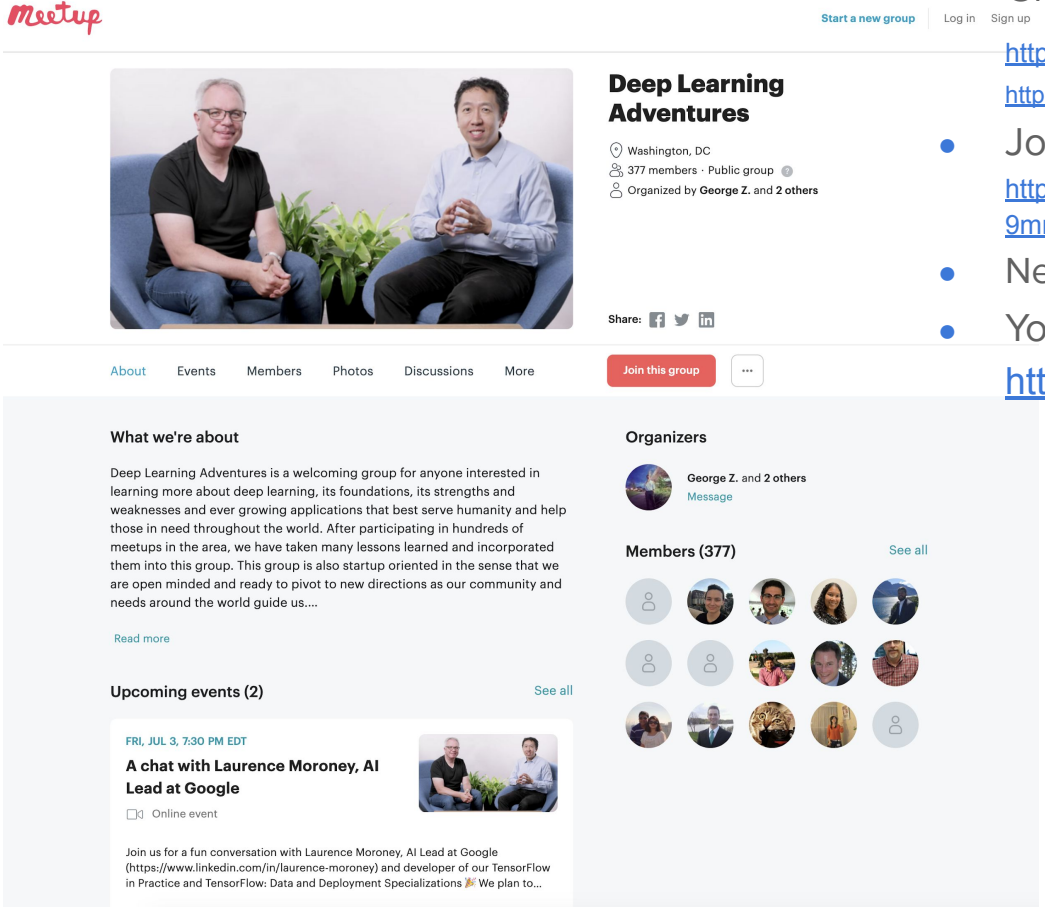
Deep Learning Adventures Chapter 1 - Presentation 1

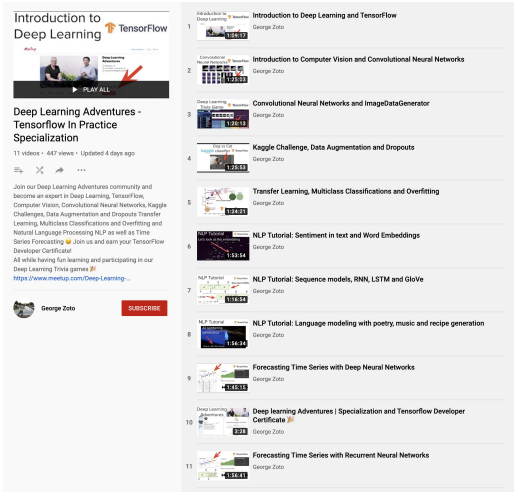
A quick overview of Coursera's Tensorflow in Practice specialization course

George Zoto
https://www.meetup.com/Deep-Learning-Adventures

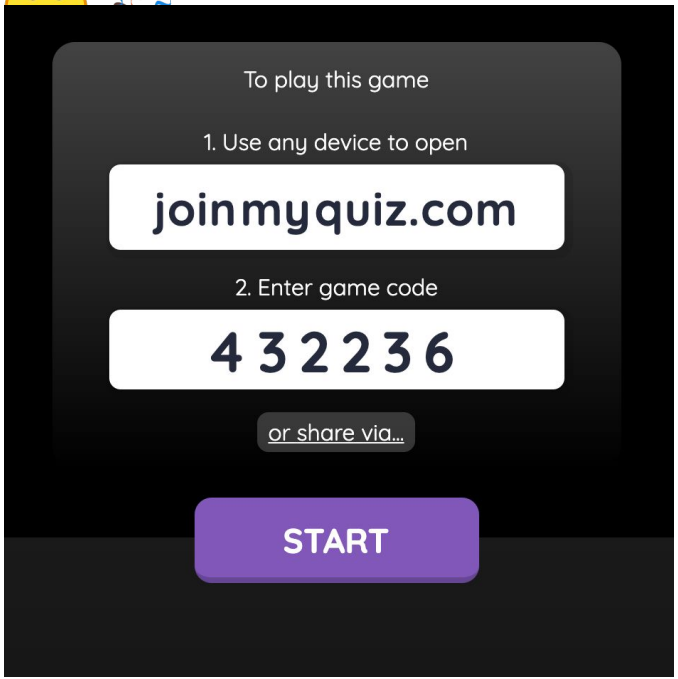
In the beginning...

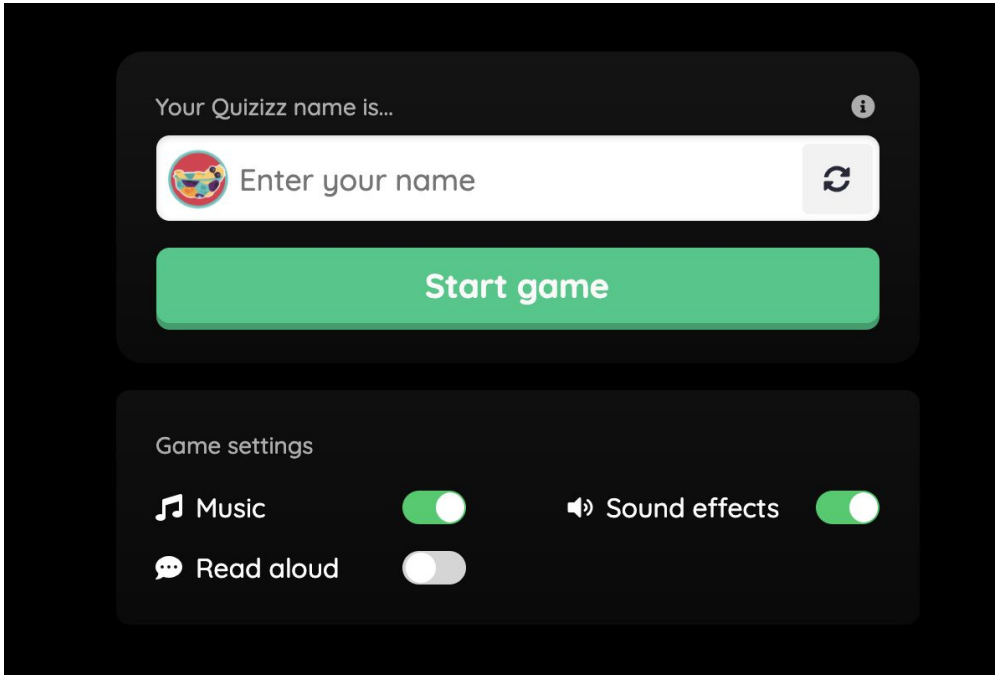
- Meetup Link
  https://www.meetup.com/Deep-Learning-Adventures
- GitHub repository
  https://github.com/georgezoto/Deep-Learning-Adventures
  https://github.com/georgezoto/TensorFlow-in-Practice
- Join us on Slack
  https://join.slack.com/t/deeplearninga-nmk8930/shared_invite/zt-d52h9mm9-h~Q0ZXw5PXsTDzPIINIvog
- Need a refresher or new to Deep Learning?
- YouTube recordings of all our Meetups 😀
  https://bit.ly/deep-learning-tf



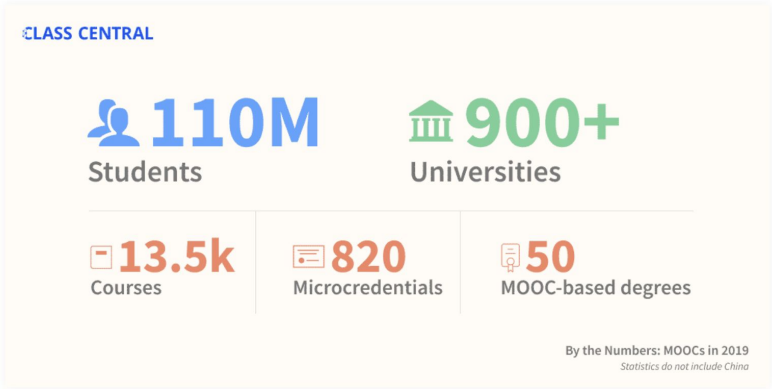


Not a typical Meetup... Get ready for a fun game 😮🎉





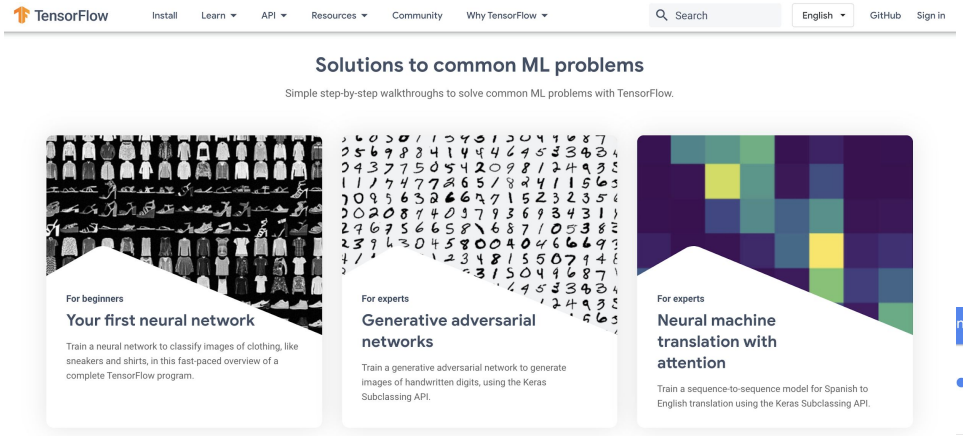
MOOCs, Coursera, TensorFlow

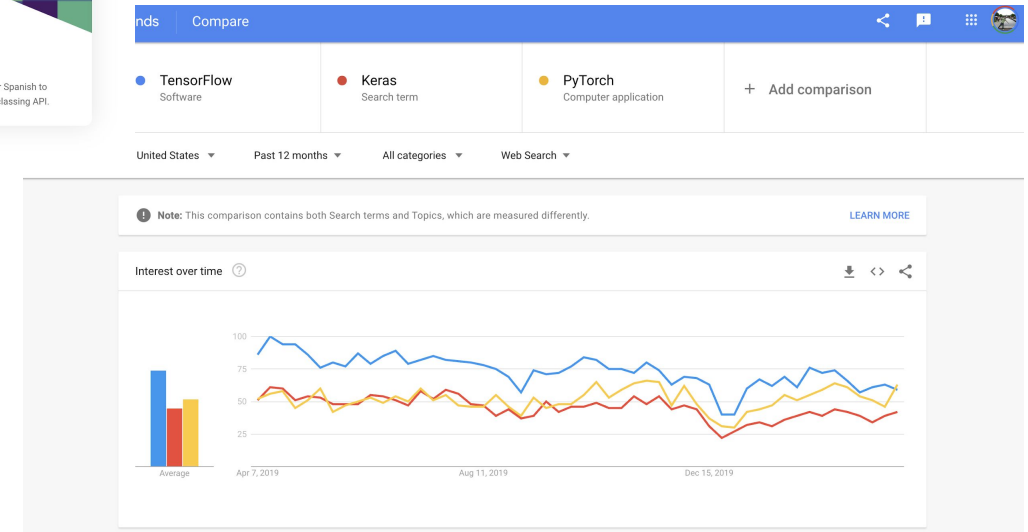


	Learners	Courses	Microcredentials	Degrees
Coursera	45 million	3,800	420	16
edX	24 million	2,640	292	10
Udacity	11.5 million	200	40	1
FutureLearn [2,4]	10 million	880	49	23
Swayam [2,3]	10 million	1,000	0	0

Source: https://www.classcentral.com/report/mooc-stats-2019/

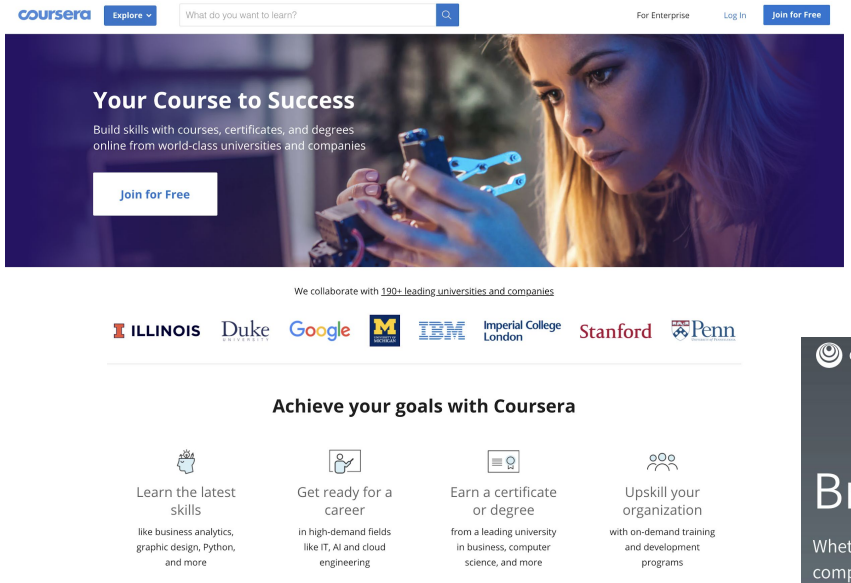
TensorFlow: An end-to-end open source machine learning platform

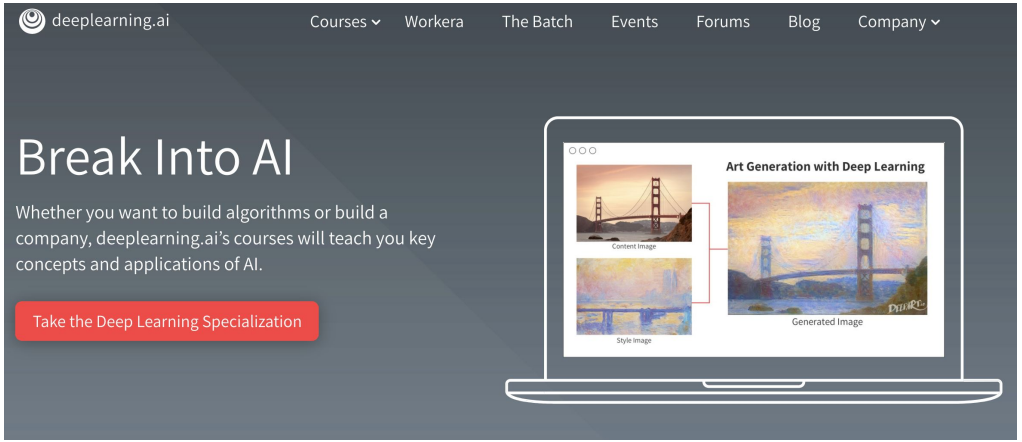




Source: https://www.tensorflow.org/

Attribution to Coursera and deeplearning.ai





Source:
https://www.coursera.org/about/terms
https://www.coursera.org/
https://www.deeplearning.ai/

Chapter 1 - TensorFlow in Practice Specialization

About this Specialization

199,621 recent views

Discover the tools software developers use to build scalable AI-powered algorithms in TensorFlow, a popular open-source machine learning framework.

In this four-course Specialization, you'll explore exciting opportunities for AI applications. Begin by developing an understanding of how to build and train neural networks. Improve a network's performance using convolutions as you train it to identify real-world images. You'll teach machines to understand, analyze, and respond to human speech with natural language processing systems. Learn to process text, represent sentences as vectors, and input data to a neural network. You'll even train an AI to create original poetry!

AI is already transforming industries across the world. After finishing this Specialization, you'll be able to apply your new TensorFlow skills to a wide range of problems and projects.

Looking for more advanced TensorFlow content? Check out the new **TensorFlow: Data and Deployment Specialization**.

Source: https://www.coursera.org/specializations/tensorflow-in-practice

Chapter 1 - TensorFlow in Practice Specialization

There are 4 Courses in this Specialization

COURSE 1

Introduction to TensorFlow for Artificial Intelligence, Machine Learning, and Deep Learning

★★★★½ 4.7 6,196 ratings • 1,282 reviews

If you are a software developer who wants to build scalable AI-powered algorithms, you need to understand how to use the tools to build them. This course is part of the upcoming Machine Learning in Tensorflow Specialization and will teach you best practices for using TensorFlow, a popular open-source framework for machine learning.

SHOW ALL

COURSE 2

Convolutional Neural Networks in TensorFlow

★★★★½ 4.7 2,751 ratings • 410 reviews

If you are a software developer who wants to build scalable AI-powered algorithms, you need to understand how to use the tools to build them. This course is part of the upcoming Machine Learning in Tensorflow Specialization and will teach you best practices for using TensorFlow, a popular open-source framework for machine learning.

SHOW ALL

COURSE 3

Natural Language Processing in TensorFlow

★★★★½ 4.6 2,037 ratings • 277 reviews

If you are a software developer who wants to build scalable AI-powered algorithms, you need to understand how to use the tools to build them. This Specialization will teach you best practices for using TensorFlow, a popular open-source framework for machine learning.

SHOW ALL

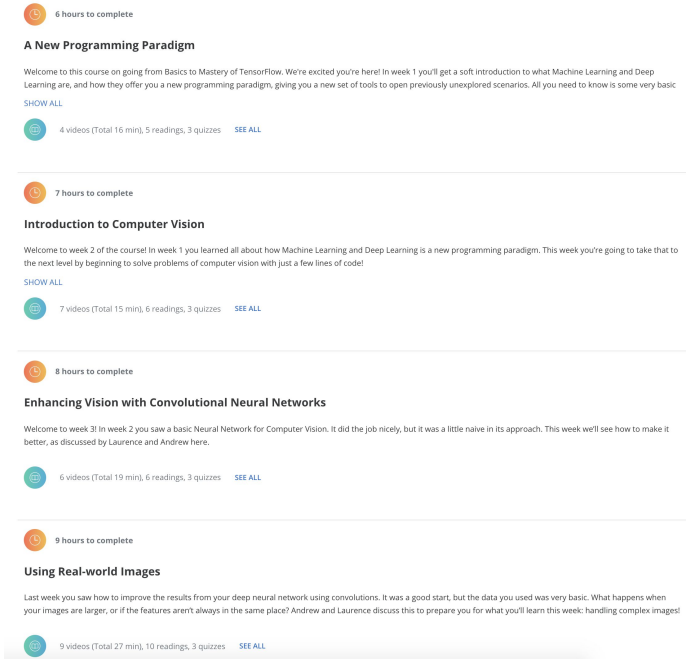
COURSE 4

Sequences, Time Series and Prediction

★★★★½ 4.6 1,374 ratings • 223 reviews

If you are a software developer who wants to build scalable AI-powered algorithms, you need to understand how to use the tools to build them. This Specialization will teach you best practices for using TensorFlow, a popular open-source framework for machine learning.

SHOW ALL



Source: https://www.coursera.org/specializations/tensorflow-in-practice

Setup

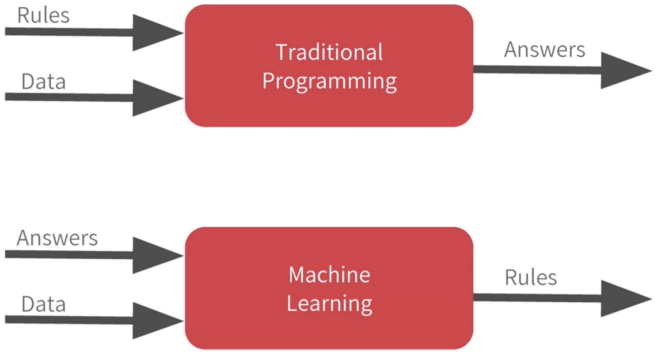
Colaboratory is a free Jupyter notebook environment that requires no setup and runs entirely in the cloud. You can write and execute code, save and share your analyses, and access powerful computing resources, all for free from your browser.

https://colab.research.google.com

Course 1: Introduction to TensorFlow for Artificial Intelligence, Machine Learning, and Deep Learning

Week 1: A New Programming Paradigm

A New Programming Paradigm



deeplearning.ai

Activity Recognition

```
if(speed<4){
    status=WALKING;
}
```

```
if(speed<4){
    status=WALKING;
} else {
    status=RUNNING;
}
```

```
if(speed<4){
    status=WALKING;
} else if(speed<12){
    status=RUNNING;
} else {
    status=BIKING;
}
```

```
// Oh crap
```

deeplearning.ai

A New Programming Paradigm

Activity Recognition

```
01010010101001010101
10010101010011101
01001010100101010010
01010010101001010010
```

Label = WALKING

```
10101001010010101010
01010100100100100001
00100111111010101111
10101001001111010011
```

Label = RUNNING

```
10010100111111010101
11010101110101011110
10101011110101010011
11111100011111010101
```

Label = BIKING

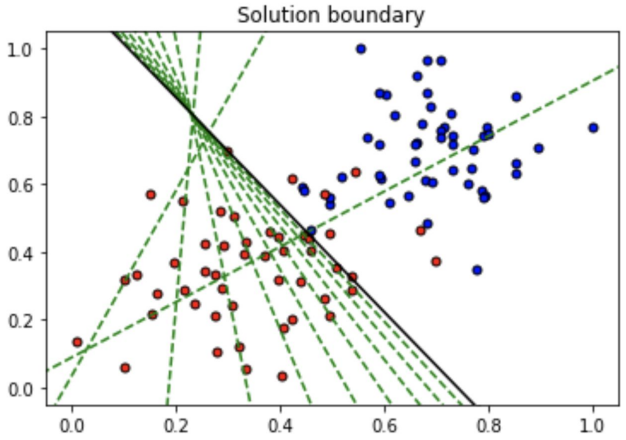
```
11111111110100111101
00111111010111110101
01011101010101011110
10101010100100111110
```

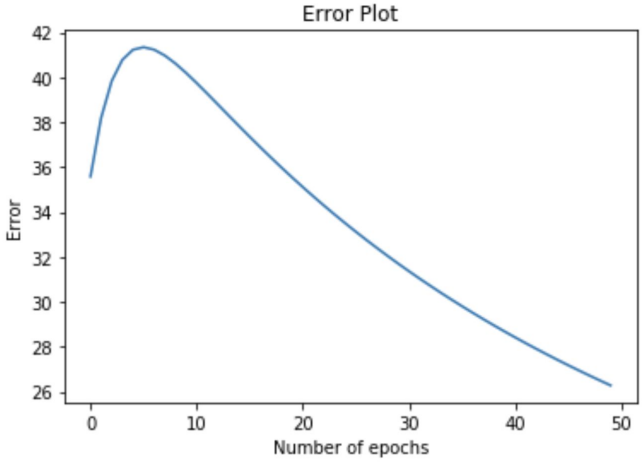
Label = GOLFING
(Sort of)

deeplearning.ai

$$X = -1, \quad 0, \quad 1, \quad 2, \quad 3, \quad 4$$

$$Y = -3, \quad -1, \quad 1, \quad 3, \quad 5, \quad 7$$





A New Programming Paradigm

```
model = keras.Sequential([keras.layers.Dense(units=1, input_shape=[1])])
model.compile(optimizer='sgd', loss='mean_squared_error')

xs = np.array([-1.0, 0.0, 1.0, 2.0, 3.0, 4.0], dtype=float)
ys = np.array([-3.0, -1.0, 1.0, 3.0, 5.0, 7.0], dtype=float)

model.fit(xs, ys, epochs=500)

print(model.predict([10.0]))
```

A New Programming Paradigm

The Hello World of Deep Learning with Neural Networks

Like every first app you should start with something super simple that shows the overall scaffolding for how your code works.

In the case of creating neural networks, the sample I like to use is one where it learns the relationship between two numbers. So, for example, if you were writing code for a function like this, you already know the 'rules' —

```
float hw_function(float x){
    float y = (2 * x) - 1;
    return y;
}
```

So how would you train a neural network to do the equivalent task? Using data! By feeding it with a set of Xs, and a set of Ys, it should be able to figure out the relationship between them.

This is obviously a very different paradigm than what you might be used to, so let's step through it piece by piece.

Imports

Let's start with our imports. Here we are importing TensorFlow and calling it tf for ease of use.

We then import a library called numpy, which helps us to represent our data as lists easily and quickly.

The framework for defining a neural network as a set of Sequential layers is called keras, so we import that too.

```
[2] import tensorflow as tf
    import numpy as np
    from tensorflow import keras
```

```
[3] print(tf.__version__)
```

```
2.2.0-rc2
```

```
[4] print(keras.__version__)
```

```
2.3.0-tf
```

Define and Compile the Neural Network

Next we will create the simplest possible neural network. It has 1 layer, and that layer has 1 neuron, and the input shape to it is just 1 value.

```
[5] model = tf.keras.Sequential([keras.layers.Dense(units=1, input_shape=[1])])
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 1)                 2
=================================================================
Total params: 2
Trainable params: 2
Non-trainable params: 0
_________________________________________________________________
```

A New Programming Paradigm

Over time you will learn the different and appropriate loss and optimizer functions for different scenarios.

```
[7]  model.compile(optimizer='sgd', loss='mean_squared_error')
```

Providing the Data

Next up we'll feed in some data. In this case we are taking 6 xs and 6ys. You can see that the relationship between these is that y=2x-1, so where x = -1, y=-3 etc. etc.

A python library called 'Numpy' provides lots of array type data structures that are a defacto standard way of doing it. We declare that we want to use these by specifying the values as an np.array[]

```
[10] xs = np.array([-1.0,  0.0, 1.0, 2.0, 3.0, 4.0], dtype=float)
     ys = np.array([-3.0, -1.0, 1.0, 3.0, 5.0, 7.0], dtype=float)
```

A New Programming Paradigm

Training the Neural Network

The process of training the neural network, where it 'learns' the relationship between the Xs and Ys is in the **model.fit** call. This is where it will go through the loop we spoke about above, making a guess, measuring how good or bad it is (aka the loss), using the opimizer to make another guess etc. It will do it for the number of epochs you specify. When you run this code, you'll see the loss on the right hand side.

```
[13] model.fit(xs, ys, epochs=300)
```

```
Epoch 1/300
1/1 [==============================] - 0s 2ms/step - loss: 0.2848
Epoch 2/300
1/1 [==============================] - 0s 1ms/step - loss: 0.2789
Epoch 3/300
1/1 [==============================] - 0s 3ms/step - loss: 0.2732
Epoch 4/300
1/1 [==============================] - 0s 3ms/step - loss: 0.2676
Epoch 5/300
1/1 [==============================] - 0s 1ms/step - loss: 0.2621
Epoch 6/300
1/1 [==============================] - 0s 1ms/step - loss: 0.2567
Epoch 7/300
1/1 [==============================] - 0s 1ms/step - loss: 0.2514
```

A New Programming Paradigm

```
1/1 [==============================] - 0s 2ms/step - loss: 5.8673e-04
Epoch 300/300
1/1 [==============================] - 0s 2ms/step - loss: 5.7468e-04
<tensorflow.python.keras.callbacks.History at 0x7fe71cb594e0>
```

Ok, now you have a model that has been trained to learn the relationshop between X and Y. You can use the **model.predict** method to have it figure out the Y for a previously unknown X. So, for example, if X = 10, what do you think Y will be? Take a guess before you run this code:

```
[14] print(model.predict([10.0]))
```

```
[[18.93006]]
```

You might have thought 19, right? But it ended up being a little under. Why do you think that is?

A New Programming Paradigm

That brings you to the end of what you need to look at for Week 1. If you're eager to learn more, before we go to Week 2, there are some great resources you can check out:

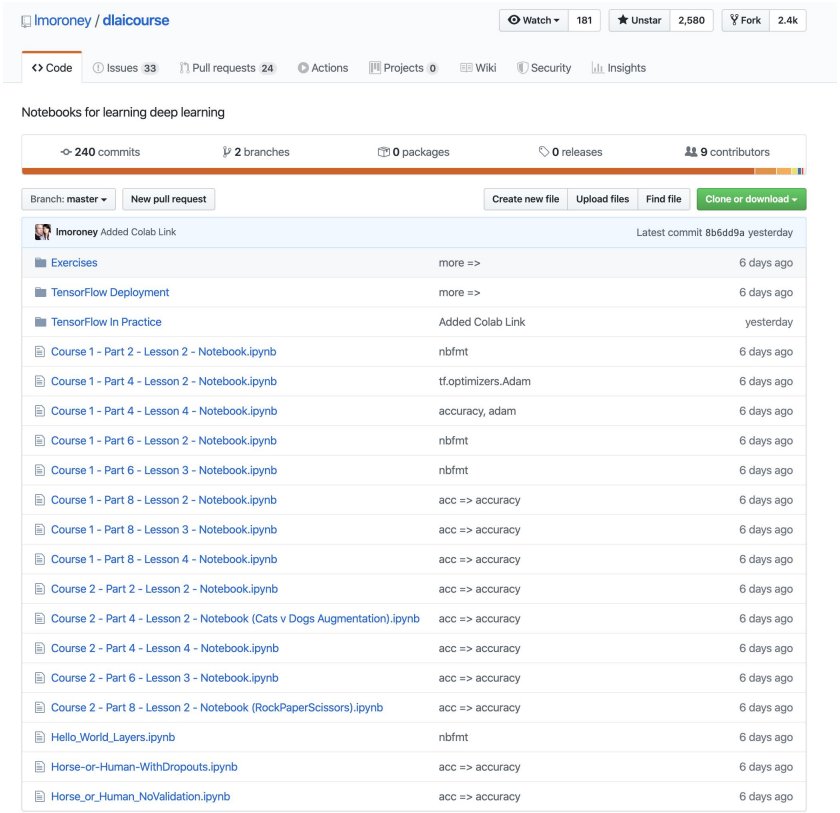
- AI For Everyone is a non-technical course that will help you understand many of the AI technologies we will discuss later in this course, and help you spot opportunities in applying this technology to solve your problems. https://www.deeplearning.ai/ai-for-everyone/

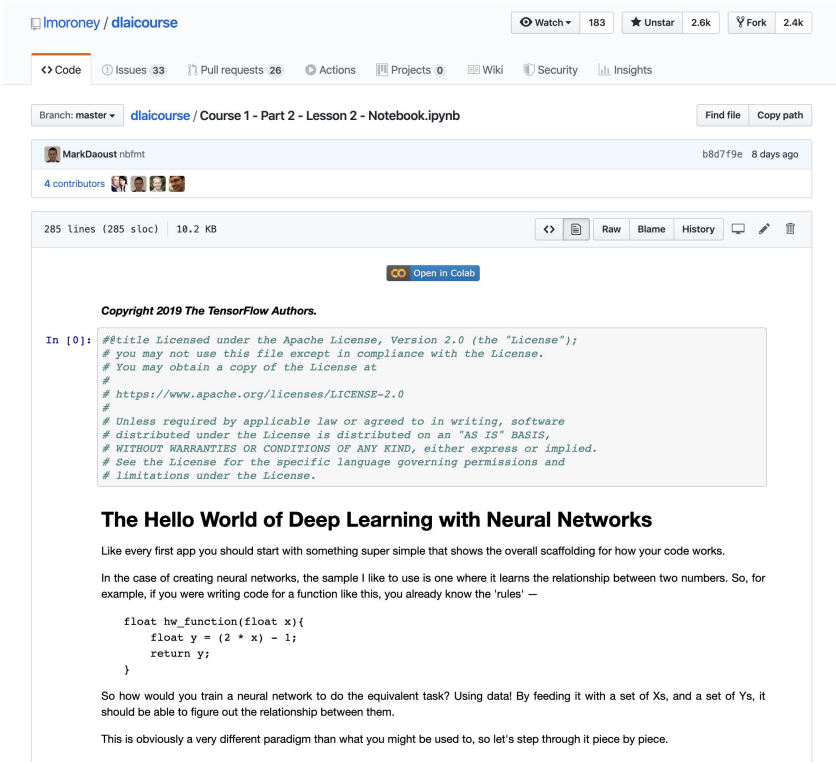
- TensorFlow is available at TensorFlow.org, and video updates from the TensorFlow team are at youtube.com/tensorflow

Play with a neural network right in the browser at http://playground.tensorflow.org. See if you can figure out the parameters to get the neural network to pattern match to the desired groups. The spiral is particularly challenging!

The 'Hello World' notebook that we used in this course is available on GitHub here.

A New Programming Paradigm





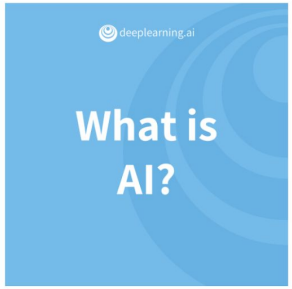
Source: https://github.com/lmoroney/dlaicourse

Check out these resources

AI For Everyone

AI is not only for engineers. “AI for Everyone”, a non-technical course, will help you understand AI technologies and spot opportunities to apply AI to problems in your own organization. You will see examples of what today’s AI can – and cannot – do. Finally, you will understand how AI is impacting society and how to navigate through this technological change.

If you are a non-technical business professional, “AI for Everyone” will help you understand how to build a sustainable AI strategy. If you are a machine learning engineer or data scientist, this is the course to ask your manager, VP or CEO to take if you want them to understand what you can (and cannot!) do.



Week 1
What is AI

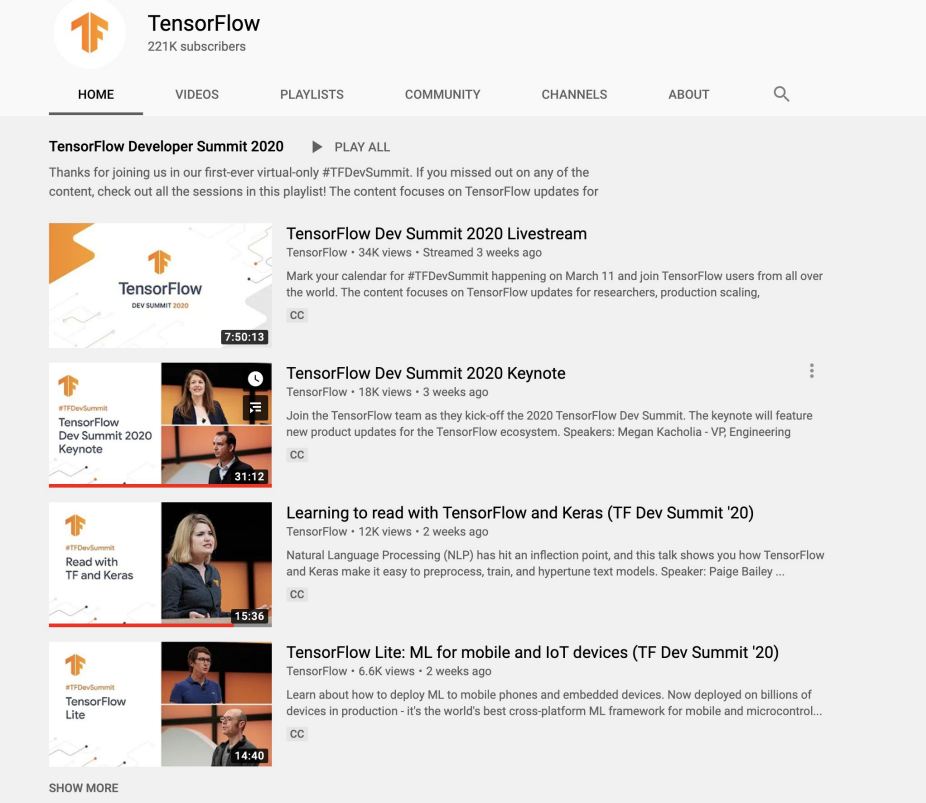
Week 2
Building AI Projects

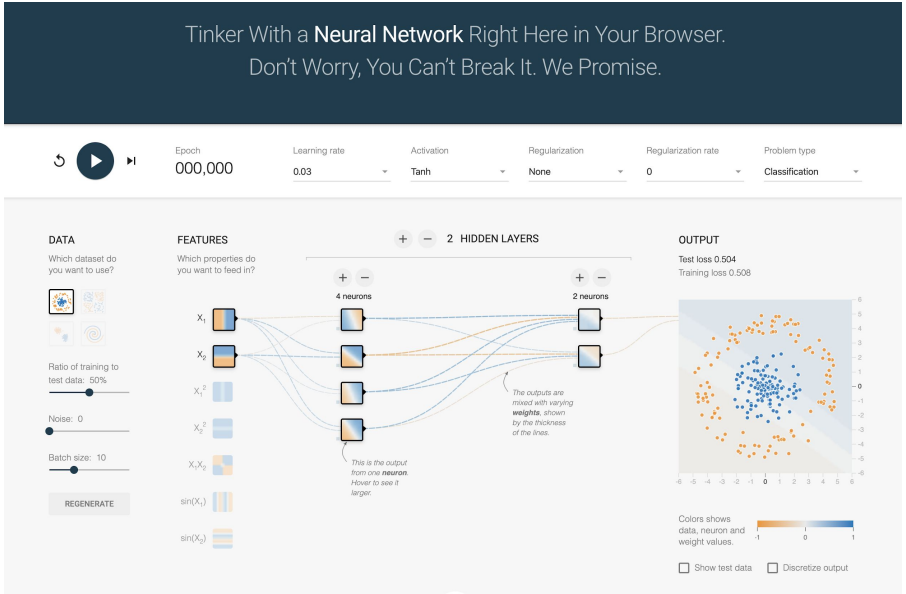
Week 3
AI in Your Company

Week 4
AI and Society

Source: https://www.deeplearning.ai/ai-for-everyone/

Check out these resources





Source:
https://www.youtube.com/tensorflow
http://playground.tensorflow.org/

Check out these events and Meetups

New AI For Medicine Specialization coming soon!

Saturday, March 28, 2020

AI Meet & Greet

Hosted by
viraf and 2 others

Thursday, April 2, 2020

Virtual Event : Session 3 - [Udacity] Intro to Deep Learning using PyTorch

Hosted by
Pragyansmita Nayak, Ph.D and Data Community DC (DC2)

Thursday, April 2, 2020

Remote Session: Attention Is All You Need

Hosted by
Julius

Thursday, March 26, 2020

Online: Using Wikipedia and Wikidata for Natural Language Processing

Hosted by
Seth Grimes

Source:
https://www.youtube.com/watch?v=zGFKSQlef_0
https://www.meetup.com/DC-NLP/events/269502685
https://www.meetup.com/Bethesda-Artificial-Intelligence-Meetup
https://www.meetup.com/novadeeplearning
https://www.meetup.com/Machine-Learning-Paper-Club

Check out this new certification and online conference

TensorFlow Core

Introducing the TensorFlow Developer Certificate!

March 12, 2020

Posted by Alina Shinkarsky, on behalf of the TensorFlow Team

In the AI world today, more and more companies are looking to hire machine learning talent, and simultaneously, an increasing number of students and developers are looking for ways to gain and showcase their ML knowledge with formal recognition. In addition to the courses and learning resources available online, we want to help developers showcase their ML proficiency and help companies hire ML developers to solve challenging problems.

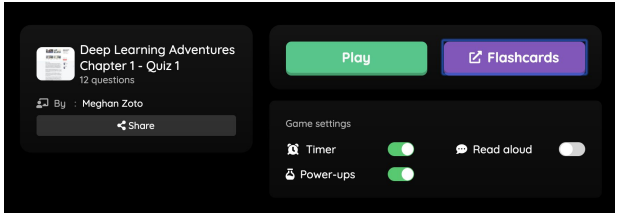
NVIDIA DEEP LEARNING INSTITUTE

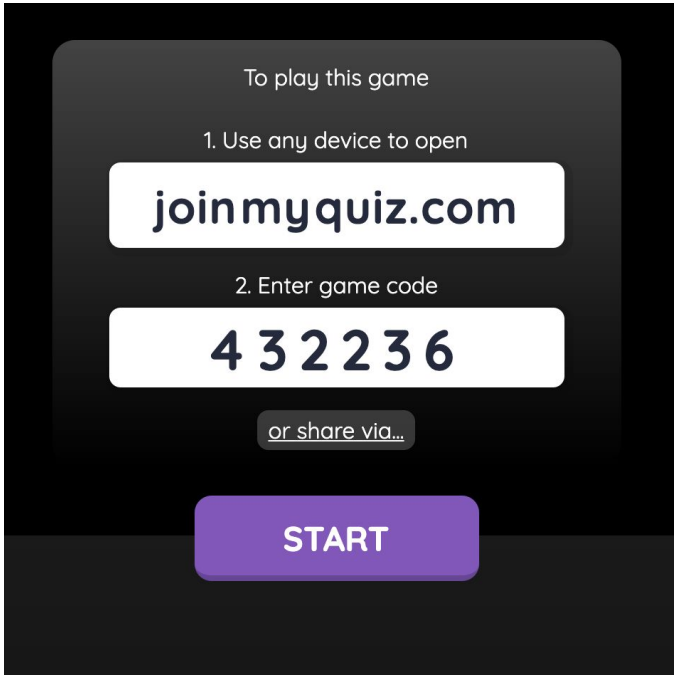
Workshop Outline

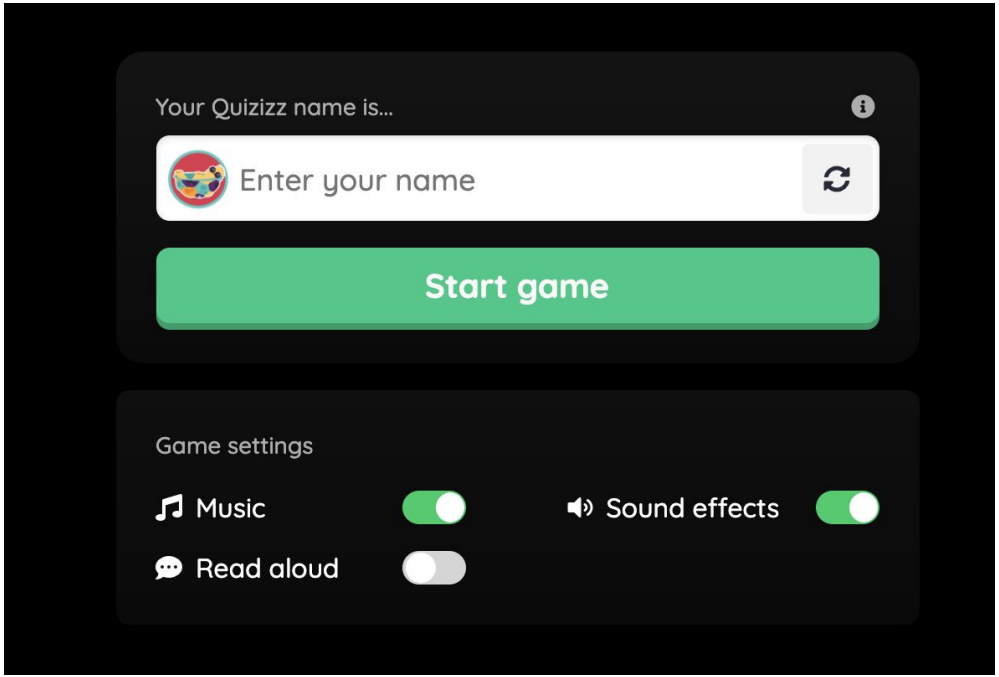
TOPIC	DESCRIPTION
Introduction (15 mins)	> Meet the instructor. > Create an account at **courses.nvidia.com/join**
Image Segmentation with TensorFlow (120 mins)	> Compare image segmentation to other computer vision problems. > Experiment with TensorFlow tools. > Implement effective metrics for assessing model performance.
Break (60 mins)	
Word Generation with TensorFlow (120 mins)	> Learn about natural language processing (NLP) and recurrent neural networks (RNNs). > Create network inputs from text data. > Test with new data and iterate to improve performance.
Break (15 mins)	
Image and Video Captioning (120 mins)	> Combine computer vision and natural language processing to describe scenes. > Learn to harness the functionality of convolutional neural networks (CNNs) and RNNs.
Final Review (15 mins)	> Review key learnings and wrap up questions. > Complete the assessment to earn a certificate. > Take the workshop survey.

Source:
https://blog.tensorflow.org/2020/03/introducing-tensorflow-developer-certificate.html?m=1
https://www.nvidia.com/en-us/gtc/

Time for a fun game 😀🎉



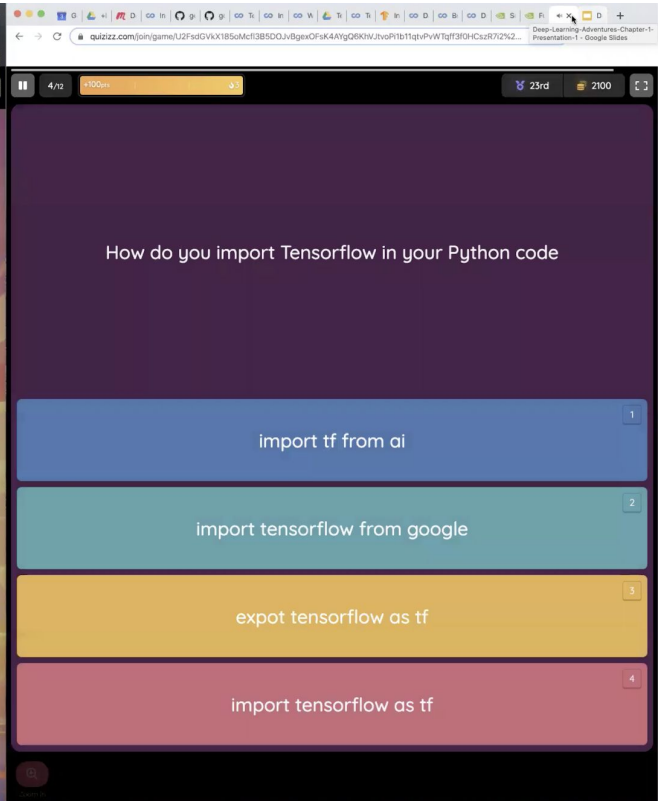




Practice here or use Flashcards:
quizizz.com/join/quiz/5e87bdbc07fa7f001b120404/start?from=soloLinkShare&referrer=5d921444d0fa99001a135336

Time for a fun game 😀🎉

Practice here or use Flashcards:
https://quizizz.com/join/quiz/5e87bdbc07fa7f001b120404/start?from=soloLinkShare&referrer=5d921444d0fa99001a135336



Questions

Discussion

Standard notations for Deep Learning

This document has the purpose of discussing a new standard for deep learning mathematical notations.

1 Neural Networks Notations.

General comments:

· superscript (i) will denote the i^{th} training example while superscript [l] will denote the l^{th} layer

Sizes:

·m : number of examples in the dataset

·n_x : input size

·n_y : output size (or number of classes)

·$n_h^{[l]}$: number of hidden units of the l^{th} layer

In a for loop, it is possible to denote $n_x = n_h^{[0]}$ and $n_y = n_h{}^{[\text{number of layers } +1]}$.

·L : number of layers in the network.

Objects:

·$X \in \mathbb{R}^{n_x \times m}$ is the input matrix

·$x^{(i)} \in \mathbb{R}^{n_x}$ is the i^{th}example represented as a column vector

·$Y \in \mathbb{R}^{n_y \times m}$ is the label matrix

·$y^{(i)} \in \mathbb{R}^{n_y}$ is the output label for the i^{th} example

·$W^{[l]} \in \mathbb{R}^{\text{number of units in next layer} \times \text{number of units in the previous layer}}$ is the weight matrix,superscript [l] indicates the layer

·$b^{[l]} \in \mathbb{R}^{\text{number of units in next layer}}$ is the bias vector in the l^{th} layer

·$\hat{y} \in \mathbb{R}^{n_y}$ is the predicted output vector. It can also be denoted $a^{[L]}$ where L is the number of layers in the network.

Common forward propagation equation examples:

$a = g^{[l]}(W_x x^{(i)} + b_1) = g^{[l]}(z_1)$ where $g^{[l]}$ denotes the l^{th} layer activation function

$\hat{y}^{(i)} = softmax(W_h h + b_2)$

· General Activation Formula: $a_j^{[l]} = g^{[l]}(\sum_k w_{jk}^{[l]} a_k^{[l-1]} + b_j^{[l]}) = g^{[l]}(z_j^{[l]})$

· $J(x, W, b, y)$ or $J(\hat{y}, y)$ denote the cost function.

Examples of cost function:

· $J_{CE}(\hat{y}, y) = -\sum_{i=0}^{m} y^{(i)} \log \hat{y}^{(i)}$

· $J_1(\hat{y}, y) = \sum_{i=0}^{m} \mid y^{(i)} - \hat{y}^{(i)} \mid$

2 Deep Learning representations

For representations:

· nodes represent inputs, activations or outputs

· edges represent weights or biases

Here are several examples of Standard deep learning representations

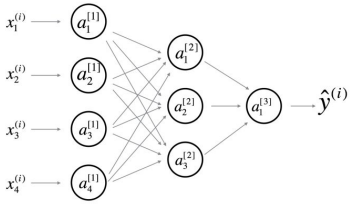


Figure 1: Comprehensive Network: representation commonly used for Neural Networks. For better aesthetic, we omitted the details on the parameters ($w_{ij}^{[l]}$ and $b_i^{[l]}$ etc...) that should appear on the edges

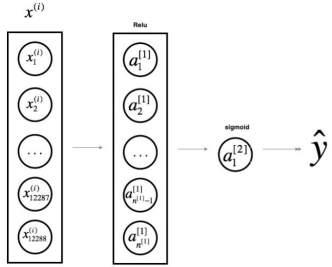


Figure 2: Simplified Network: a simpler representation of a two layer neural network, both are equivalent.

Content added after our 1st session, based on group feedback

George Zoto 1:53 PM

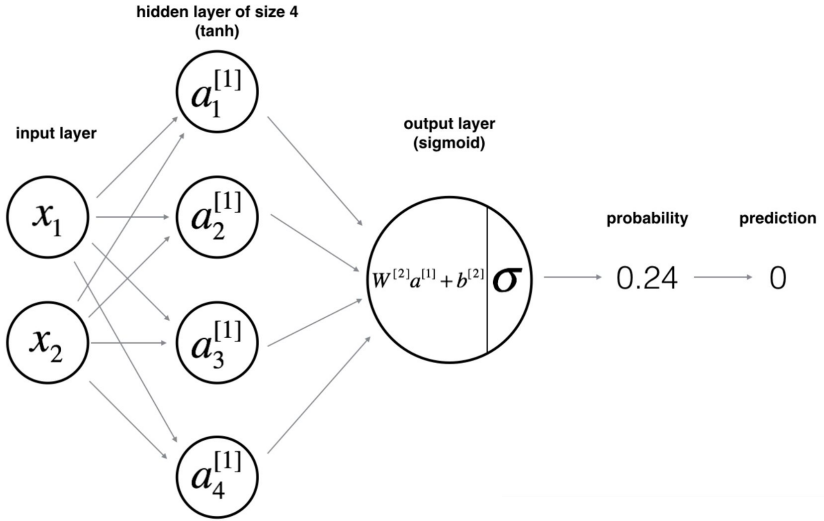
Hi friends! Looking at our comments last night about our first model and what is it really doing under the hood, I updated our code repository to include:

- A sample architecture of a 2 layer network (1 hidden + 1 output layer). Our model is even simpler than this, with just 1 neuron in the hidden layer.
- A custom callback function that is called during training
- A detailed chart plotted after each epoch showing how our model's weight (w) and bias (b) are adjusted by the optimizer (sgd) to minimize our loss function (mean_squared_error)

Our model was even simpler than this, with just 1 neuron in the hidden layer as its only output layer.

Sample architecture of a 2 layer network (1 hidden + 1 output layer)

It has 2 inputs/features (x_1 and x_2), 1 hidden layer of 4 nodes/neurons (a_1 through a_4) and final output layer of 1 node/neuron



Our code is available on GitHub and on our Slack channel:
https://github.com/georgezoto/TensorFlow-in-Practice/blob/master/Course-1-Introduction-to-TensorFlow/C1W1_A_new_programming_paradigm_1.ipynb

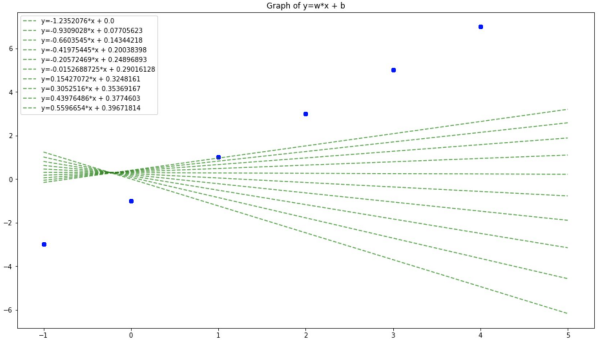
#tensorflow-in-practice-specialization

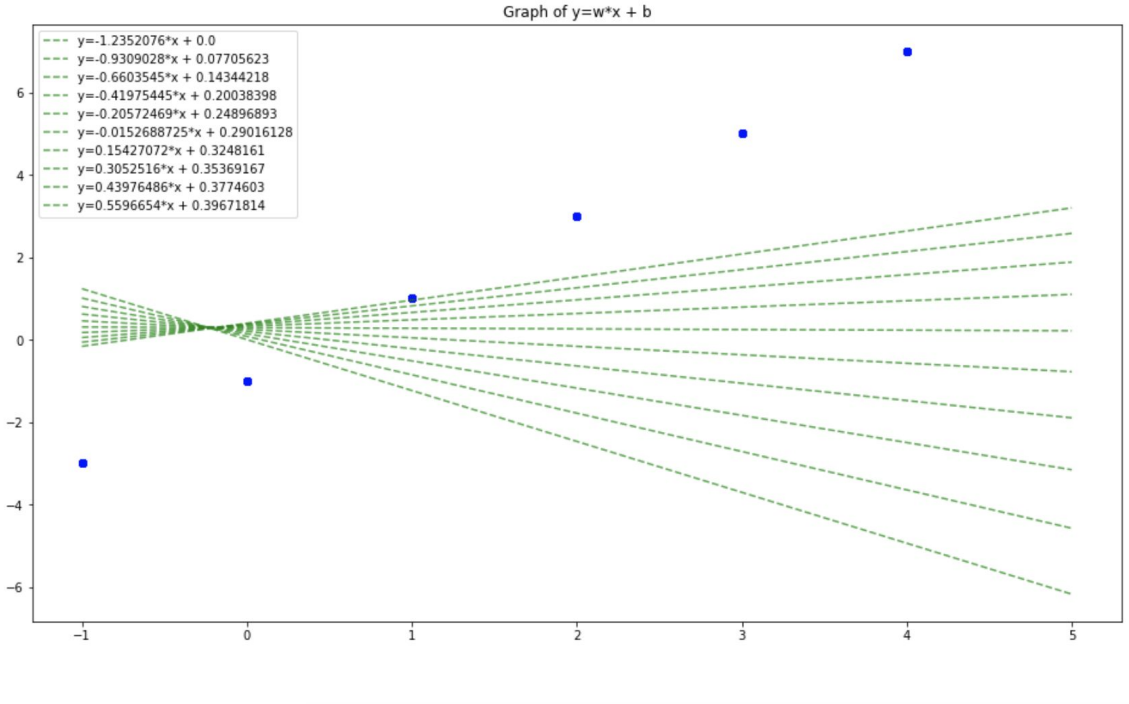
You created this channel on March 31st. This is the very beginning of the **#tensorflow-in-practice-specialization** channel.

Content added after our 1st session, based on group feedback

```
model.fit(xs, ys, epochs=10, callbacks=[get_weight_and_bias()])

Epoch 1/10
-1.2352076 0.0
1/1 [==============================] - 0s 2ms/step - loss: 45.3716
Epoch 2/10
-0.9309028 0.07705623
1/1 [==============================] - 0s 2ms/step - loss: 36.0725
Epoch 3/10
-0.6603545 0.14344218
1/1 [==============================] - 0s 2ms/step - loss: 28.7486
Epoch 4/10
-0.41975445 0.20038398
1/1 [==============================] - 0s 1ms/step - loss: 22.9789
Epoch 5/10
-0.20572469 0.24896893
1/1 [==============================] - 0s 1ms/step - loss: 18.4323
Epoch 6/10
-0.0152688725 0.29016128
1/1 [==============================] - 0s 1ms/step - loss: 14.8479
Epoch 7/10
0.15427072 0.3248161
1/1 [==============================] - 0s 1ms/step - loss: 12.0208
Epoch 8/10
0.3052516 0.35369167
1/1 [==============================] - 0s 1ms/step - loss: 9.7895
Epoch 9/10
0.43976486 0.3774603
1/1 [==============================] - 0s 1ms/step - loss: 8.0273
Epoch 10/10
0.5596654 0.39671814
1/1 [==============================] - 0s 1ms/step - loss: 6.6342
<tensorflow.python.keras.callbacks.History at 0x7f77aee0da20>
```





Our code is available on GitHub and on our Slack channel:
https://github.com/georgezoto/TensorFlow-in-Practice/blob/master/Course-1-Introduction-to-TensorFlow/C1W1_A_new_programming_paradigm_1.ipynb

#tensorflow-in-practice-specialization

You created this channel on March 31st. This is the very beginning of the **#tensorflow-in-practice-specialization** channel.