Course summary

Here are the course summary as its given on the course link:

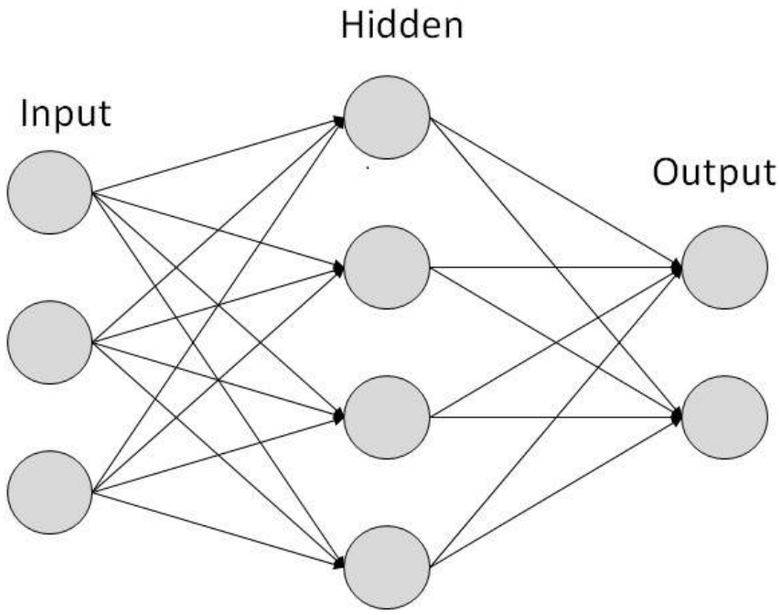
> If you want to break into cutting-edge AI, this course will help you do so. Deep learning engineers are highly sought after, and mastering deep learning will give you numerous new career opportunities. Deep learning is also a new "superpower" that will let you build AI systems that just weren't possible a few years ago.
>
> In this course, you will learn the foundations of deep learning. When you finish this class, you will:
>
> - Understand the major technology trends driving Deep Learning
> - Be able to build, train and apply fully connected deep neural networks
> - Know how to implement efficient (vectorized) neural networks
> - Understand the key parameters in a neural network's architecture
>
> This course also teaches you how Deep Learning actually works, rather than presenting only a cursory or surface-level description. So after completing it, you will be able to apply deep learning to a your own applications. If you are looking for a job in AI, after this course you will also be able to answer basic interview questions.

Introduction to deep learning

> Be able to explain the major trends driving the rise of deep learning, and understand where and how it is applied today.

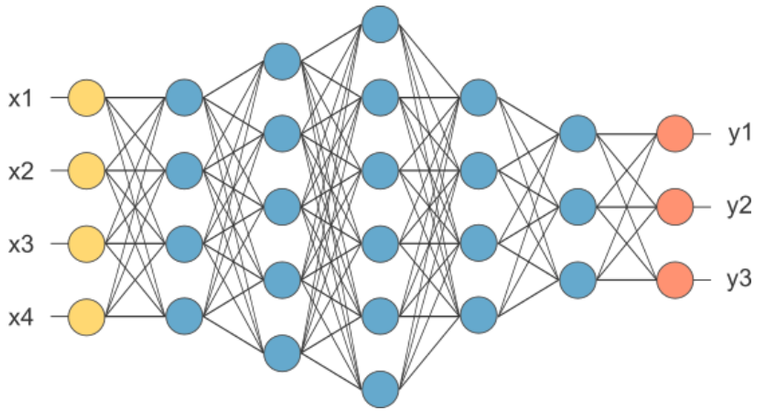
What is a (Neural Network) NN?

- Single neuron == linear regression without applying activation(perceptron)
- Basically a single neuron will calculate weighted sum of input(W.T*X) and then we can set a threshold to predict output in a perceptron. If weighted sum of input cross the threshold, perceptron fires and if not then perceptron doesn't predict.
- Perceptron can take real values input or boolean values.
- Actually, when w·x+b=0 the perceptron outputs 0.
- Disadvantage of perceptron is that it only output binary values and if we try to give small change in weight and bais then perceptron can flip the output. We need some system which can modify the output slightly according to small change in weight and bias. Here comes sigmoid function in picture.
- If we change perceptron with a sigmoid function, then we can make slight change in output.
- e.g. output in perceptron = 0, you slightly changed weight and bias, output becomes = 1 but actual output is 0.7. In case of sigmoid, output1 = 0, slight change in weight and bias, output = 0.7.
- If we apply sigmoid activation function then Single neuron will act as Logistic Regression.
- we can understand difference between perceptron and sigmoid function by looking at sigmoid function graph.
- Simple NN graph:



  - Image taken from tutorialspoint.com
- RELU stands for rectified linear unit is the most popular activation function right now that makes deep NNs train faster now.
- Hidden layers predicts connection between inputs automatically, thats what deep learning is good at.

- Deep NN consists of more hidden layers (Deeper layers)



  - Image taken from opennn.net

- Each Input will be connected to the hidden layer and the NN will decide the connections.
- Supervised learning means we have the (X,Y) and we need to get the function that maps X to Y.

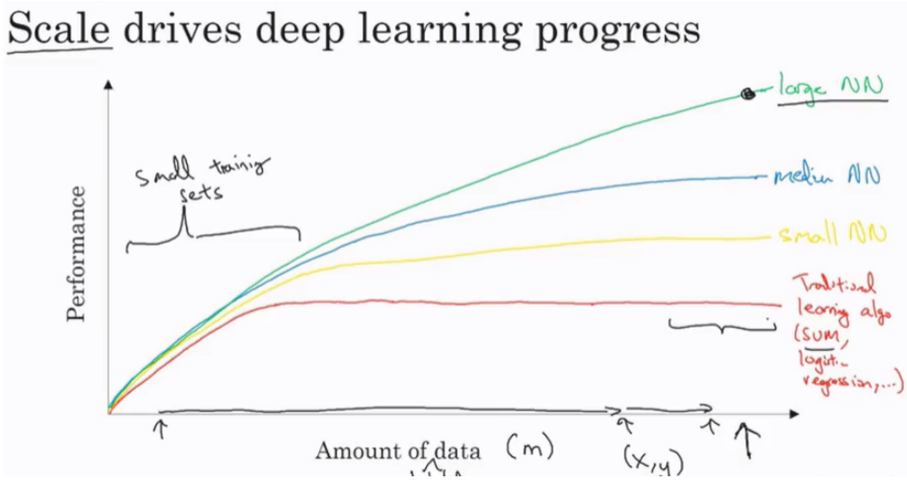
Supervised learning with neural networks

- Different types of neural networks for supervised learning which includes:
  - CNN or convolutional neural networks (Useful in computer vision)
  - RNN or Recurrent neural networks (Useful in Speech recognition or NLP)
  - Standard NN (Useful for Structured data)
  - Hybrid/custom NN or a Collection of NNs types
- Structured data is like the databases and tables.
- Unstructured data is like images, video, audio, and text.
- Structured data gives more money because companies relies on prediction on its big data.

Why is deep learning taking off?

- Deep learning is taking off for 3 reasons:

  i. Data:
  - Using this image we can conclude:



  - For small data NN can perform as Linear regression or SVM (Support vector machine)
  - For big data a small NN is better that SVM
  - For big data a big NN is better that a medium NN is better that small NN.
  - Hopefully we have a lot of data because the world is using the computer a little bit more
    - Mobiles
    - IOT (Internet of things)

  ii. Computation:
  - GPUs.
  - Powerful CPUs.
  - Distributed computing.
  - ASICs

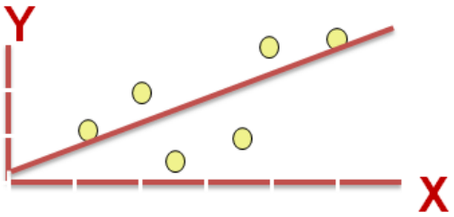
  iii. Algorithm:
  a. Creative algorithms has appeared that changed the way NN works.
    - For example using RELU function is so much better than using SIGMOID function in training a NN because it helps with the vanishing gradient problem.

Neural Networks Basics

> Learn to set up a machine learning problem with a neural network mindset. Learn to use vectorization to speed up your models.

Binary classification

- Mainly he is talking about how to do a logistic regression to make a binary classifier.



  - Image taken from 3.bp.blogspot.com
- He talked about an example of knowing if the current image contains a cat or not.
- Here are some notations:
  - `M is the number of training vectors`
  - `Nx is the size of the input vector`
  - `Ny is the size of the output vector`
  - `X(1) is the first input vector`
  - `Y(1) is the first output vector`
  - `X = [x(1) x(2).. x(M)]`
  - `Y = (y(1) y(2).. y(M))`
- We will use python in this course.
- In NumPy we can make matrices and make operations on them in a fast and reliable time.

Logistic regression

- Algorithm is used for classification algorithm of 2 classes.
- Equations:
  - Simple equation: `y = wx + b`
  - If x is a vector: `y = w(transpose)x + b`
  - If we need y to be in between 0 and 1 (probability): `y = sigmoid(w(transpose)x + b)`
  - In some notations this might be used: `y = sigmoid(w(transpose)x)`
    - While `b` is `w0` of `w` and we add `x0 = 1`. but we won't use this notation in the course (Andrew said that the first notation is better).
- In binary classification `Y` has to be between `0` and `1`.
- In the last equation `w` is a vector of `Nx` and `b` is a real number

Logistic regression cost function

- First loss function would be the square root error: `L(y',y) = 1/2 (y' - y)^2`
  - But we won't use this notation because it leads us to optimization problem which is non convex, means it contains local optimum points.
- This is the function that we will use: `L(y',y) = - (y*log(y') + (1-y)*log(1-y'))`
- To explain the last function lets see:
  - if `y = 1` ==> `L(y',1) = -log(y')` ==> we want `y'` to be the largest ==> `y'` biggest value is 1
  - if `y = 0` ==> `L(y',0) = -log(1-y')` ==> we want `1-y'` to be the largest ==> `y'` to be smaller as possible because it can only has 1 value.
- Then the Cost function will be: `J(w,b) = (1/m) * Sum(L(y'[i],y[i]))`
- The loss function computes the error for a single training example; the cost function is the average of the loss functions of the entire training set.

Gradient Descent

- We want to predict `w` and `b` that minimize the cost function.
- Our cost function is convex.
- First we initialize `w` and `b` to 0,0 or initialize them to a random value in the convex function and then try to improve the values the reach minimum value.
- In Logistic regression people always use 0,0 instead of random.
- The gradient decent algorithm repeats: `w = w - alpha * dw` where alpha is the learning rate and `dw` is the derivative of `w` (Change to `w`) The derivative is also the slope of `w`
- Looks like greedy algorithms. the derivative give us the direction to improve our parameters.