 2^{1} 2^{1} 1^{1$	<pre>ImageFile.LOAD_TRUNCA</pre>
$2^{1} 2^{1} 2^{1} 2^{1} 1^{1} 1^{1} 1^{1} 1^{1}$	<pre>from fine_tuning_cont</pre>
2; 2; <sup>2; 2</sup> ; <sup>2</sup> ; <sup>1</sup> ; 18	
2, 2; <sup>2; 2; 2</sup> , <sup>2</sup> , <sup>2</sup> , <sup>19</sup>	<pre>## If you want to kee</pre>
2, 2, 2, 2, 2, 20	
$2^{1}_{21}$ $2^{1}_{21}$ $2^{1}_{21}$ $2^{1}_{21}$ $2^{1}_{21}$ $2^{1}_{21}$ $2^{1}_{21}$ $2^{1}_{21}$	<pre>if USE_TENSORBOARD:</pre>
2, 2, <sup>2</sup> ,	from pycrayon imp
2, 2, <sup>2</sup> ,	<pre>cc = CrayonClient</pre>
21 2 <sup>1</sup> 2 <sup>1</sup> 21 2 <sup>1</sup> 24	try:
2: 2: 2: 2: 2: 2: 2: 2: 2: 2: 2: 2: 2: 2	cc.remove_exp
3, 3, <sup>2, 2, 2</sup> , 2, 26	except:
3, 3, <sup>3</sup> , <sup>2</sup>	pass
3; 3; 3; 3; 2; 28	foo = cc.create_e
3, 3; <sup>3; 3; 3;</sup> 3, 3; 29	
3, <sup>3; 3; 3;</sup> 30	
3, 3; 5, 3; 31	<pre>## If you want to use</pre>
3, 5, 3, 3; 32	
3, 33	use_gpu = GPU_MODE
34	if use apu:

#### Files in that project





rt everythina we will be needina. port everythina we will be needina. import everythina we will be needina. s import everythina we will be needina. tt's import everythina we will be needina. let's import everythina we will be needina. c, let's import everythina we will be needina.

rt print\_function, division

n s optim import Variable

<mark>ort</mark> datasets, models, transforms plot <mark>as</mark> plt

eFile ATED\_IMAGES = True fig\_file import \*

eep a track of your network on tensorboard, set USE\_

port CrayonClient
t(hostname=TENSORBOARD\_SERVER)

periment(EXP\_NAME)

experiment(EXP\_NAME)

e the GPU, set GPU\_MODE TO 1 in config file

import torch
from torchvision import datasets

def train\_model():

def evaluate\_model():

**Code in these files** 





1 ### Section 1 ## Section 1 ## Section 1 ## Section 1 ## Section 1 ## Secti	<pre>1 - First, let's import everything we will be needing. on 1 - First, let's import everything we will be needing. tion 1 - First, let's import everything we will be needing. ection 1 - First, let's import everything we will be needing. Section 1 - First, let's import everything we will be needing. ## Section 1 - First, let's import everything we will be needing. ### Section 1 - First, let's import everything we will be needing.</pre>
	<pre>fromfuture import print_function, division</pre>
	import torch
1 5	<pre>import torch.nn as nn</pre>
$\frac{1}{1}$ 1 $(1)$ $(1)$ $(1)$ $(1)$ $(1)$	<pre>import torch.optim as optim</pre>
	from torch.autograd import Variable
$\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$	import numpy as np
	import torchvision
1, 1, <sup>1</sup> ,	from torchvision import datasets, models, transforms
1, 1, 1, 1, 1, 1, 1, 1	<pre>import matplotlib.pyplot as plt</pre>
$1^{1}_{1}$ 1 $(1^{1}_{1}$ 1 $(1^{1}_{1}$ 1 $(1^{1}_{1}$	import time
	import copy
111111111111111111111111111111111111	import os
11 + 11 + 11 + 11 + 15	from PIL import ImageFile
$21 \frac{1}{12} \frac{1}{12$	<pre>ImageFile.LOAD_TRUNCATED_IMAGES = True</pre>
$2^{-1}$ $2^{-1}$ $1^{-1}$ $1^{-1}$ $1^{-1}$ $1^{-1}$	<pre>from fine_tuning_config_file import *</pre>
$2^{-1}_{2}$ $2^{-1}_{2}$ $2^{-1}_{2}$ $1^{+1}_{2}$ $18$	
2, <sup>2</sup> , <sup>1</sup>	<pre>## If you want to keep a track of your network on tensorboard, set USE_</pre>
2, 2, 2, 2, 2, 20	
2; 2; <sup>2; 2; 2; 2; 21</sup>	if USE_TENSORBOARD:
2 <sup>1</sup> 2 <sup>1</sup> 2 <sup>1</sup> 2 <sup>1</sup> 2 <sup>1</sup> 2 <sup>2</sup> 2 <sup>2</sup> 2 <sup>2</sup>	from pycrayon import CrayonClient
2, 2, <sup>2</sup> ,	<pre>cc = CrayonClient(hostname=TENSORBOARD_SERVER)</pre>
21 21 21 21 21 21 24	try:
2: 2: <sup>2: 2</sup> 2: <sup>2</sup> 2: <sup>2</sup>	<pre>cc.remove_experiment(EXP_NAME)</pre>
$2^{1}$ $2^{1}$ $2^{1}$ $2^{1}$ $2^{1}$ $2^{1}$ $2^{1}$ $2^{1}$ $2^{1}$ $2^{1}$	except:
2, 3; <sup>3</sup> <sup>2</sup> 2; <sup>2</sup> 2; <sup>2</sup>	pass
2, 3, 3, 3, <sup>2</sup> 2, 3, <sup>3</sup> 3, <sup>2</sup> 3, <sup>2</sup> 2, <sup>28</sup>	<pre>foo = cc.create_experiment(EXP_NAME)</pre>
3, 3; <sup>3; 3;</sup> 3; 3; 3; 29	
3, 3, 3, 3, 3, 30	
3, <sup>3; 5</sup> , <sub>3</sub> , 31	<pre>## If you want to use the GPU, set GPU_MODE TO 1 in config file</pre>
3, 5, 3, 32	
3, 33	use_gpu = GPU_MODE
24	if was south

#### Files in that project





#### •What kind of files would I have?

#### •How would these be opened?

#### •Which ones need to be shared and how?









## What is Jupyter?

1		+				1	t.	C		13	aunc	her			>	<	E E	xplore &	Vis
_	÷.									а	+	Ж	õ	Ů	۲		C	Code	
0	Nam	1e				-		Last Mo	dified										
-	•	Diamon	ds.ipyni	b				6 minute	es ago				Plot	t prie	ces o	of d	liam	onds	by
ħ	• •	Explore Maps.ip	& Visua ynb	alize.ip;	ynb			a minut second	te ago is ago		[]	8]:	lib	rary	ggplo	ot2	)		
		My Note	book f	or Data	Scien	ce.ipy	nb	5 minute	es ago		r e		aan	lot(	liamor	nds	. ae	s(x=ca	rat
2		Play Vid	eos.ipy	nb				7 minute	es ago			1.	999	anda			, uc	found	lar
	<b>B</b>	medals.	csv					17 minute	es ago				NO R	ende	rer co	JUIC	d be	rouna	TOP
2	=	mtcars.	CSV					an ho	ur ago									11	
															-			•	
														150	- 00				
																			•••
														1000	- 00			100	
																	2		
														50	- 00	-		1	
																6	2		
															0-	2			
																		VSO	
																		432	-
																	2	53	•
														150	- 00			1.5	
																		14	
													e	100	- 00		6	2	
													pri	1001		1			
															-		7		
														500	- 00		5		
															,	1			
															0-	-			
																		VVS1	
																	-		
																	3		
														150	- 00		ē.	1	
														1000	- 00	-	1	-	
																1	2		
														EO	-00	2			
														200.0	JU -				



ualize.ipynb $\times$	💌 Maps.ipynb	×	Diamonds.joynb	×	Play Videos.ipynb	×	My Notebook for Data Sc	×
~			`				R	0
size, price, c	ut and clarity							
								i l
, y=price, co	lor=cut)) + geom_g	point() +	<pre>facet_wrap(~ clarit</pre>	y)				i I
output. It has	the following MIME	types:						
	SI2		SI1					
1.00	85							
•								
			1074 I					
	• <b>1</b> 7		<b>4</b>					
	18							
1	<b>7</b>		<b>F</b>					Sup
9								port
	VS1		VVS2					
	BMX"		<b>A.</b>					
			<b>1</b>	CL	it Sala			
	<b>2</b> .		<b>6</b>		Good			
					Very Good			
			r		Premium			
	r				Ideal			
/	1							
	IF	ò	2 3 4 5					
	15.1							
	21 °							
	<u>a</u>							
	D.							
	<b>*</b>							







## Intro to Google Colab



## Google Colab









## Summary of Part 1 and 2



#### GitHub



### Section	1 - First, let's import everything we will be needing.
### Sectio	on 1 – First, let's import everything we will be needing.
1 ### Sec	ection 1 - First, let's import everything we will be needing.
 	Section 1 - First, let's import everything we will be needing.
i 1 #1	## Section 1 - First, let's import everything we will be needing.
	<pre>### Section 1 - First, let's import everything we will be needing.</pre>
! <u> </u> 2	
	<pre>fromfuture import print_function, division</pre>
	import torch
ł ; , ( <sup>5</sup>	import torch.nn as nn
	<pre>import torch.optim as optim</pre>
1 7	from torch.autograd import Variable
1: 10 8	import numpy as np
$\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$ $\frac{9}{1}$	import torchvision
1: 1: 1: 1: 10	<pre>from torchvision import datasets, models, transforms</pre>
$1, \frac{1}{2}, \frac{1}{1}, \frac{1}{1}, \frac{11}{1}$	<pre>import matplotlib.pyplot as plt</pre>
$1! \frac{1}{1} \frac{1}{1} \frac{1}{1} \frac{1}{1}$	import time
$1(\frac{1!}{1},\frac{1}{1},\frac{1}{1},\frac{13}{1})$	import copy
1 1 1 1 1 1 14	import os
$11 \frac{1}{1} \frac{1}{1} \frac{1}{1} \frac{1}{1} \frac{15}{1}$	from PIL import ImageFile
1! <sup>1</sup> 1: 16	<pre>ImageFile.LOAD_TRUNCATED_IMAGES = True</pre>
21 11 11 17	<pre>from fine_tuning_config_file import *</pre>
$2^{2} \frac{2}{2} \frac{1}{2} \frac{1}{1!} \frac{18}{1!}$	
2; <sup>2;2</sup> ,2;19	<code>## If you want to keep a track of your network on tensorboard, set USE_T</code>
2: 2: 2: 20	
2, 2; 2, 2; 21	if USE_TENSORBOARD:
2! <sup>2, 2,</sup> 2; 22	from pycrayon import CrayonClient
21 <sup>2, 2</sup> , 23	<pre>cc = CrayonClient(hostname=TENSORBOARD_SERVER)</pre>
$2^{2} \frac{2}{2} \frac{2}{2} \frac{2}{2} \frac{2}{2}$	try:
21 <sup>2:21</sup> 21 <sup>2:21</sup> 21 <sup>25</sup>	<pre>cc.remove_experiment(EXP_NAME)</pre>
2! <sup>2</sup> 2 <sup>2</sup> 2 <sup>2</sup> 26	except:
31 <sup>2121</sup> 2127	pass
$3:\frac{3}{2}:\frac{2}{2}:\frac{2}{2}:28$	<pre>foo = cc.create_experiment(EXP_NAME)</pre>
3; 3; 5; 2; 29	
3: <sup>3: 3</sup> , 3: 30	
3, 3; 3, 3; 31	<pre>## If you want to use the GPU, set GPU_MODE T0 1 in config file</pre>
3, 3, 5, 32	
3, 33	use_gpu = GPU_MODE
34	if use_gpu:

#### **Jupyter Notebooks**





### import torch from torchvision import datasets def train\_model(): def evaluate\_model():

### Part 3





- What language/framework should we use to write code in our files?
- We will use PyTorch.





- Facebook's framework for ML/DL research.
- Easy to install.
- Easy to prototype.
- Resource for PyTorch: <u>https://pytorch.org/tutorials/beginner/</u> deep learning 60min blitz.html

# O PyTorch







## A very brief introduction to ML/DL





...and how PyTorch allows us to code for ML





# What is Machine Learning?

- Very elusive to define.
- Tom Mitchell (CMU): "A machine for which <u>performance improves</u> for some <u>specific task</u> with <u>experience</u>".









# Why is ML everywhere?



Unprocessed things



Machine Learning



## The toolkit perspective

### **INFORMATION GOES IN**

### MACHINE LEARNING MODEL







### **MORE USEFUL INFORMATION COMES OUT**









## EXAMPLES





Machine Learning System

#### Chair YES/NO





Assumption: Underlying relationship. We have reason to believe that:

- music preference, voting preference, and so on.
- Solution: Let's approximate g with f.

## What is f?



### $y_i = g(x_i) \quad \forall x_i \in X$

But, g is very, very complex/hard to pin point. Ex: protein folding,





# **Typically learning setups**

- Supervised: Both y (labels) and x (inputs) are given.
  - Ex: Is this a picture of chair? Can you translate this from English to Hindi/Mandarin/French?
- Unsupervised: Discovering patterns in x, independent of a y.
  - Ex: Anomaly detection in a subset of pictures, or predicting next word in a sequence.





# **Typically learning setups**

- Semi-supervised: Supervised + Unsupervised components.
- settings where multiple actions are taken in a sequence.



• Ex: MRI, CT scans - mixing vision + small, labeled doctor feedback.

**Reinforcement:** Really shouldn't be on the same list. Specifically cast for



# Supervised Learning $y_i = g(x_i) \sim f(x_i)$

- Goal, finding f which best approximates g.
- Let  $x_i \in X_{train}, y_i \in Y_{train}$ , is the goal to find the best fit f on  $(X_{train}, Y_{train})$ ?









- •f and g should be similar on all possible X,Y you will get.
- •This includes (1) Train set, (2) Test set, (3) Samples you don't have

•How should we quantify f ~ g?

L(f(x



 $\epsilon(f) = \int_{V \lor V} f(x) \sim g(x)$ 

$$(x), y) \rightarrow [0, inf)$$



 $\epsilon(f) = \int_{X \times Y} f(x) \sim g(x)$ 

 $L(f(x), y) \rightarrow [0, \inf)$ 

$$\epsilon(f) = \int_{X \times Y} L(y, f(x))$$

The goal of supervised learning:

 $\min_{f} \epsilon(f)$ 



- <u>function f on dataset space (X,Y)</u>
- Let's go over each one of these components



#### • We are trying to minimize, the loss function (L) incurred by our chosen



### Minimization: How do we do it?

• Gradient Descent:

(1) Measure gradient of loss w.r.t. parameter (say  $\alpha$ ),

(2) change parameter as:



$$\alpha = \alpha + r * \frac{\partial L}{\partial \alpha}$$









- Gradient descent should not work.
- We have other variants which work better in practice.





### So, we need to make the job easy.

- the function f.
- Let's see how we can modify each of them.





### • Components to play with - Loss function L, Data (X,Y), and architecture of



- It should reflect what is important to us. For classification:- don't penalize if prediction is same, and do penalize if classification is wrong.
- Should help optimization. Best to keep it continuous.
- For ex: Misclassification loss does not have good gradients. Hence, we use CrossEntropy loss for classification.
- If we have a custom use case, we can design a custom loss function.





# Making (X,Y) most useful

- shoe preference, favorite color.
- Conventional ML: features are everything.
- Finding the right features is very, very hard.



#### Features = facets of your data. Ex: For picking best advertisement - age,



# Making (X,Y) most useful

- Typical things done: PCA, mean normalization. More fancy ones whitening, sketching/streaming, random projections.
- Deep learning automates this search. First few layers = non-linear projection of your data.
- First few layers combine parts of input in non-linear way till they find something which is a good feature.









- Are we sure a CNN can be the mathematically best f?
  - Yes, universal approximation theorem proves that.
- If we're doing automatic search, why do certain architectures work better? Why CNNs, and not normal feed forward architectures?

# In the second se





### Architectures to help automatic search

- Addition of inductive bias - i.e. exploiting the structure of the problem.
- For ex:

• CNNs, RNNs are designed so this structure can be exploited.







I am going to the \_\_\_\_\_ because I am hungry.

I am hungry, so I am going to the \_\_\_\_\_.





- We were introduced to 4 components of Deep Learning today. These are:
- 1.Dataset: We want to load + Pre-Processing easily + fast.
- 2.Architectures: We may want to include specific structure of our problem.
- 3.Loss functions: Custom loss functions for our task are important.
- 4.Optimization: Variant of gradient descent used. So, gradient of loss w.r.t. NN parameters needs to be calculated.

# So, how do we code all this?







PyTorch allows great control over each of these aspects

Data-Loading: Automatically manages issues of memory and speed on GPU.

•Architectural expressivity: If you've ever used CAFFE, you would know.

libstdc++ installation

This route is not for the faint of heart. For OS X 10.10 and 10.9 you should install CUDA 7 and follow the instructions above. If that is not an option, take a deep breath and carry on.

Easy definition of losses and custom losses.

•Autograd.

•Easy implementation of different gradient algorithms.

# **OPyTorch** returns







## Let's see it in code?

https://colab.research.google.com/drive/125blGZwFZUm8g6YsRntzyJrrp-NjWkuW?authuser=1



