Introduction to Deep Learning with TensorFlow™

Jian Tao

jtao@tamu.edu

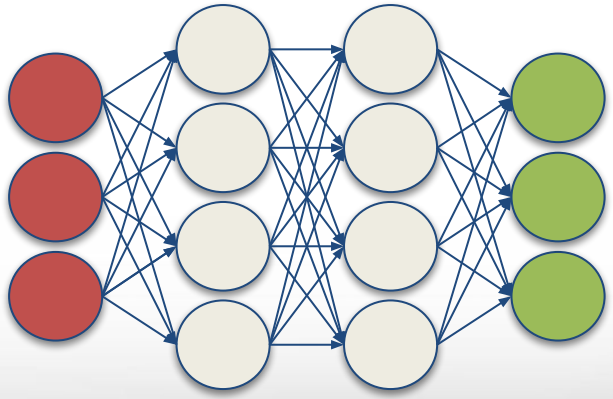
Spring 2018 HPRC Short Course

02/02/2018

Deep Learning

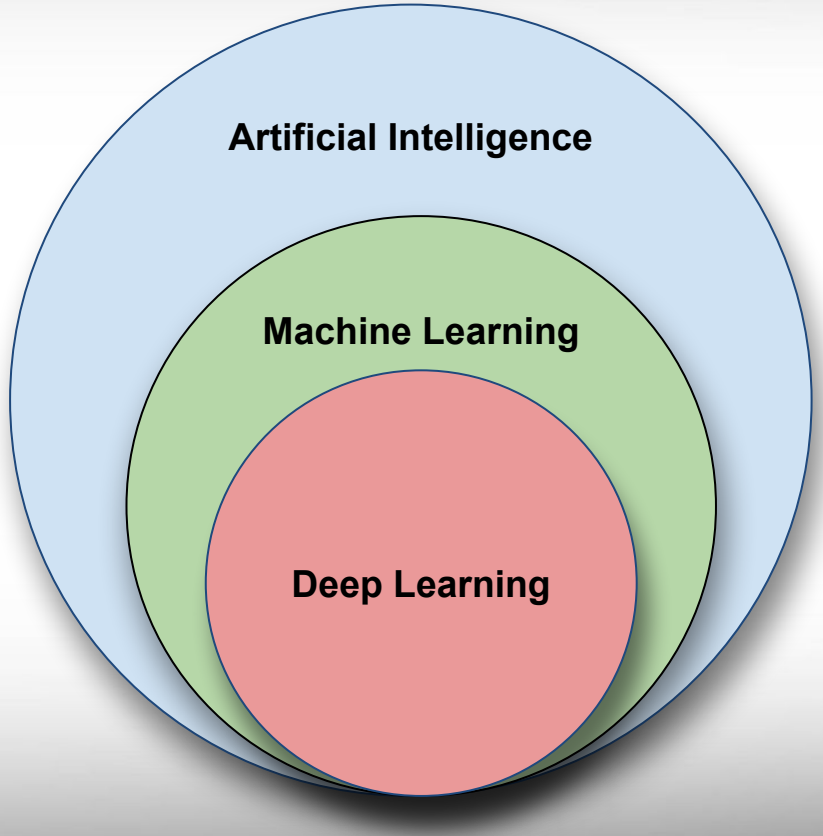
Deep Learning
by Ian Goodfellow, Yoshua Bengio, and Aaron Courville
http://www.deeplearningbook.org/

Animation of Neutron Networks
by Grant Sanderson
https://www.3blue1brown.com/

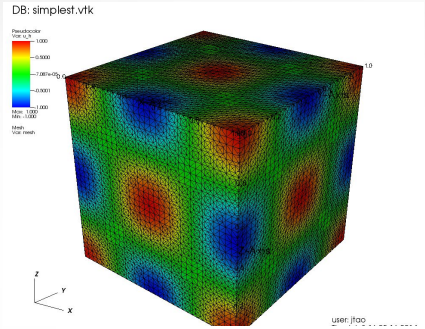


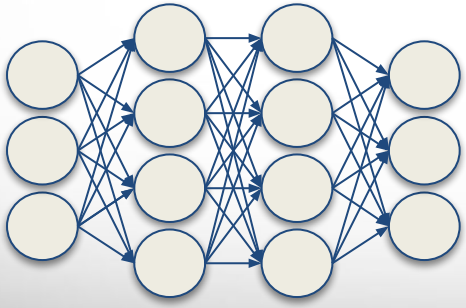
Relationship of AI, ML, and DL

- **Artificial Intelligence (AI)** is anything about man-made intelligence exhibited by machines.
- **Machine Learning (ML)** is an approach to achieve **AI**.
- **Deep Learning (DL)** is one technique to implement **ML**.

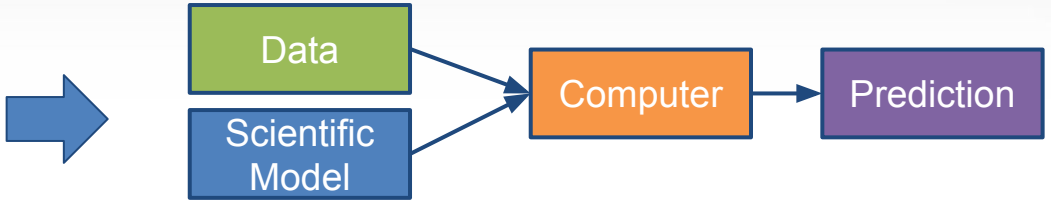


Machine Learning

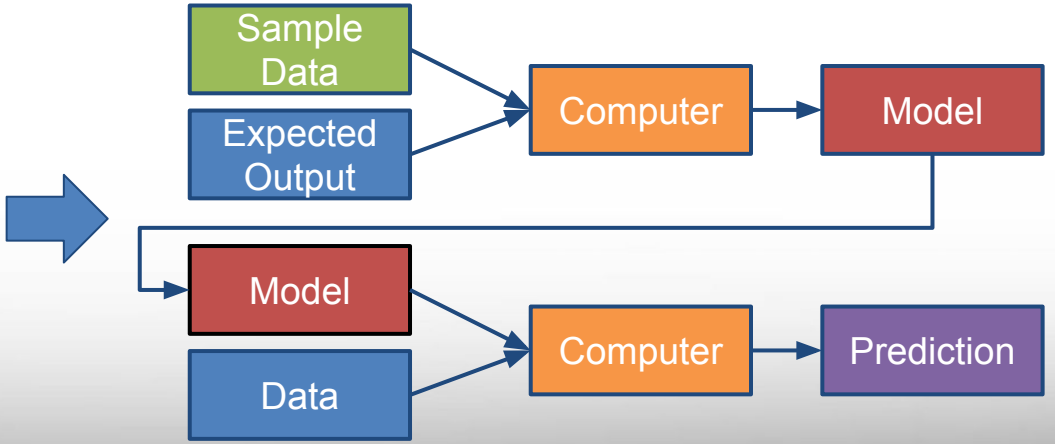




Traditional Modeling

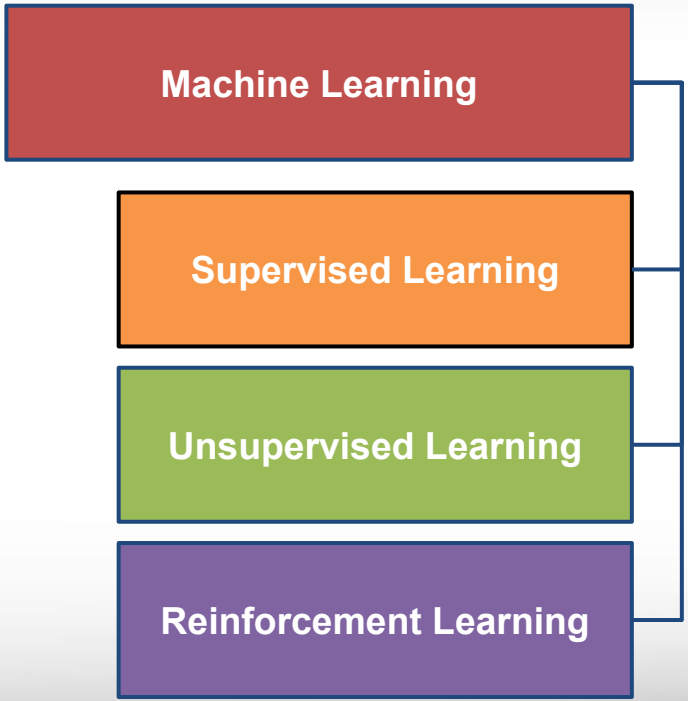


Machine Learning (Supervised Learning)



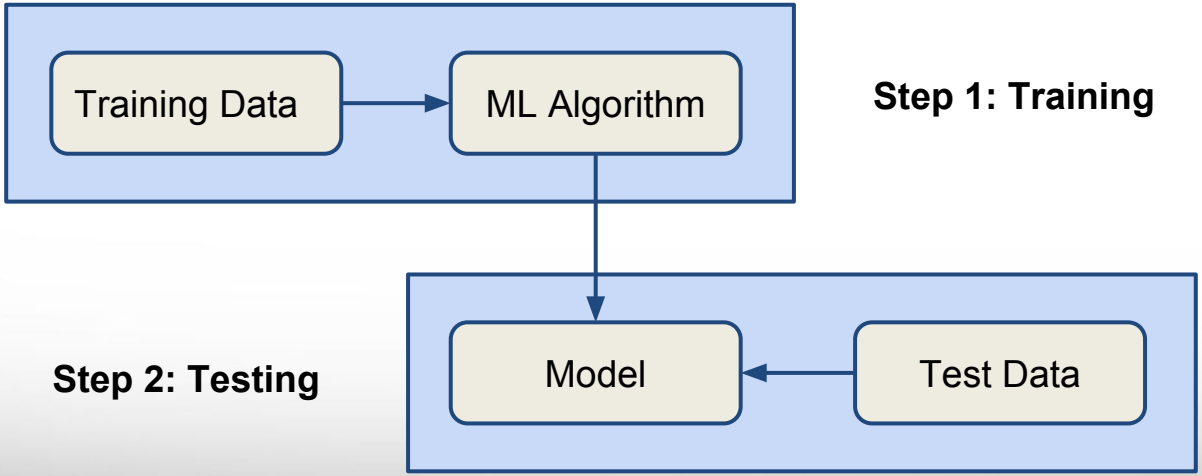
Types of ML Algorithms

- **Supervised Learning**
  - trained with labeled data; including regression and classification problems
- **Unsupervised Learning**
  - trained with unlabeled data; clustering and association rule learning problems.
- **Reinforcement Learning**
  - no training data; stochastic Markov decision process; robotics and self-driving cars.



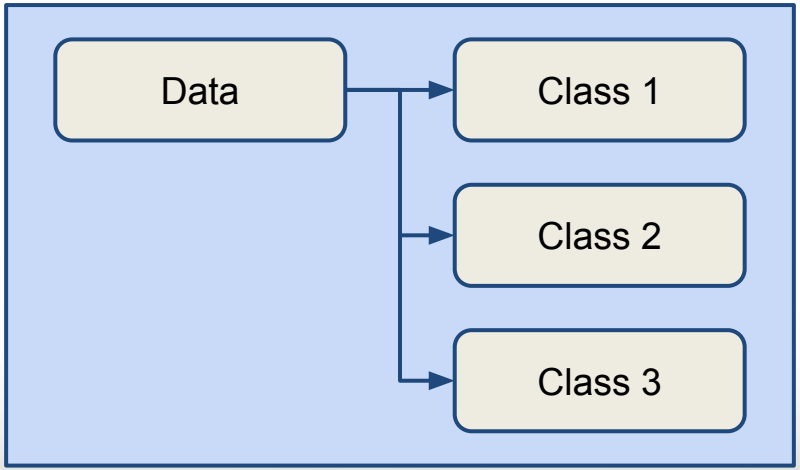
Supervised Learning

When both input variables - X and output variables - Y are known, one can approximate the mapping function from X to Y.



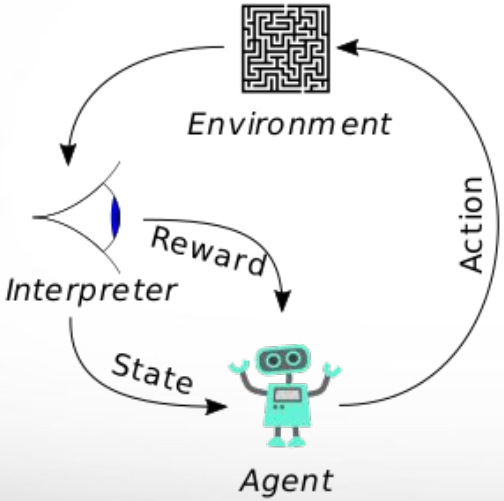
Unsupervised Learning

When only input variables - X are known and the training data is neither classified nor labeled. It is usually used for clustering problems.

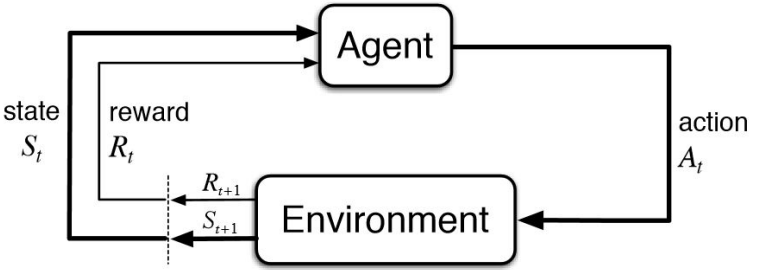


Reinforcement Learning

When the input variables are only available via interacting with the environment, reinforcement learning can be used to train an "**agent**".



(Image Credit: Wikipedia.org)



(Image Credit: deeplearning4j.org)

Why Deep Learning?

- Limitations of traditional machine learning algorithms
  - not good at handling high dimensional data.
  - difficult to do feature extraction and object recognition.

- Advantages of deep learning
  - DL is computationally expensive, but it is capable of handling high dimensional data.
  - feature extraction is done automatically.

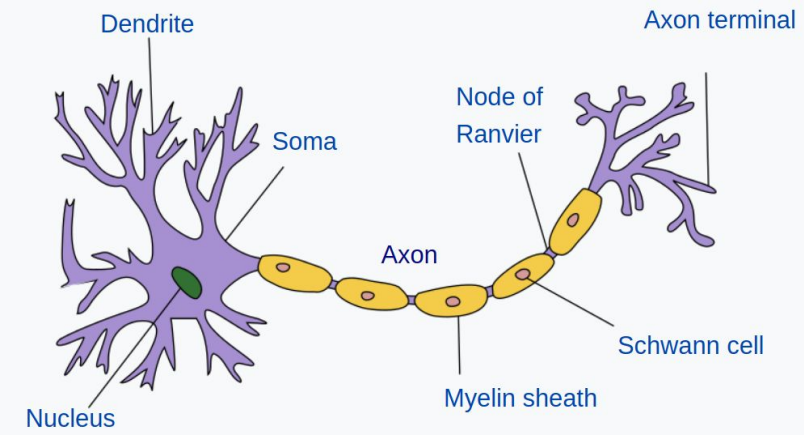
What is Deep Learning?

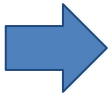
Deep learning is a class of machine learning algorithms that:

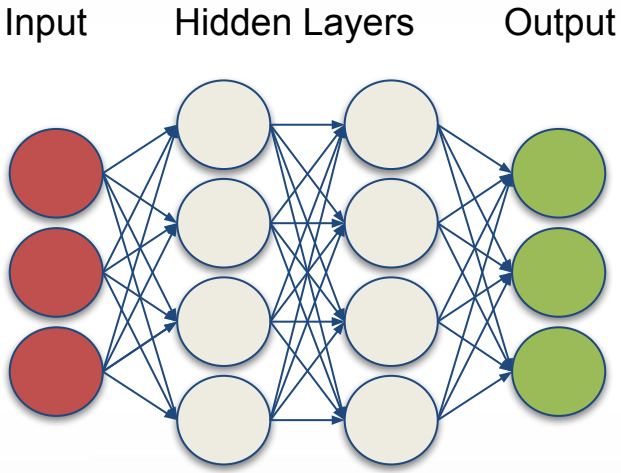
- use a cascade of multiple layers of nonlinear processing units for feature extraction and transformation. Each successive layer uses the output from the previous layer as input.
- learn in supervised (e.g., classification) and/or unsupervised (e.g., pattern analysis) manners.
- learn multiple levels of representations that correspond to different levels of abstraction; the levels form a hierarchy of concepts.

(Source: Wikipedia)

Artificial Neural Network







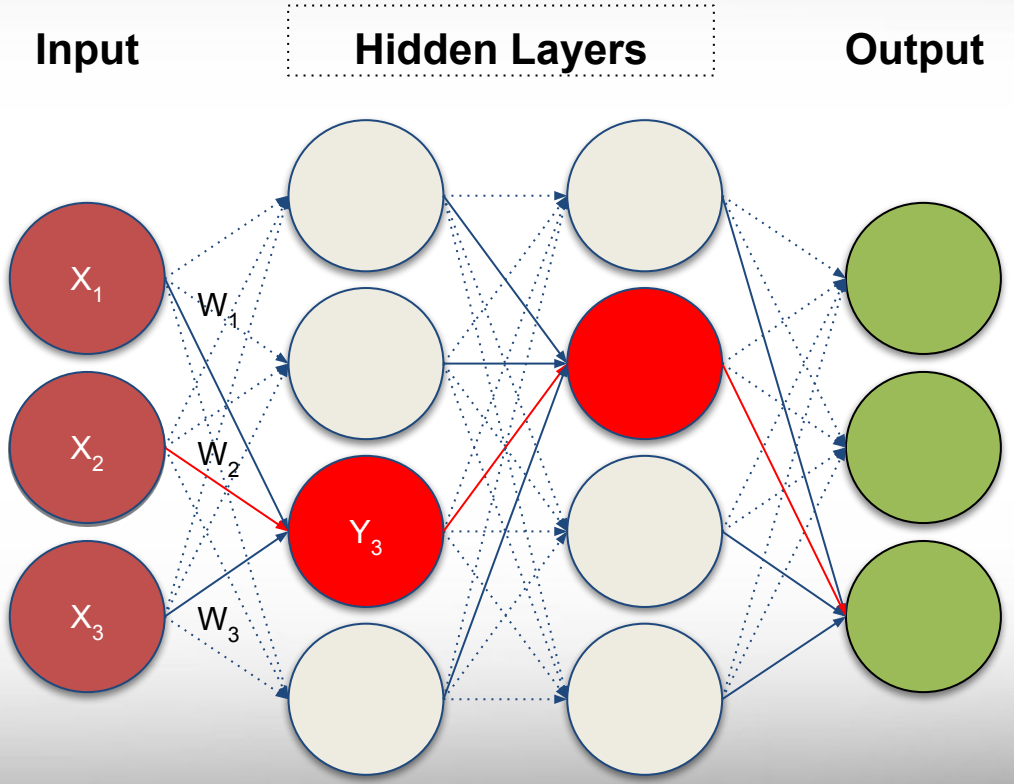
(Image Credit: Wikipedia)

Supervised Deep Learning with Neural Networks

From one layer to the next

$$Y_j = f\left(\sum_i W_i X_i + b_i\right)$$

f is the activation function, W_i is the weight, and b_i is the bias.



Training - Minimizing the Loss

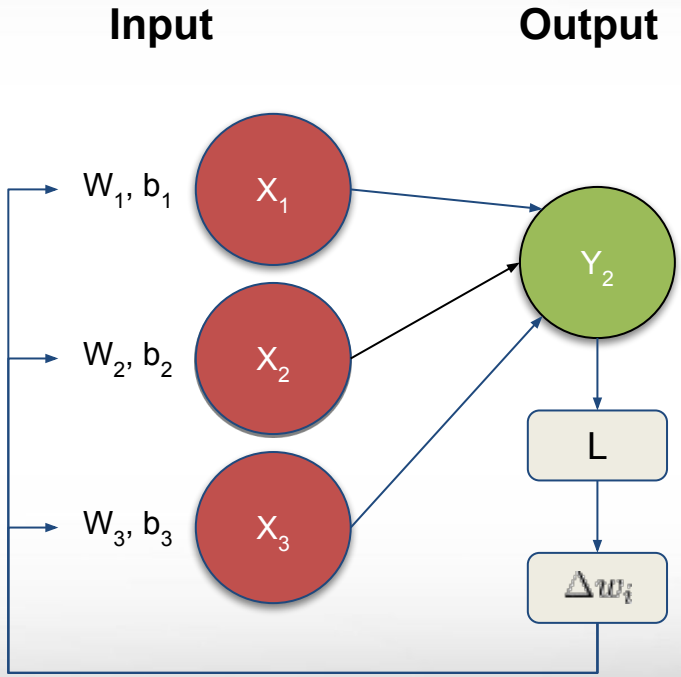
The loss function with regard to weights and biases can be defined as

$$L(\mathbf{w}, \mathbf{b}) = \frac{1}{2} \sum_i (\mathbf{Y}(\mathbf{X}, \mathbf{w}, \mathbf{b}) - \mathbf{Y}'(\mathbf{X}, \mathbf{w}, \mathbf{b}))^2$$

The weight update is computed by moving a step to the opposite direction of the cost gradient.

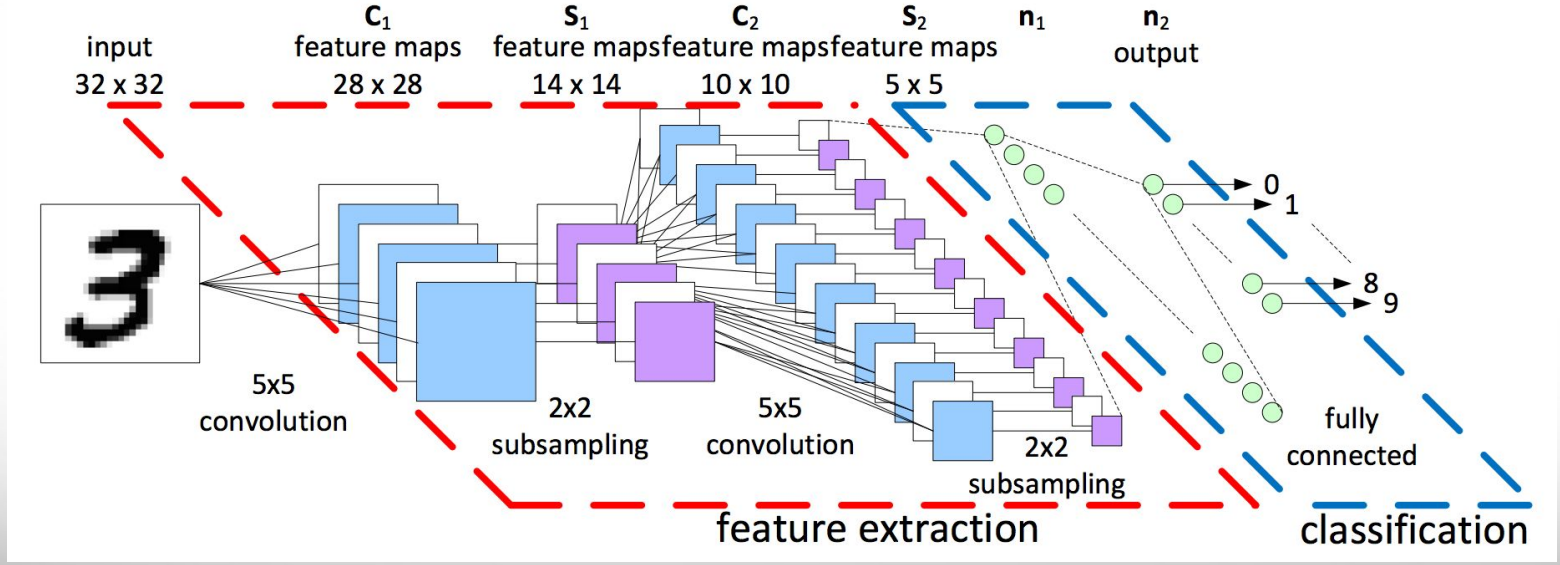
$$\Delta w_i = -\alpha \frac{\partial L}{\partial w_i}$$

Iterate until L stops decreasing.



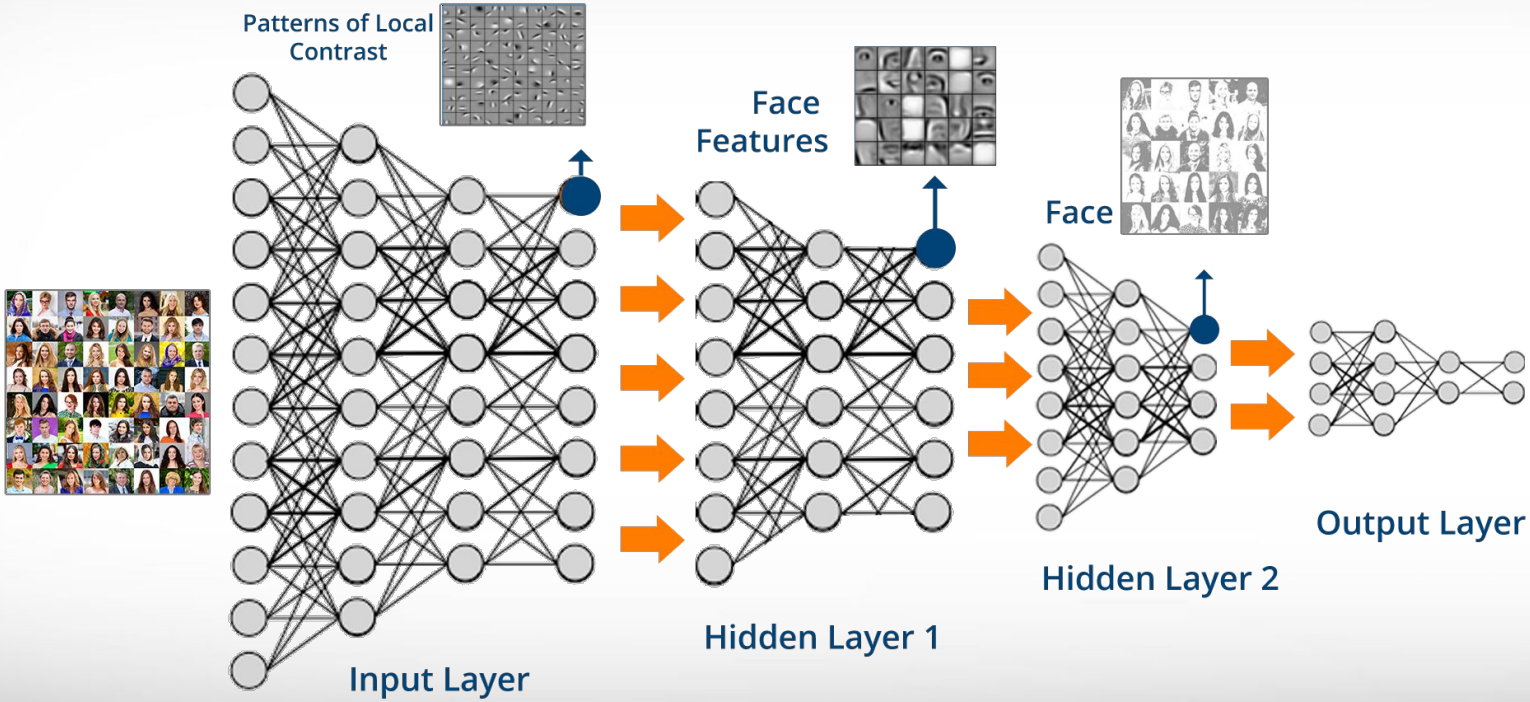
Convolutional Neural Networks

A convolutional neural network (**CNN**, or **ConvNet**) is a class of deep, feed-forward artificial neural networks that explicitly assumes that the inputs are images, which allows us to encode certain properties into the architecture.



(Image Credit: https://becominghuman.ai)

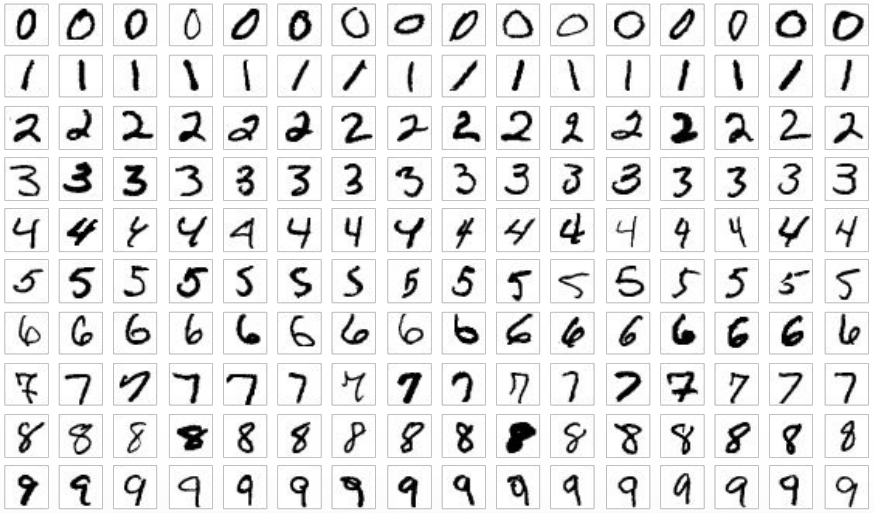
Deep Learning for Facial Recognition



(Image Credit: www.edureka.co)

MNIST - Introduction

- **MNIST** (Mixed National Institute of Standards and Technology) is a database for handwritten digits, distributed by Yann Lecun.
- 60,000 examples, and a test set of 10,000 examples.
- 28x28 pixels each.
- Widely used for research and educational purposes.

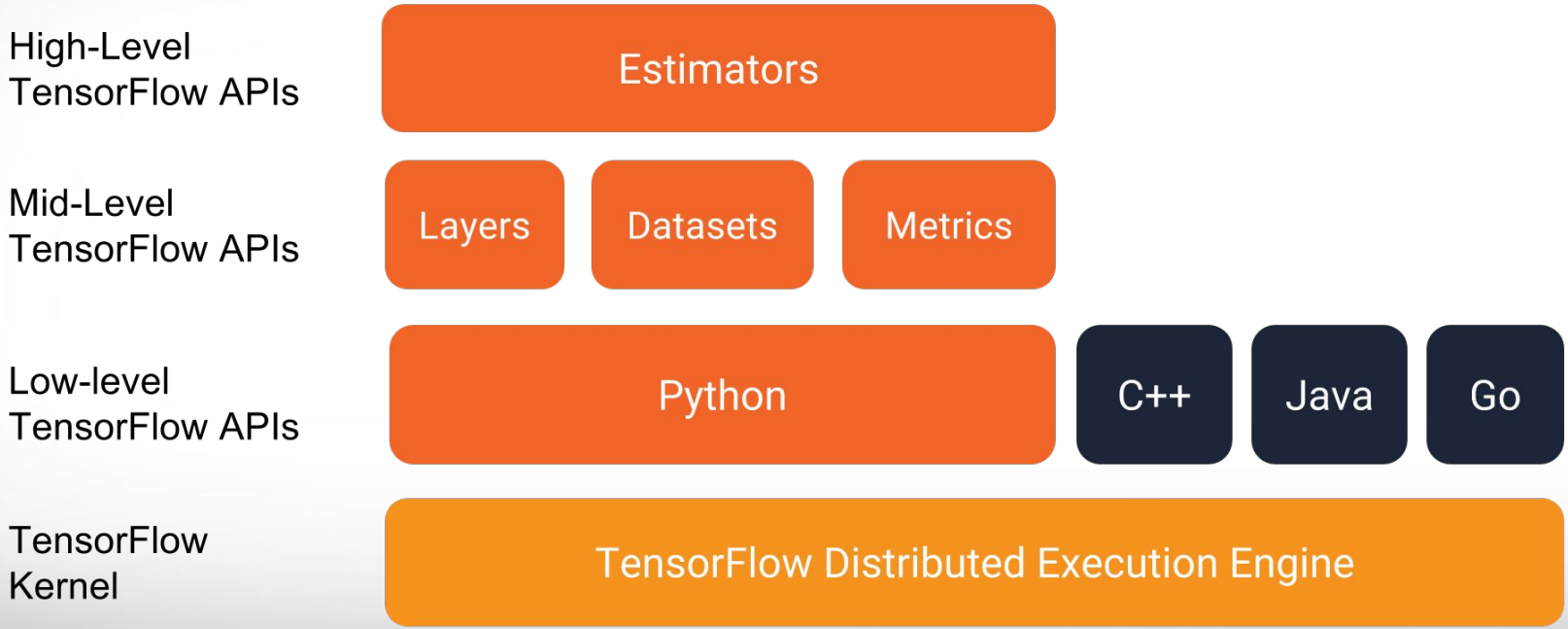


(Image Credit: Wikipedia)

TensorFlow

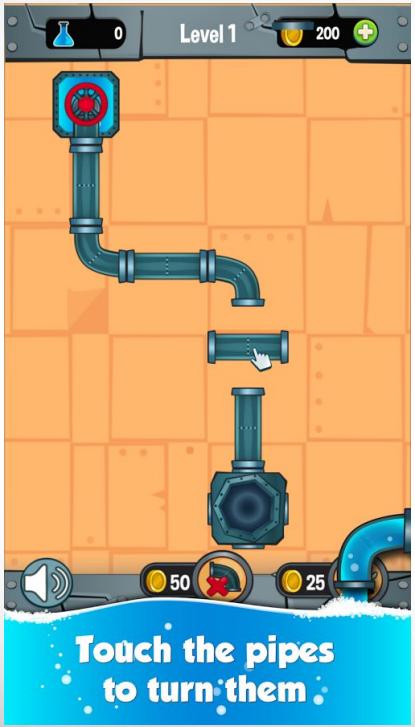
Tensorflow web site:
https://www.tensorflow.org/

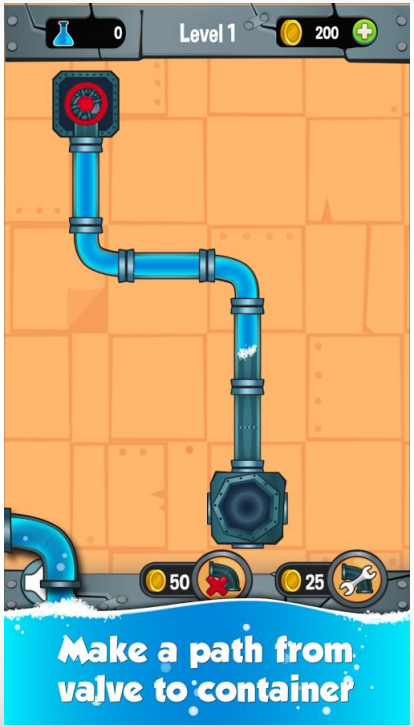
Programming Environment



(Image Credit: tensorflow.org)

A Connected Pipeline for the Flow of Tensors





(Image Credit: Plumber Game by Mobiloids)

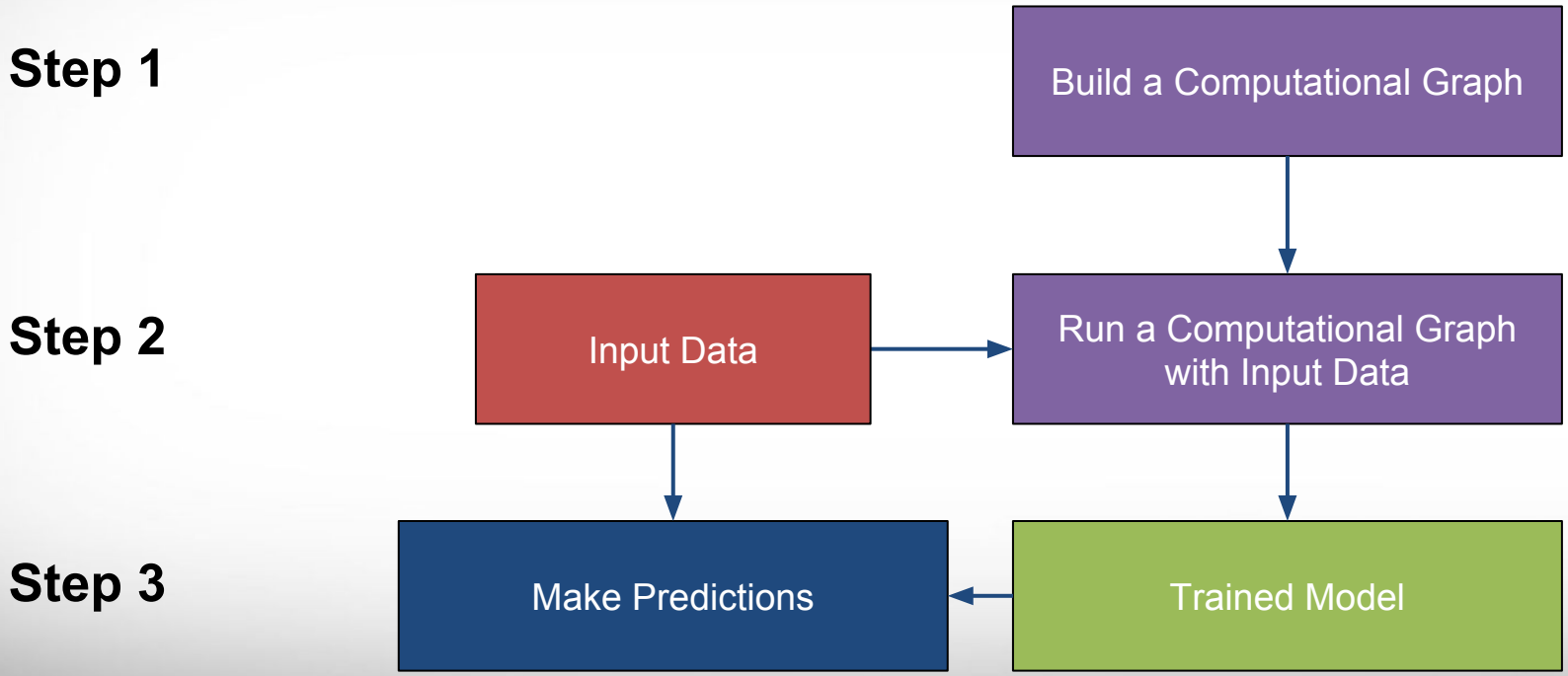
What is a Tensor in TensorFlow?

- **TensorFlow** uses a tensor data structure to represent all data. A TensorFlow tensor as an n-dimensional array or list. A tensor has a static type, a rank, and a shape.

Rank	Tensor
0	[5]
1	[1 2 3]
2	[[1 2 3 4], [5 6 7 8]]
3	...

TensorFlow Application

Eager Execution is now supported in TensofFlow 1.5



Hello World with TensorFlow

```
import tensorflow as tf

v = tf.constant("Hello World!")

sess = tf.Session()

print(sess.run(v))
```

Build a Graph

Run !

TensorFlow Constants

TensorFlow provides several operations to generate constant tensor.

```
import tensorflow as tf

x = tf.constant(1)
zeros = tf.zeros([2, 3], tf.int32)
ones = tf.ones([2, 3], tf.int32)
y = x *(zeros + ones + ones)

sess = tf.Session()
print(sess.run(y))
```

TensorFlow Placeholders

TensorFlow provides a placeholder operation that must be fed with data on execution. **Training/testing data sets** are usually stored in placeholders.

```
import tensorflow as tf
x = tf.placeholder(tf.float32, shape=(4, 4))
y = tf.matmul(x, x)

rand_array = np.random.rand(4, 4)
sess = tf.Session()
print(sess.run(y, {x: rand_array}))
```

TensorFlow Variables

TensorFlow variables can represent shared, persistent state manipulated by your program. **Weights** and **biases** are usually stored in variables.

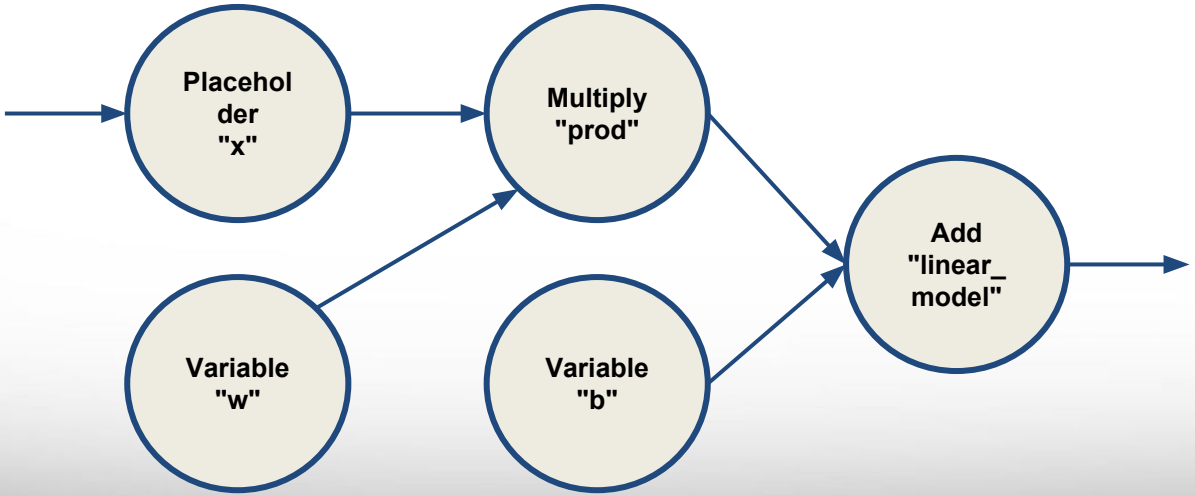
```
import tensorflow as tf

v = tf.Variable(tf.random_normal([5, 2], stddev=0.1))
init = tf.global_variables_initializer()

sess = tf.Session()
sess.run(init)
print(sess.run(v))
```

Build a Simple Linear Model Graph

```
x = tf.placeholder(tf.float32, name = 'x')
w = tf.Variable([.3], tf.float32, name = 'w')
b = tf.Variable([-.3], tf.float32, name = 'b')
linear_model = w * x + b
```



Loss of the Linear Model

TensorFlow provides functions to facilitate the calculation of standard loss functions.

```
squared_deltas = tf.square(linear_model - y)
loss = tf.reduce_sum(squared_deltas)

init = tf.global_variables_initializer()
sess=tf.Session()
sess.run(init)
print(sess.run(loss, {x: [1,2,3,4], y: [0,-1,-2,-3]}))
```

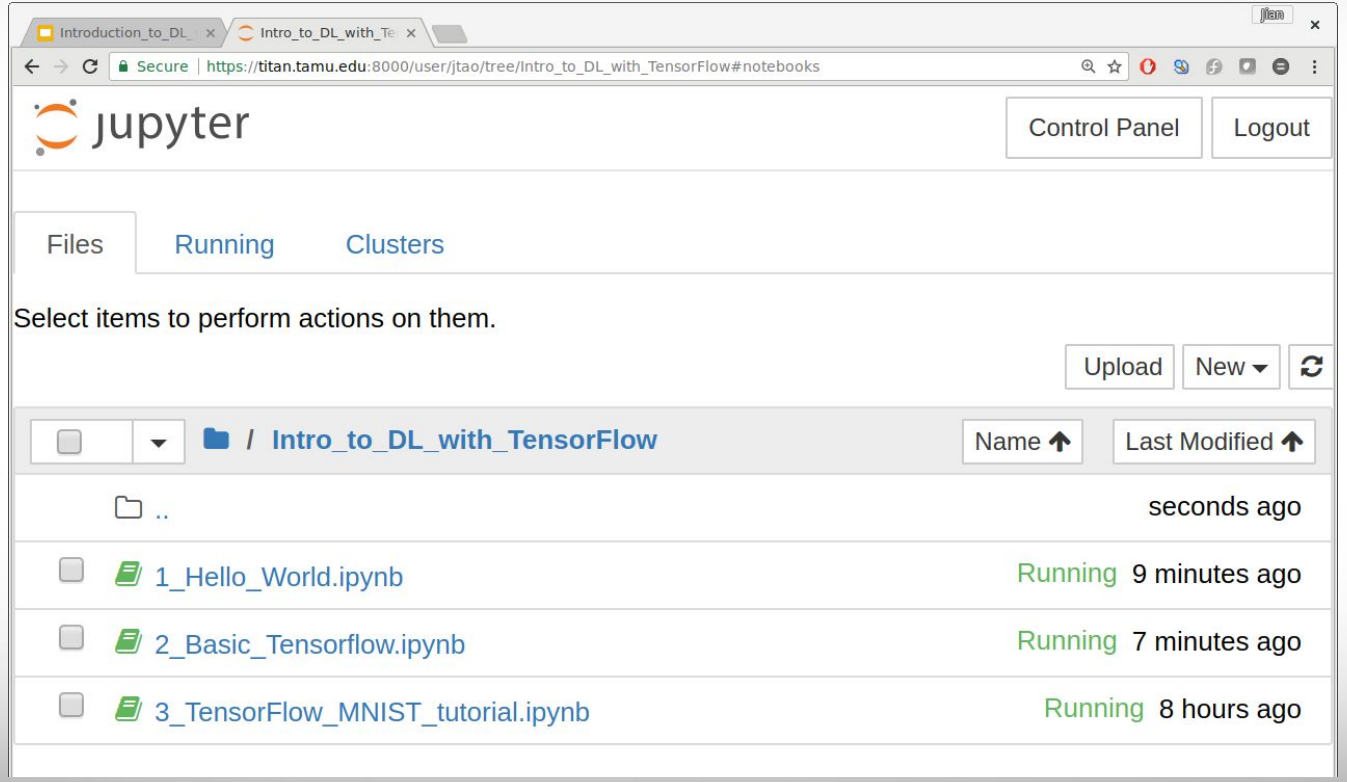
Train the Linear Model

TensorFlow Estimator APIs also provide optimizers to minimize the loss.

```
optimizer = tf.train.GradientDescentOptimizer(0.01)
train = optimizer.minimize(loss)

init = tf.global_variables_initializer()
sess=tf.Session()
sess.run(init)
for i in range(1000):
    sess.run(train, {x: [1,2,3,4], y: [0,-1,-2,-3]})
print(sess.run([w, b]))
```

Exercises on Jupyter Hub



Acknowledgements

- Supports from Texas A&M Engineering Experiment Station (TEES) and High Performance Research Computing (HPRC).