

# Review: From History to Practical Tips

Chuck

# The business

## NVIDIA INCEPTION — 1,300 DEEP LEARNING STARTUPS

### HEALTHCARE



### IOT & MANUFACTURING



### ADTECH, RETAIL, ETAIL



### AUTONOMOUS MACHINES



### FINANCIAL



### CYBER



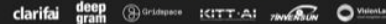
### SECURITY, IVA



### AEC\*



### PLATFORMS & APIs



### DATA MANAGEMENT



### DEVELOPMENT PLATFORMS

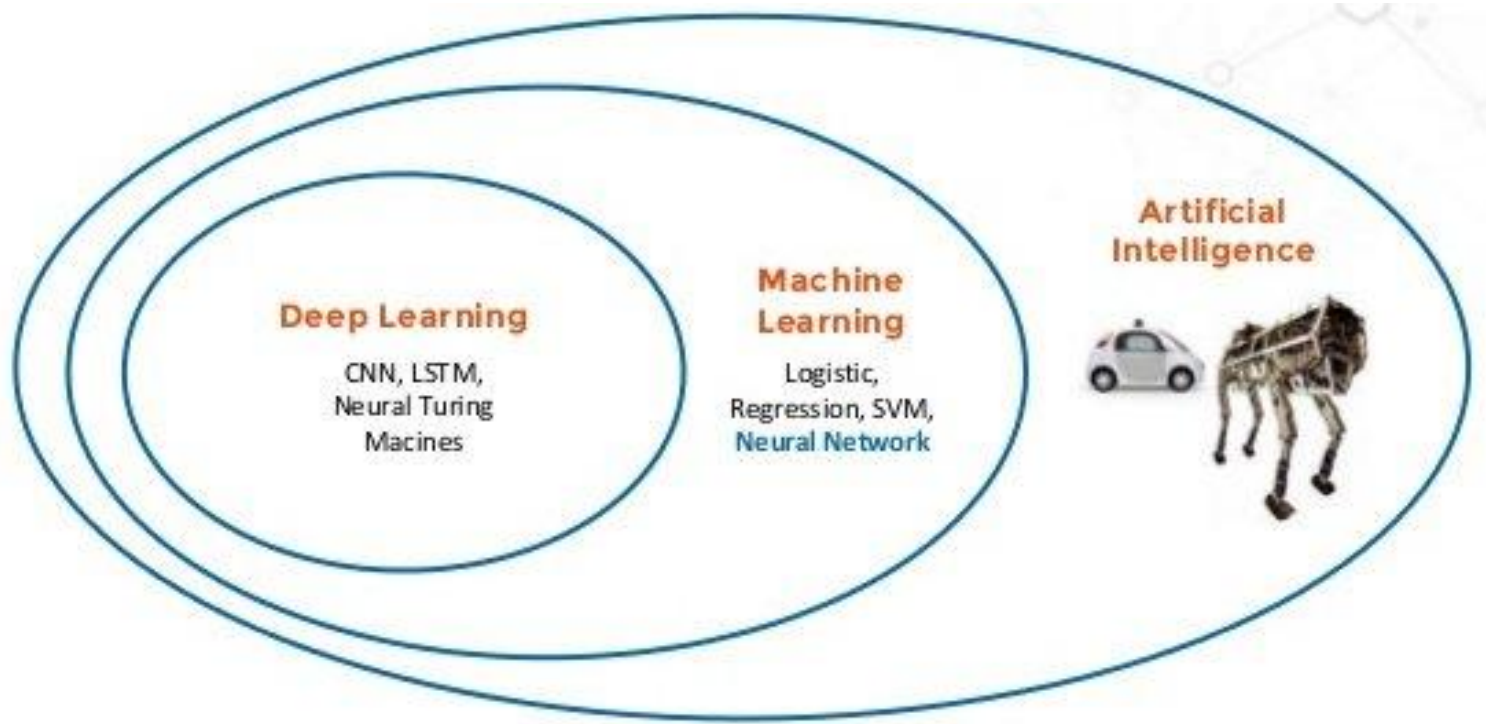


### BUSINESS INTELLIGENCE & VISUALIZATION (NON-DL)

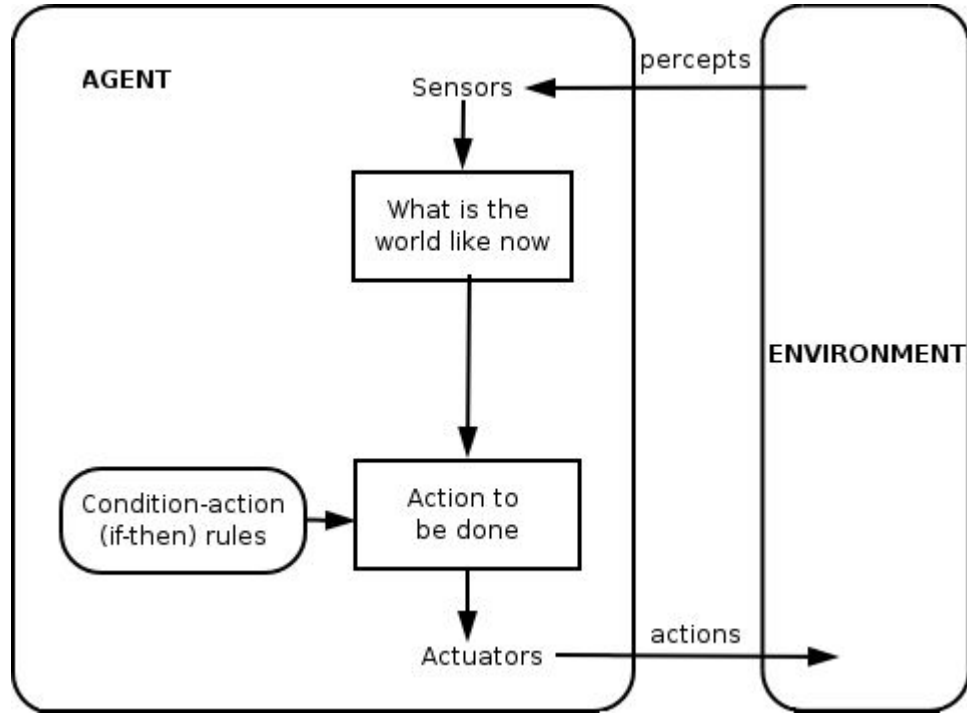


\* AEC = Architecture, Engineering, Construction

# Artificial Intelligence



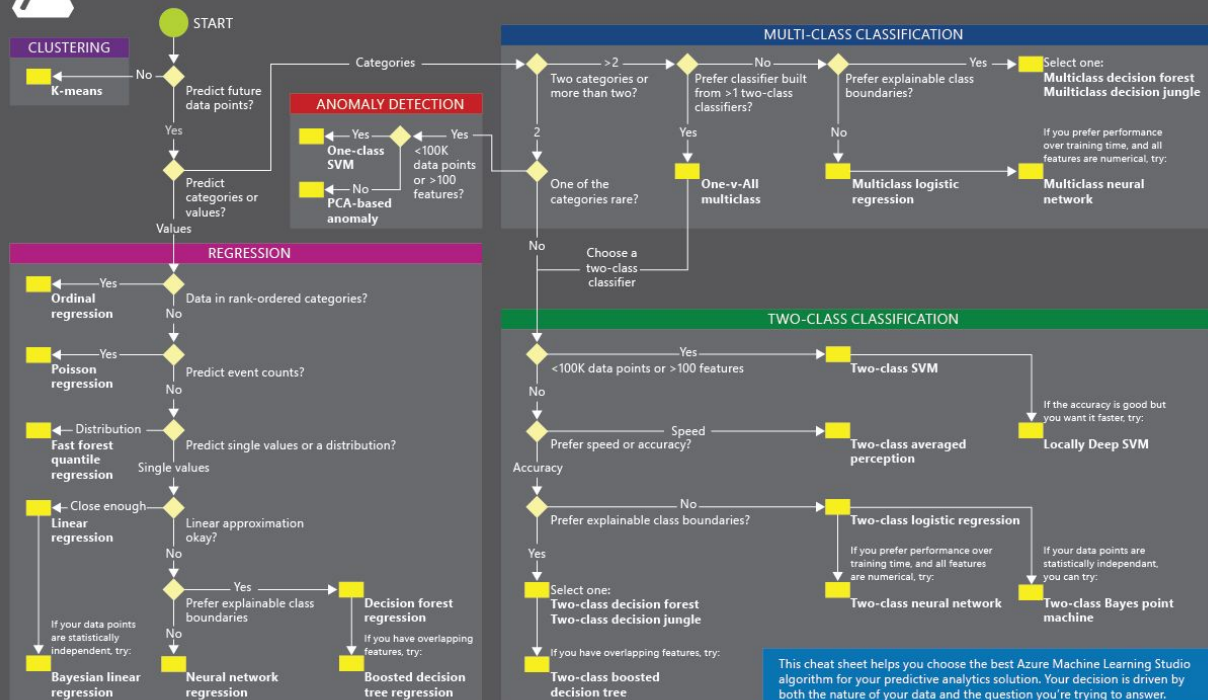
# What is “Intelligence”



# ML in one Page



## Microsoft Azure Machine Learning: Algorithm Cheat Sheet



# The History



1958 Perceptron

1974 Backpropagation



Convolution Neural Networks for  
Handwritten Recognition

1998



Google Brain Project on  
16k Cores

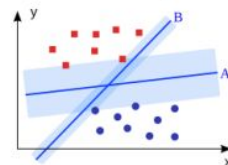
2012

1969  
Perceptron criticized



awkward silence (AI Winter)

1995  
SVM reigns



2006  
Restricted  
Boltzmann  
Machine



2012  
AlexNet wins  
ImageNet  
IMAGENET

# Basic DL Questions in 5 seconds

- What is the fastest mobile device?
- What is the fastest mobile device?
- What is the fastest mobile device?
- Can I run DL on a laptop?
- What is the fastest laptop?
- What is the fastest laptop?

Type	Name	Flops	Cost
Mobile	Raspberry Pi 1 <sup>st</sup> Gen, 700 Mhz	0,04 Gflops	\$35
Mobile	Apple A8	1,4 Gflops	\$700 (in iPhone 6)
CPU	Intel Core i7-4930K (Ivy Bridge), 3.7 GHz	140 Gflops	\$700
CPU	Intel Core i7-5960X (Haswell), 3.0 GHz	350 Gflops	\$1300
GPU	NVidia GTX 980	4612 Gflops (single precision), 144 Gflops (double precision)	\$600 + cost of PC (~\$1000)
GPU	NVidia Tesla K80	8740 Gflops (single precision), 2910 Gflops (double precision)	\$4500 + cost of PC (~1500)

ns that human encounter

# What is the minimum Linear Algebra you must know

1. Scalar, Vector, Matrices(2D), Narray (tensor)
2. Matrices Operation: Transpose, Addition & Subtraction (broadcasting), Multiplication

That is all you need!!

But it is good to handle:

3. Partial Derivative, Chain Rules, Jacobian, Determinant

Question: What is gradient vanishing? What cause gradient vanishing?



# Questions: Supervised Learning(1)

1. What is Linear Regression, Logistic Regression?
2. What is the formula of Gradient Descent (write it in 10s)?
3. How to solve Linear Regression and Logistic Regression by using GD and by closed form? (derive the formula)

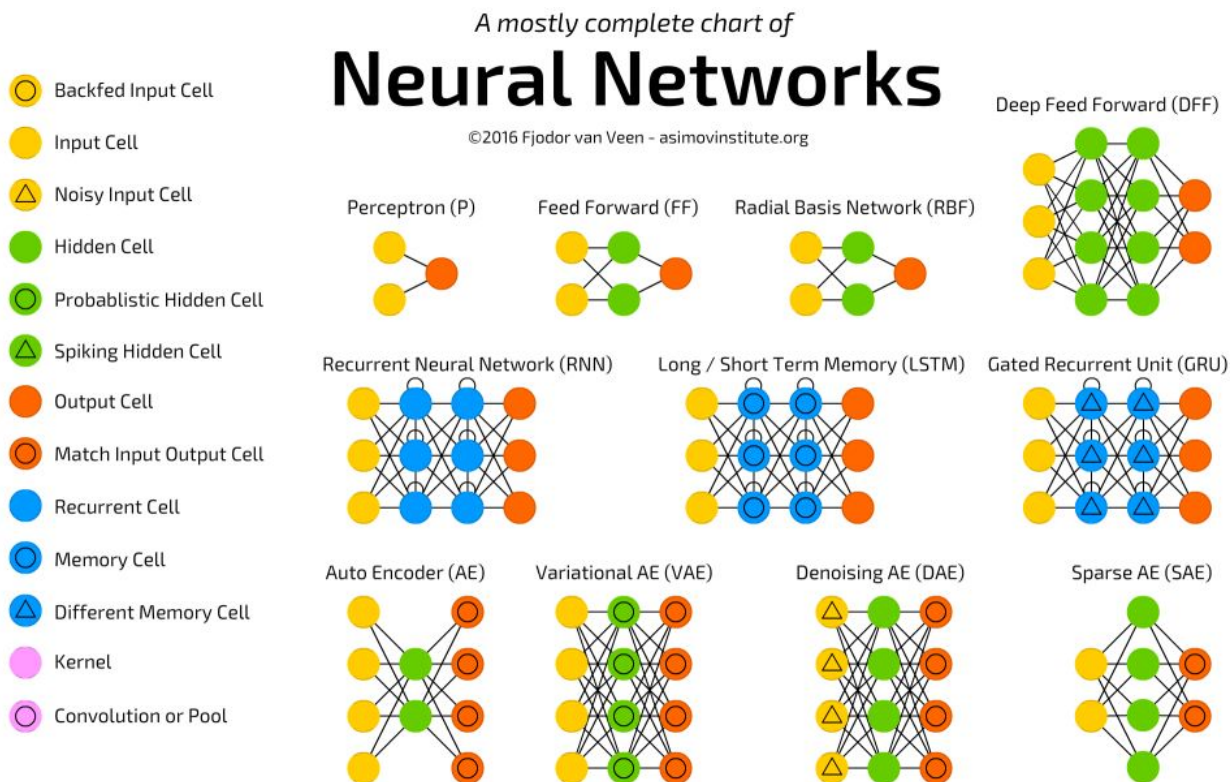
Even More

4. SVM, why it can max-margin. How come it did overshadow Neural Network for 20 years.
5. Decision Tree, KD Tree, Bayesian, Graph, Boltzman Machine, SNN, and many others

# Questions: Supervised Learning(2)

- 1.Hypothesis: Linear vs Non-linear
- 2.Cost Function design
- 3.Overfitting(Variance) & Underfitting(Bias), the methods to solve them
- 4.Precision & Recall
5. Local Minima

# Neural Network



# Deep Learning != Brain Learning

- The artificial neuron fires totally different than the brain
- A human brain has 100 billion neurons and 100 trillion connections (synapses) and operates on 20 watts(enough to run a dim light bulb) - in comparison the biggest neural network have 10 million neurons and 1 billion connections on 16,000 CPUs (about 3 million watts)
- The brain is limited to 5 types of input data from the 5 senses.
- Children do not learn what a cow is by reviewing 100,000 pictures labelled “cow” and “not cow”, but this is how machine learning works.
- Probably we don't learn by calculating the partial derivative of each neuron related to our initial concept. (By the way we don't know how we learn)

# Basic Questions: Deep Learning

1. Derive backpropagation (convolution, activation, cost function)
2. How to do vectorize and parallel
3. Gradient Checking
4. Weight Initialization
5. Training Steps
6. Training/Testing/Validation Data

Other knowledge points:

Layer types (list all the layer type for CNN architecture), Avoid Overfitting in DL (Dropout, Drop connection, Regularization, Data Augmentation),



# Optimization

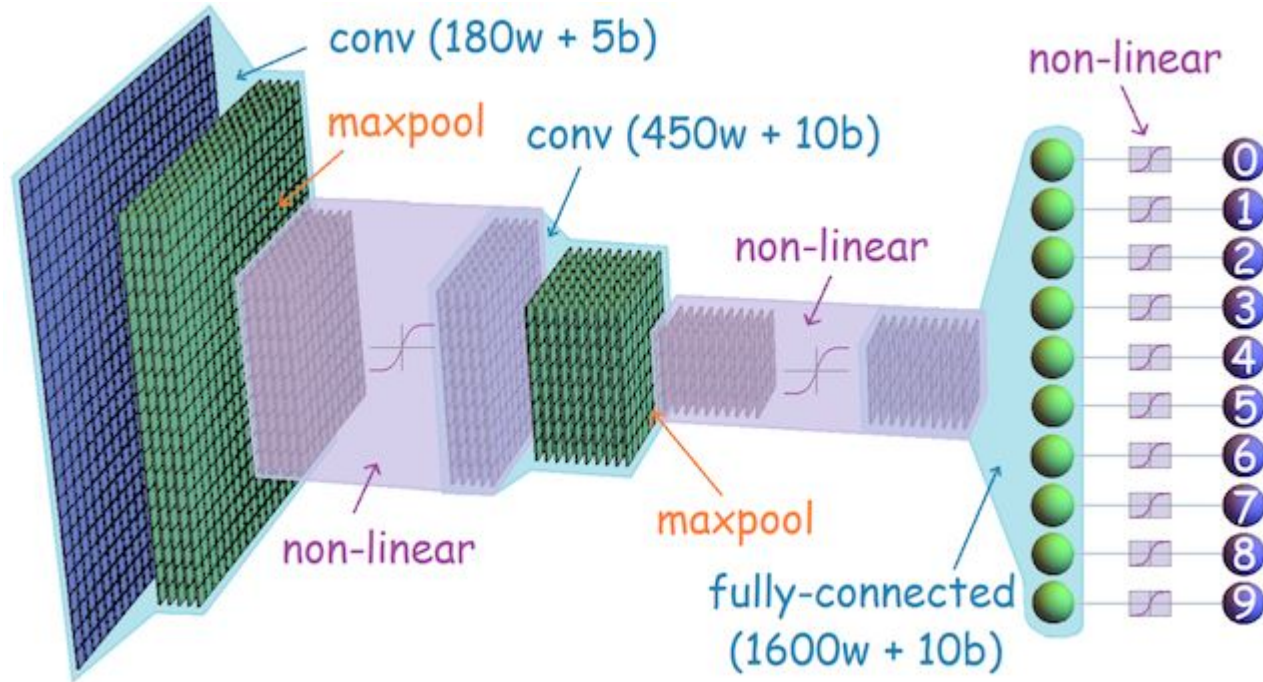
## All about the SGD:

- Stochastic Gradient Descent (SGD)
- Stochastic Gradient Descent with momentum (Very popular)
- Nestorov's accelerated gradient (NAG)
- Adaptive gradient (AdaGrad)
- Adam (Very good because you need to take less care about learning rate)
- RMSprop

## One interesting Argument:

An important thing to note is that learning rate scale proportionally with batch size so if we increase our batch size by 2x we can double our learning rate.

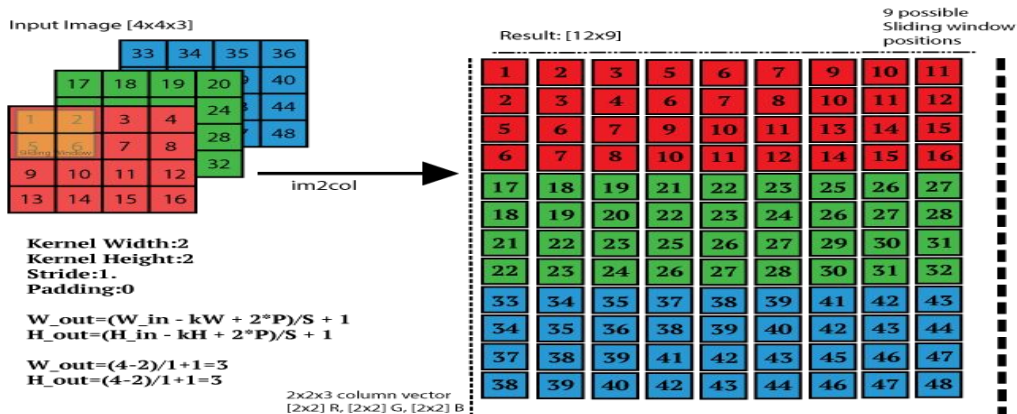
Just can't not share with you the graph (where is BN)



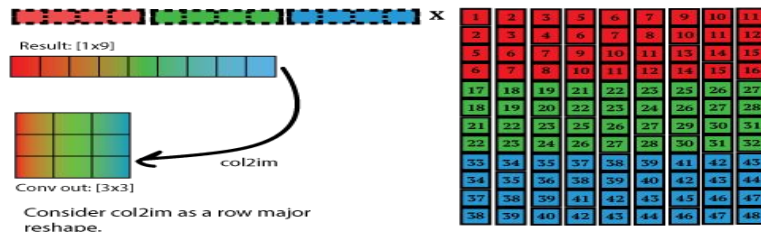
# Convolution: Make it fast! (im2col, col2im)

## Image to column operation (im2col)

Slide the input image like a convolution but each patch become a column vector.



We can multiply this result matrix [12x9] with a kernel [1x12].  
result = kernel x matrix  
The result would be a row vector [1x9].  
We need another operation that will convert this row vector into a image [3x3].



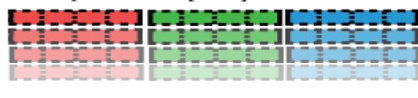
We get true performance gain

when the kernel has a large number of filters, ie:  $F=4$

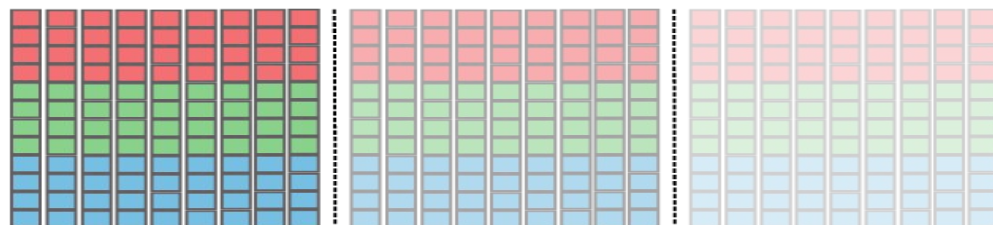
and/or you have a batch of images ( $N=4$ ). Example for the input batch [4x4x3x4], convolved with 4 filters [2x2x3x2].

The only problem with this approach is the amount of memory

Reshaped kernel: [4x12]



Converted input batch [12x56]





# MXNET v.s. Tensorflow

## SUMMARY

- Lots of momentum and support behind TensorFlow
  - TensorFlow has better RNN capabilities
  - TensorFlow has better tutorials and online guides. It also has more supporting material like Stack Overflow questions etc.
- TensorFlow enjoys greater support on the cloud, and has more deployment options
- MxNet has more language bindings, and is usually faster

MXNET v.s. Tensorflow

# Data Format

Data Format: **.rec** v.s. **.tfrecord**

.rec:

[https://github.com/BitTiger-MP/DS502-AI-Engineer/blob/master/DS502-1702/MXNET\\_course/mxnet-week4n5-final-project/data\\_ulti.py](https://github.com/BitTiger-MP/DS502-AI-Engineer/blob/master/DS502-1702/MXNET_course/mxnet-week4n5-final-project/data_ulti.py)

.tfrecord:

[http://www.machinelearningguru.com/deep\\_learning/tensorflow/basics/tfrecord/tfrecord.html](http://www.machinelearningguru.com/deep_learning/tensorflow/basics/tfrecord/tfrecord.html)

# Symbol Build

MXNET:

[https://github.com/BitTiger-MP/DS502-AI-Engineer/blob/master/DS502-1702/MXNET\\_course/mxnet-week3/cifar10/symbols/resnet.py](https://github.com/BitTiger-MP/DS502-AI-Engineer/blob/master/DS502-1702/MXNET_course/mxnet-week3/cifar10/symbols/resnet.py)

Tensorflow: <https://github.com/tensorflow/models/blob/master/tutorials/image/cifar10/cifar10.py#L188-#L295>

# Logging System

MXNET: <https://github.com/dmlc/tensorboard>

Tensorflow: [https://www.tensorflow.org/get\\_started/summaries\\_and\\_tensorboard](https://www.tensorflow.org/get_started/summaries_and_tensorboard)

# Training Strategy

MXNET:

[https://github.com/BitTiger-MP/DS502-AI-Engineer/blob/master/DS502-1702/MXNET\\_course/mxnet-week4n5-final-project/run\\_train.py#L68-L95](https://github.com/BitTiger-MP/DS502-AI-Engineer/blob/master/DS502-1702/MXNET_course/mxnet-week4n5-final-project/run_train.py#L68-L95)

Tensorflow:

<https://github.com/tensorflow/models/blob/master/tutorials/image/cifar10/cifar10.py#L325-L378>

# Trigger the Training

MXNET:

[https://github.com/BitTiger-MP/DS502-AI-Engineer/blob/master/DS502-1702/MXNET\\_course/mxnet-week4n5-final-project/run\\_train.py#L68-L95](https://github.com/BitTiger-MP/DS502-AI-Engineer/blob/master/DS502-1702/MXNET_course/mxnet-week4n5-final-project/run_train.py#L68-L95)

Tensorflow:

[https://github.com/tensorflow/models/blob/master/tutorials/image/cifar10/cifar10\\_train.py](https://github.com/tensorflow/models/blob/master/tutorials/image/cifar10/cifar10_train.py)

# Multi-GPU Training

MXNET:

[https://github.com/BitTiger-MP/DS502-AI-Engineer/blob/master/DS502-1702/MXNET\\_course/mxnet-week4n5-final-project/run\\_train.py#L26](https://github.com/BitTiger-MP/DS502-AI-Engineer/blob/master/DS502-1702/MXNET_course/mxnet-week4n5-final-project/run_train.py#L26)

Tensorflow:

[https://github.com/tensorflow/models/blob/master/tutorials/image/cifar10/cifar10\\_multi\\_gpu\\_train.py](https://github.com/tensorflow/models/blob/master/tutorials/image/cifar10/cifar10_multi_gpu_train.py)



# Distributed Training

MXNET:

[https://github.com/apache/incubator-mxnet/blob/master/docs/how\\_to/cloud.md](https://github.com/apache/incubator-mxnet/blob/master/docs/how_to/cloud.md)

Tensorflow:

<https://www.tensorflow.org/deploy/distributed>

<https://aws.amazon.com/it/blogs/compute/distributed-deep-learning-made-easy/>

# Summary

## MXNET

1. Focuses on DL only.
2. Faster, Lighter, and is maintained and optimized by AWS officially.
3. Multi-language supporting
4. Very friendly to research and engineering

## Tensorflow

1. Not only for DL, but for all ML (aggressive)
2. Heavy, and only one session one GPU
3. Only Python and C++
4. Big Community, not friendly(?) to engineering

# Let's deliver a DL product!

<https://blogs.dropbox.com/tech/2017/04/creating-a-modern-ocr-pipeline-using-computer-vision-and-deep-learning/>

Team:

PM/Hire M, SDE, Machine Learning Engineer, DevOps (5~15 people)

Time: 6~12 month

Data: Cost of Data collection, Legal (for big company), annotation

Hardware Support: AWS or Data Center

# Let's read a chapter in DL book

<http://www.deeplearningbook.org/contents/guidelines.html>

Thank You for your commitment

Lifelong Learning is lifelong joy. Enjoy it!