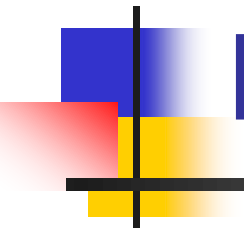
Deep Learning Using R & TensorFlow



Dr. Ash Pahwa

OC R User's Group

March 26, 2019

April 30, 2019

Bio: Dr. Ash Pahwa

- Ph.D. Computer Science
- Website: www.AshPahwa.com
- Affiliation
  - California Institute of Technology, Pasadena
  - UC Irvine
  - UCLA
  - UCSD
  - Chapman University: Adjunct
- Field of Expertise
  - Machine Learning, Deep Learning, Digital Image Processing, Database Management, CD-ROM/DVD
- Worked for
  - General Electric, AT&T Bell Laboratories, Oracle, UC Santa Barbara

Caltech Course

Deep Learning with TensorFlow

SCHEDULE

Classes are held Saturdays, 8:00 AM to 5:00 PM, on the Caltech campus in Pasadena, California.

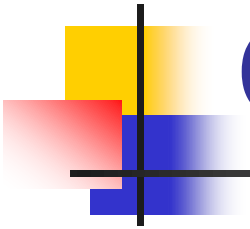
	Spring 2019	Add to Cart
Deep Learning with TensorFlow	April 6, 13, 20	Register

INSTRUCTOR

Ash Pahwa, PhD

Outline of the Course

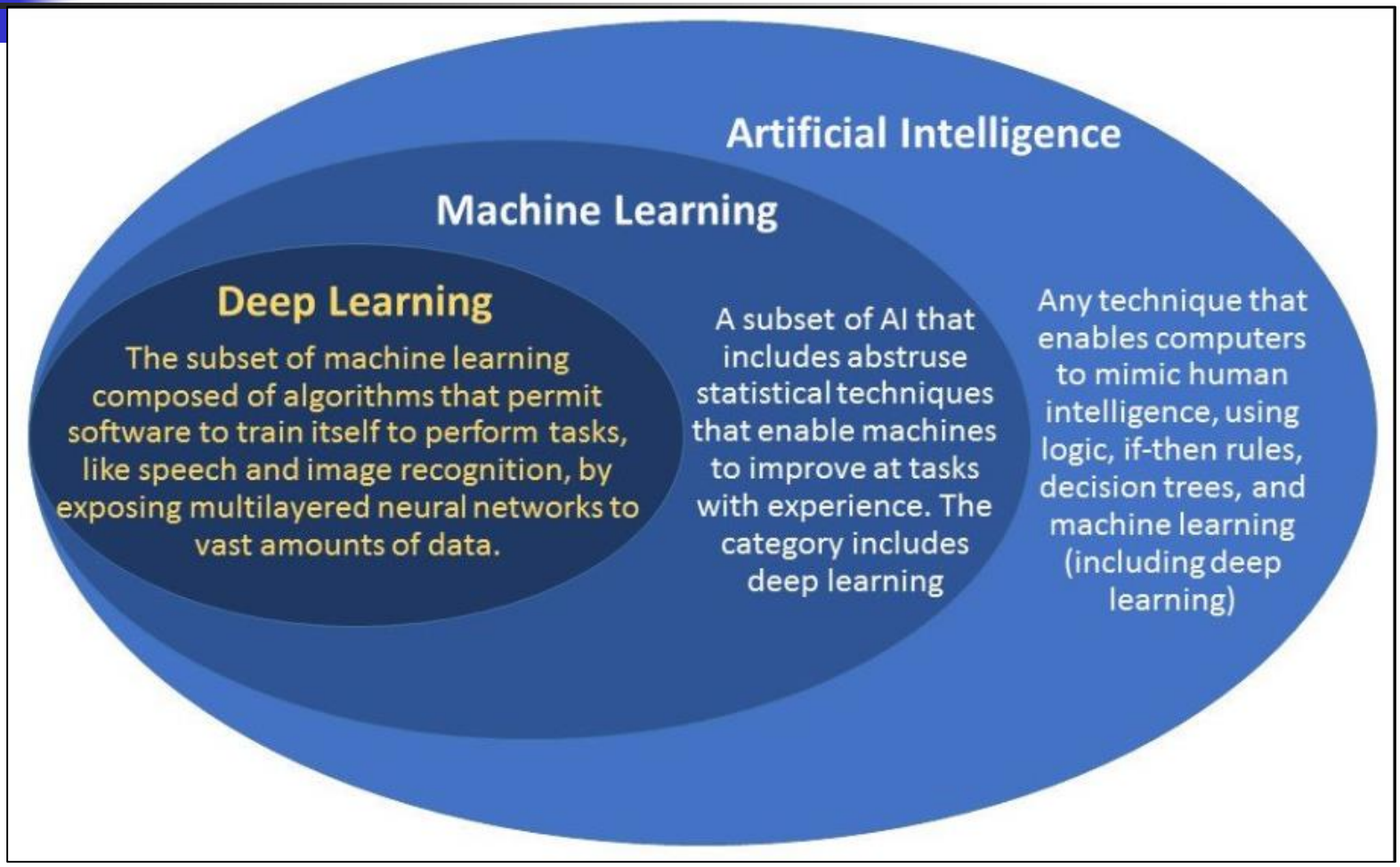
	Date	Lesson Content	Lesson Content
1	April 6, 2019	• Machine Learning and Deep Learning • TensorFlow Architecture	• Neural Network and NN Math • Building Neural Networks in TensorFlow
2	April 13, 2019	• Linear Regression in TensorFlow • Logistic Regression in TensorFlow • kNN in TensorFlow	• Gradient Descent Algorithm • Optimization Techniques: Adam • Backpropagation algorithm
3	April 20, 2019	• Convolutional Neural Networks • Convolution and Pooling • Building CNN in TF	• Recurrent Neural Networks • Building RNN in TF • Reinforcement Learning



Outline

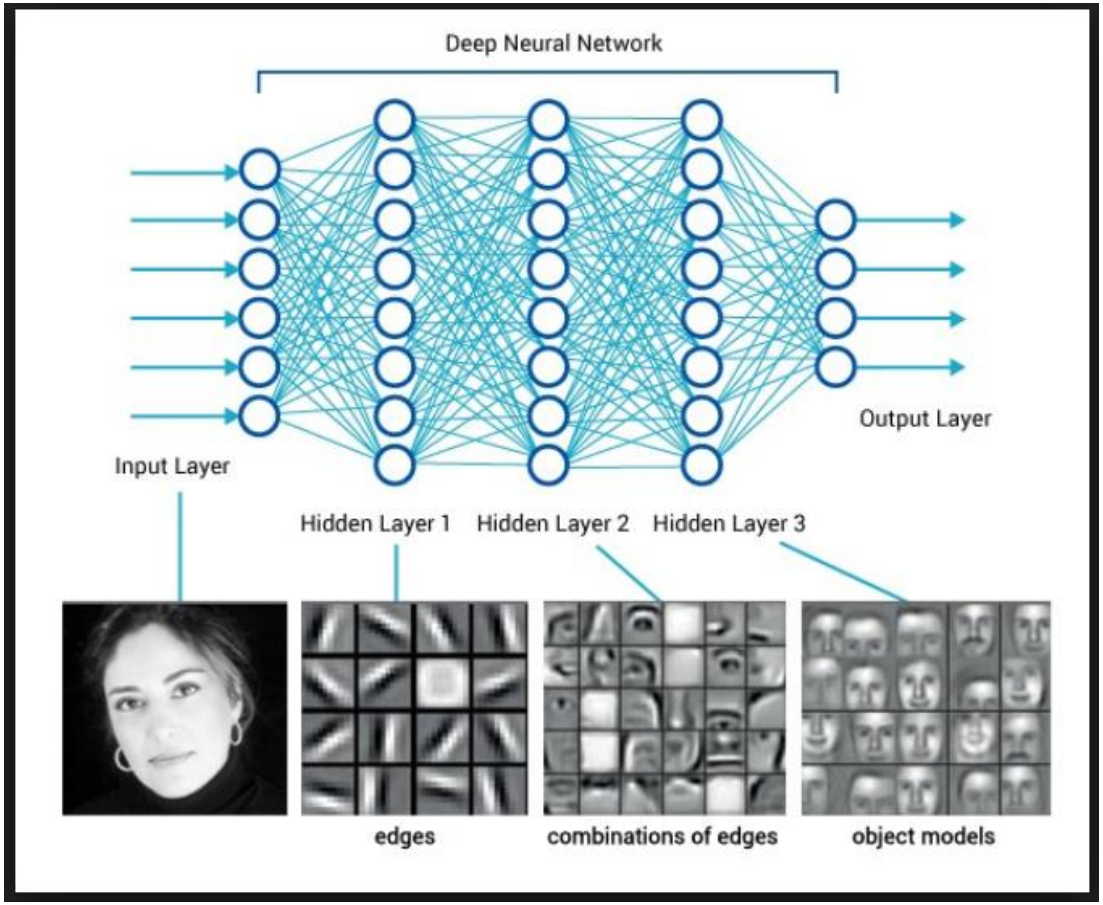
- What is Deep Learning
- Deep Learning Applications
- Tools for Deep Learning
- Misunderstanding about TensorFlow
- Availability of GPU
- TensorFlow Architecture
- R Code with TensorFlow

Artificial Intelligence



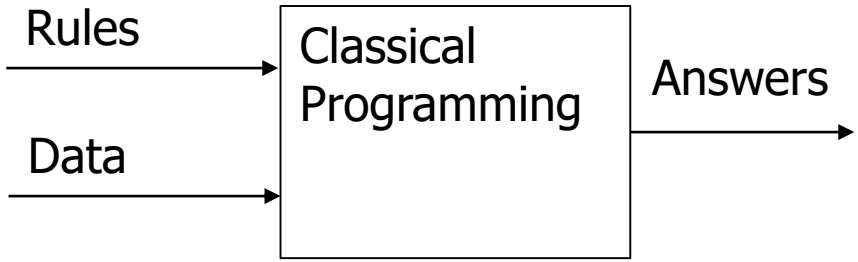
Deep Neural Network

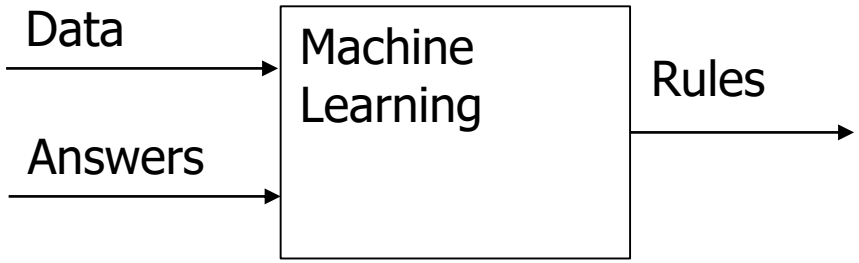
- Every layer of the DNN allows a more sophisticated build-up
  - From simple elements
  - To more complex ones



Problems that can use Neural Networks

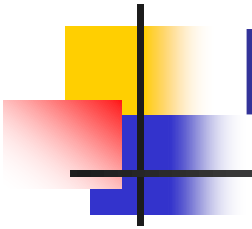
- For simple problems we can define the rules
  - We can automate the process
  - Write software
- For complex problems
  - We cannot define the rules
  - Object recognition in an image
- To solve these types of problems
  - We provide data and the answers
  - System will create the rules





Main Applications of Deep Learning Neural Networks

- Image Recognition
  - Convolution Neural Networks
- Image Classification
  - Convolution Neural Networks
- Hand Writing Identification
- Speech Recognition
  - Long Short-Term Memory Networks



Deep Learning Applications

- Convolutional Neural Networks (CNN)
  - Autonomous Driving
- Recurrent Neural Networks (RNN)
  - Language Translation
  - Image Caption

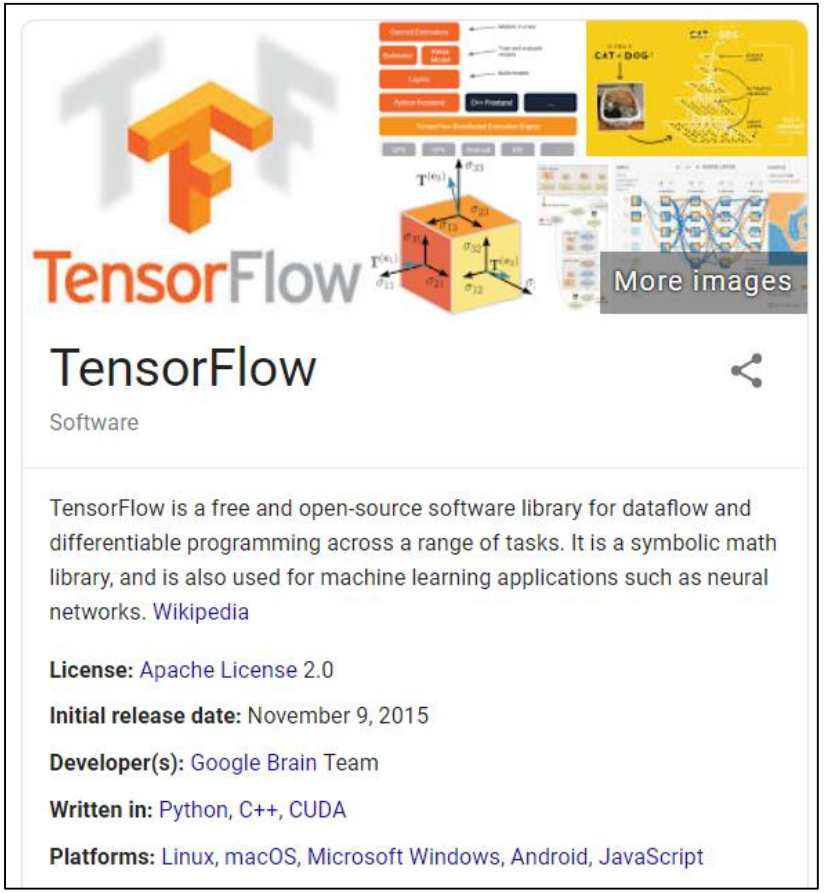
Tools for Deep Learning

- Backend
  - TensorFlow (Google)
  - Scikit-Learn (Google)
  - Theano (Univ of Montreal)
  - CNTK (Microsoft)
  - Torch + PyTorch (Facebook)
  - Caffe (UC Berkeley)
  - H2O

- Frontend
  - Python
  - R
  - Keras
  - Apache MXNet (Amazon)

TensorFlow: Google

- **TensorFlow** is an open source software library released in 2015 by Google to make it easier for developers to design, build, and train deep learning models
- At a high level, **TensorFlow** is a Python library that allows users to express arbitrary computation as a graph of data flows



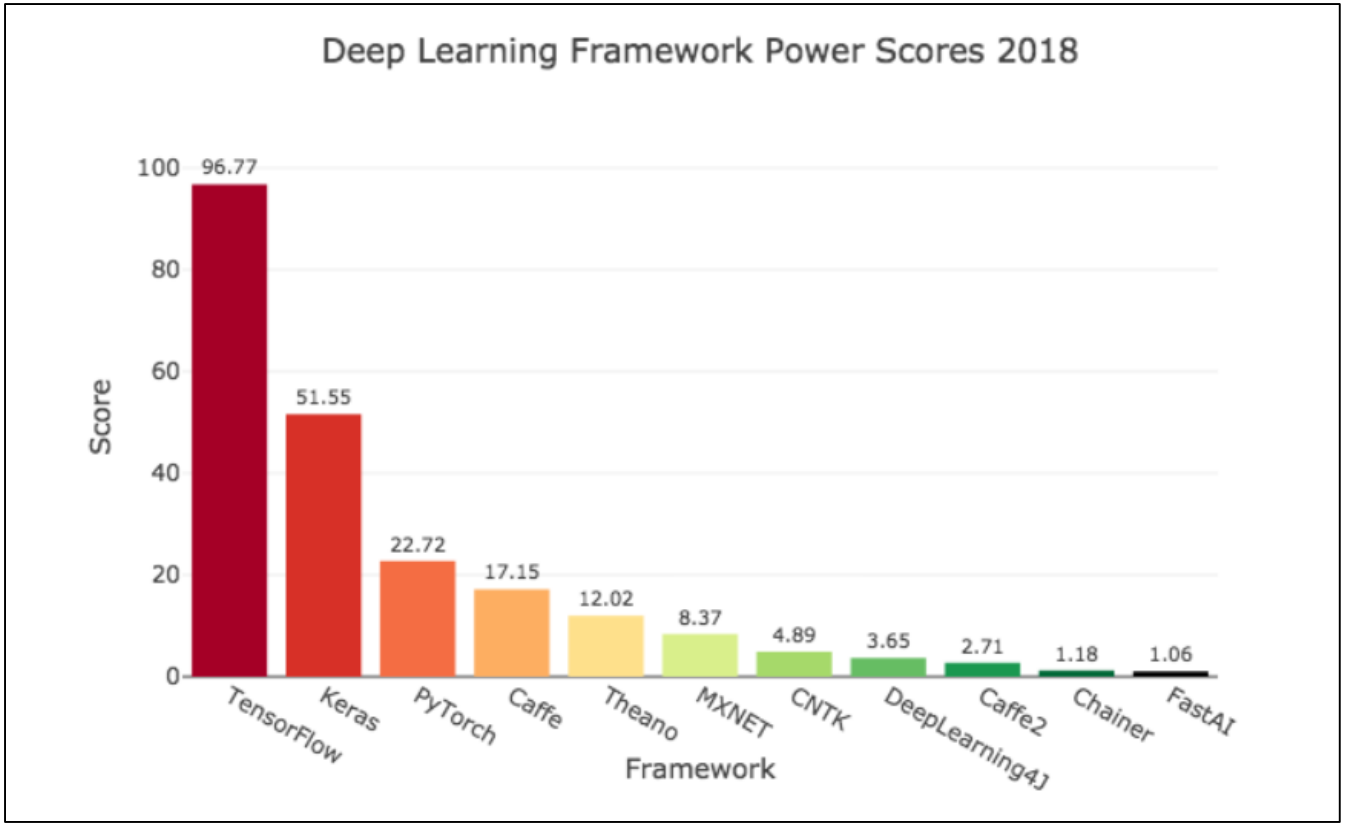
Deep Learning

Deep Learning Framework Power Scores 2018

Who's on top in usage, interest, and popularity?

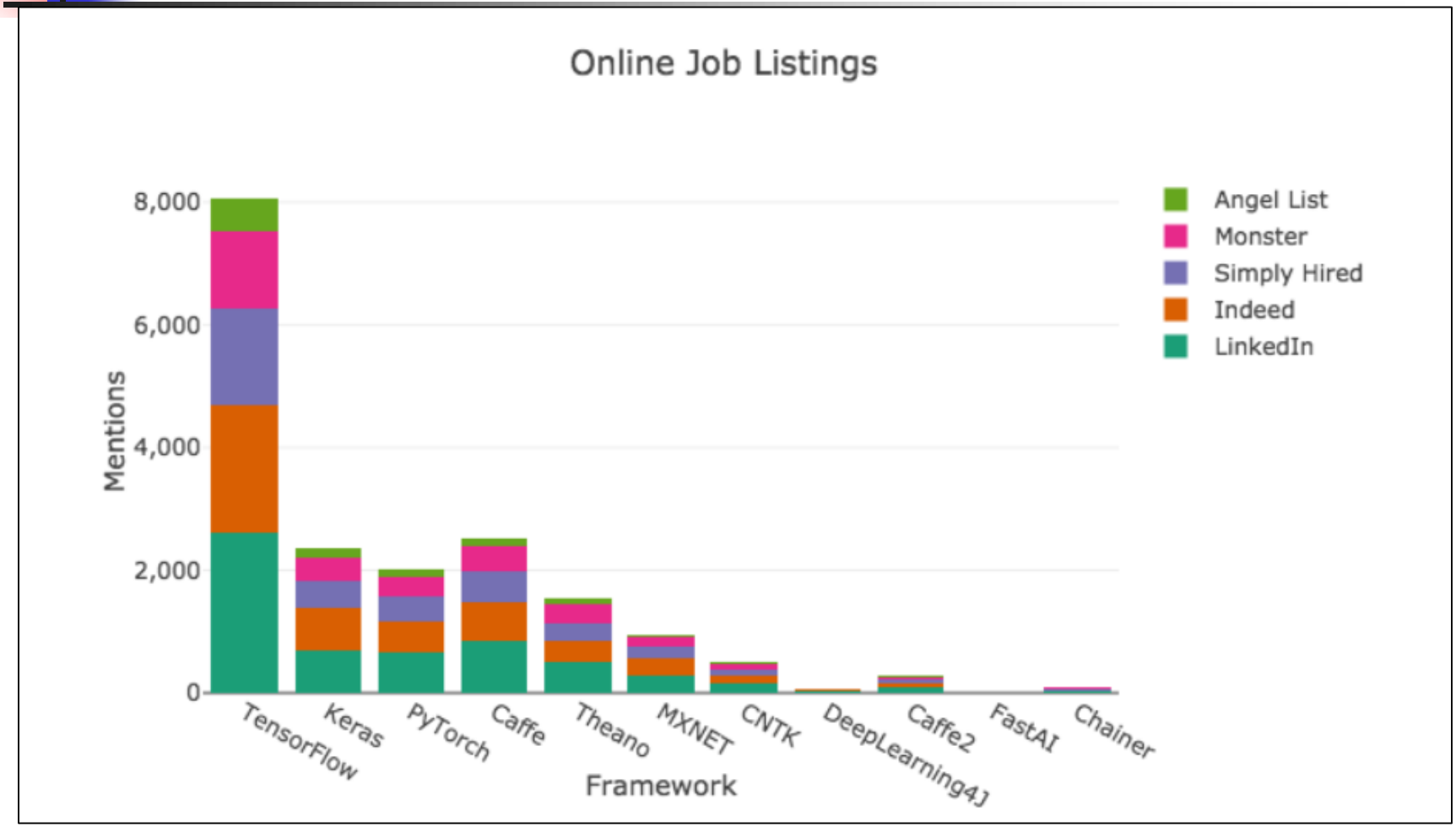
Jeff Hale Follow

Sep 19, 2018 · 10 min read



1. TensorFlow
2. Keras
3. PyTorch
4. Caffe
5. theano
6. mxnet
7. CNTK
8. DL4J
9. Caffe2
10. Chainer
11. fast.ai

Job Listings



Theano: Univ of Montreal

- Theano is a Python library and optimizing compiler for manipulating and evaluating mathematical expressions, especially matrix-valued ones.

Theano

theano

Developer(s)	Montreal Institute for Learning Algorithms (MILA), University of Montreal
Initial release	2007; 12 years ago
Stable release	1.0.4[1] / 16 January 2019; 57 days ago
Repository	github.com/Theano/Theano
Written in	Python, CUDA
Platform	Linux, macOS, Windows
Type	Machine learning library
License	The 3-Clause BSD License
Website	www.deeplearning.net/software/theano/

- Theano is an open source project primarily developed by Montreal Institute for Learning Algorithms (MILA) at the Université de Montréal.

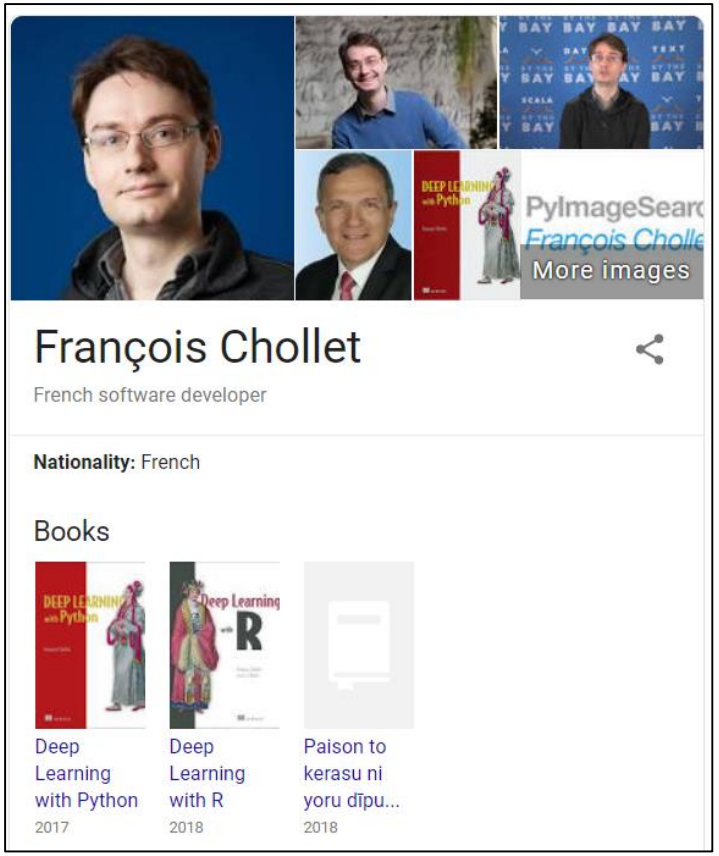
- September 2017: Yoshua Bengio, Head of MILA:
  - Major development would cease after the 1.0 release due to competing offerings by strong industrial players.

Yoshua Bengio

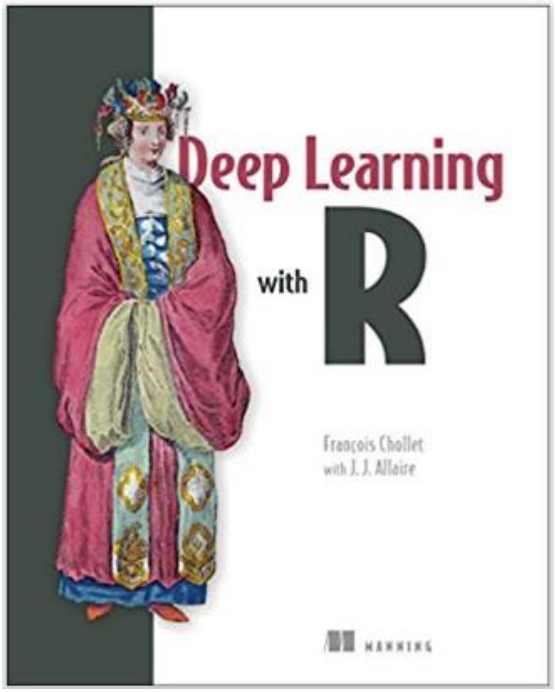
Yoshua Bengio, October 27, 2016

Born	1964 (age 54–55) France
Residence	Montreal, Quebec
Citizenship	Canada
Alma mater	McGill University

Keras: Google

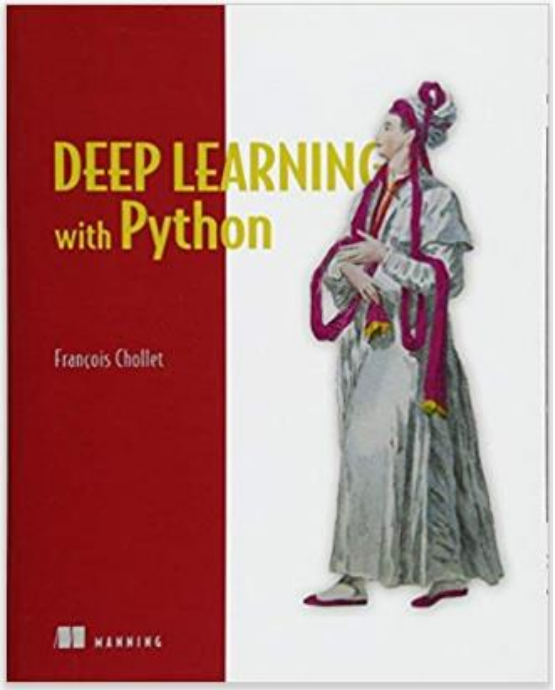


BOOK: Deep Learning with R



Authors: Francois Chollet, J. J. Allaire

J.J. Allaire is the CEO of RStudio



Authors: Francois Chollet

Misunderstanding about TensorFlow

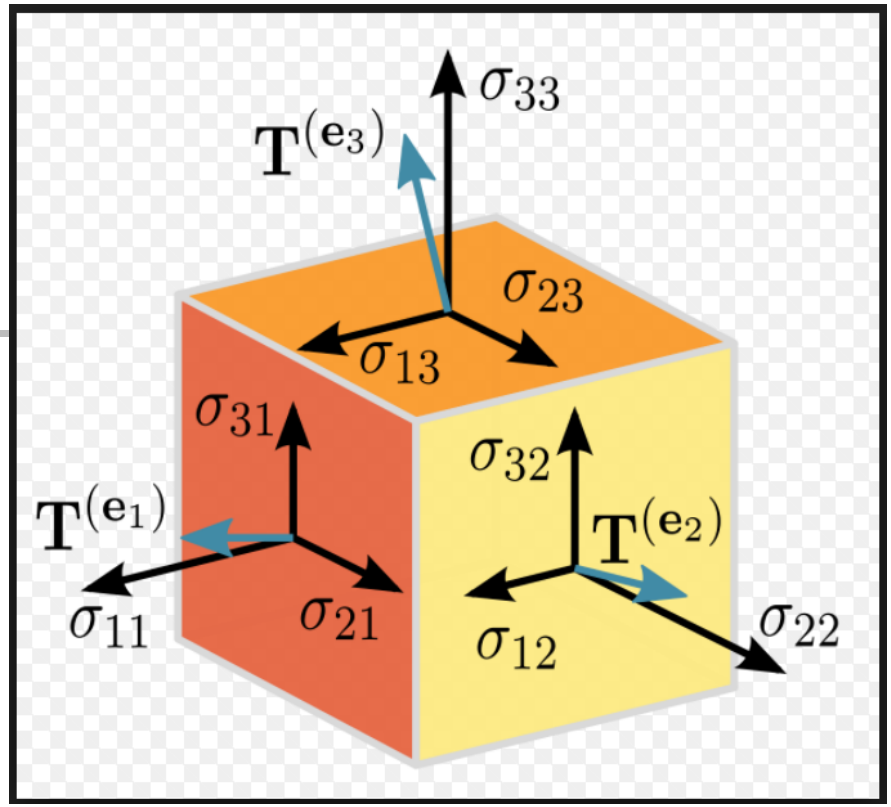
- TensorFlow is used only for Deep Learning Neural Networks
  - NOT TRUE
  - TF is a software for Mathematical Computations
- TensorFlow is used only with Python
  - NOT TRUE
  - TF can be used with many programming language like R

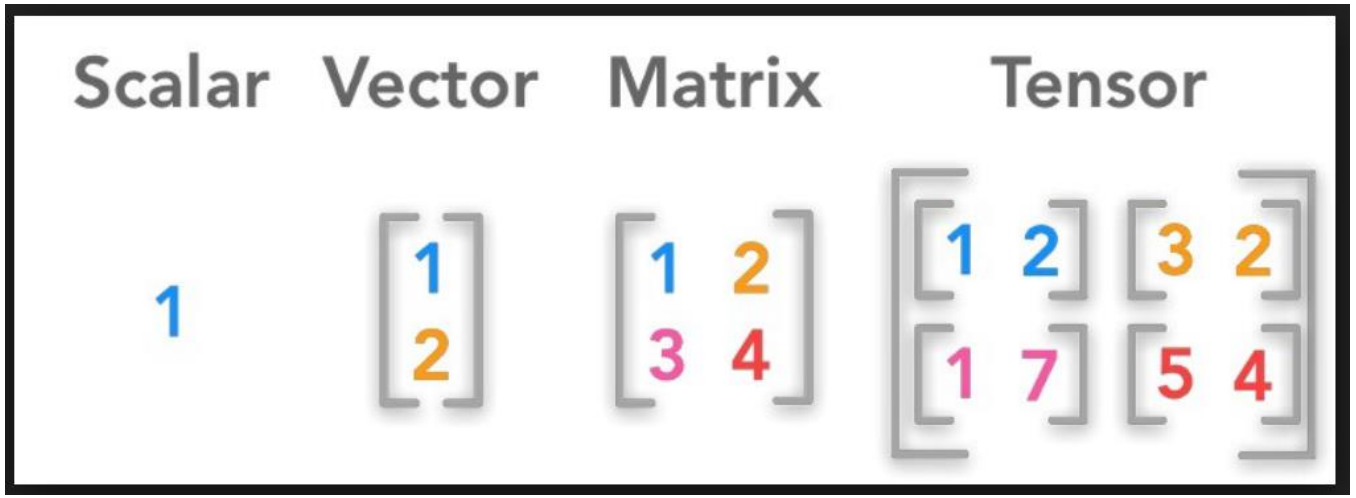
GPU: Nvidia
CUDA: Parallel Processing Platform

- Nvidia GPUs are used in
  - Deep learning
  - Artificial intelligence
  - Gaming Application

Tensor

- Scalar is a Tensor of Rank 0
- Vector is a Tensor of Rank 1
- Matrix is a Tensor of Rank 2





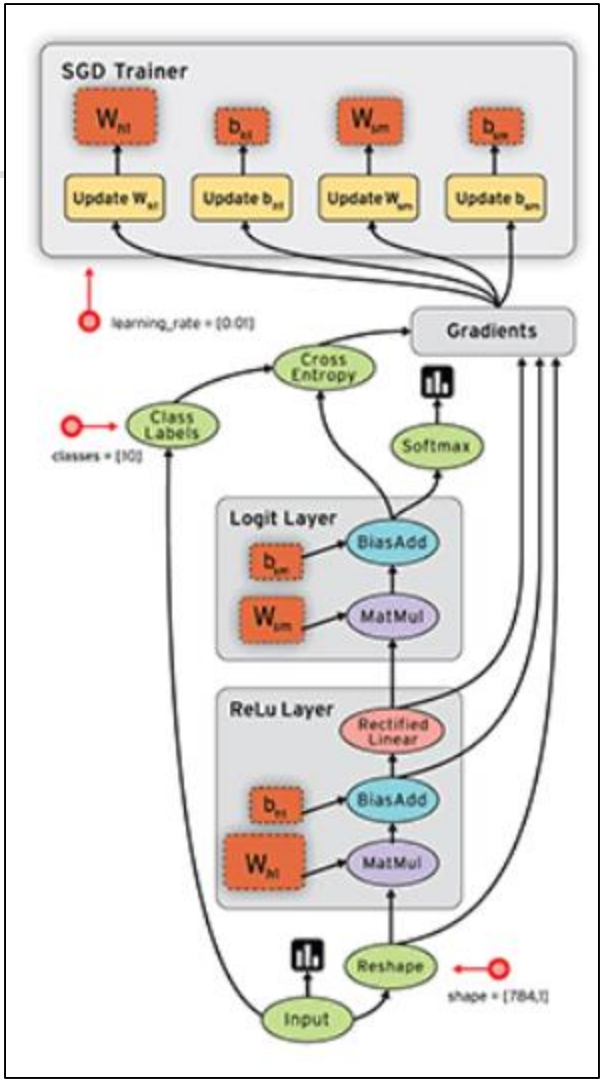
Examples of Tensors

A tensor is an N-dimensional array of data

Name	Rank	Example	Shape
Scalar	0	x0 = tf.constant(3)	()
Vector	1	x1 = tf.constant([3,5,7])	(3,)
Matrix	2	x2 = tf.constant([[3,5,7],[4,6,8]])	(2,3)
3D Tensor	3	x3 = tf.constant([[[3,5,7],[4,6,8]], [[1,2,3],[4,5,6]]])	(2,2,3)
nD Tensor	n	x1 = tf.constant([2,3,4]) x2 = tf.stack([x1, x1]) x3 = tf.stack([x2, x2, x2, x2]) x4 = tf.stack([x3, x3])	(3,) (2,3) (4,2,3) (2,4,2,3)

TensorFlow

- Creates Directed Acyclic Graph (DAG)
  - DAG represents mathematical operations
    - + - * /
    - Vector arithmetic
    - Matrix multiplication
- DAG
  - Edges
    - Input/output of math operation
    - Represents – array of data

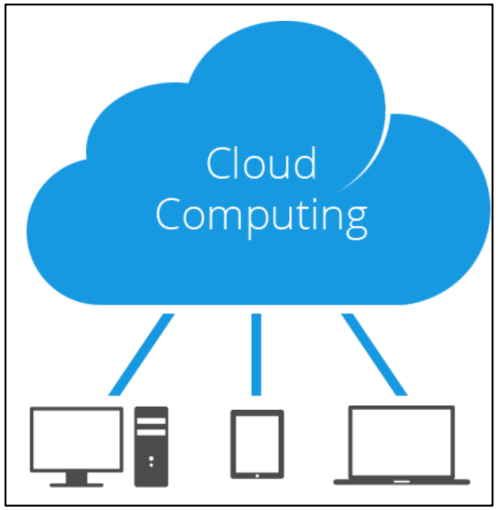


Advantages of Directed Acyclic Graph (DAG)

- Portability – hardware and software
  - DAG is language independent representation of code of your model
  - Dag can be used with C++ and Python
  - CPU or GPU
- Efficient execution of the code on different hardware running in parallel
- Similar to Java Virtual Machine (JVM)
  - Works on all platforms

Advantages of DAG

Train your model on Cloud where you have access to very powerful hardware and storage devices with lots of data

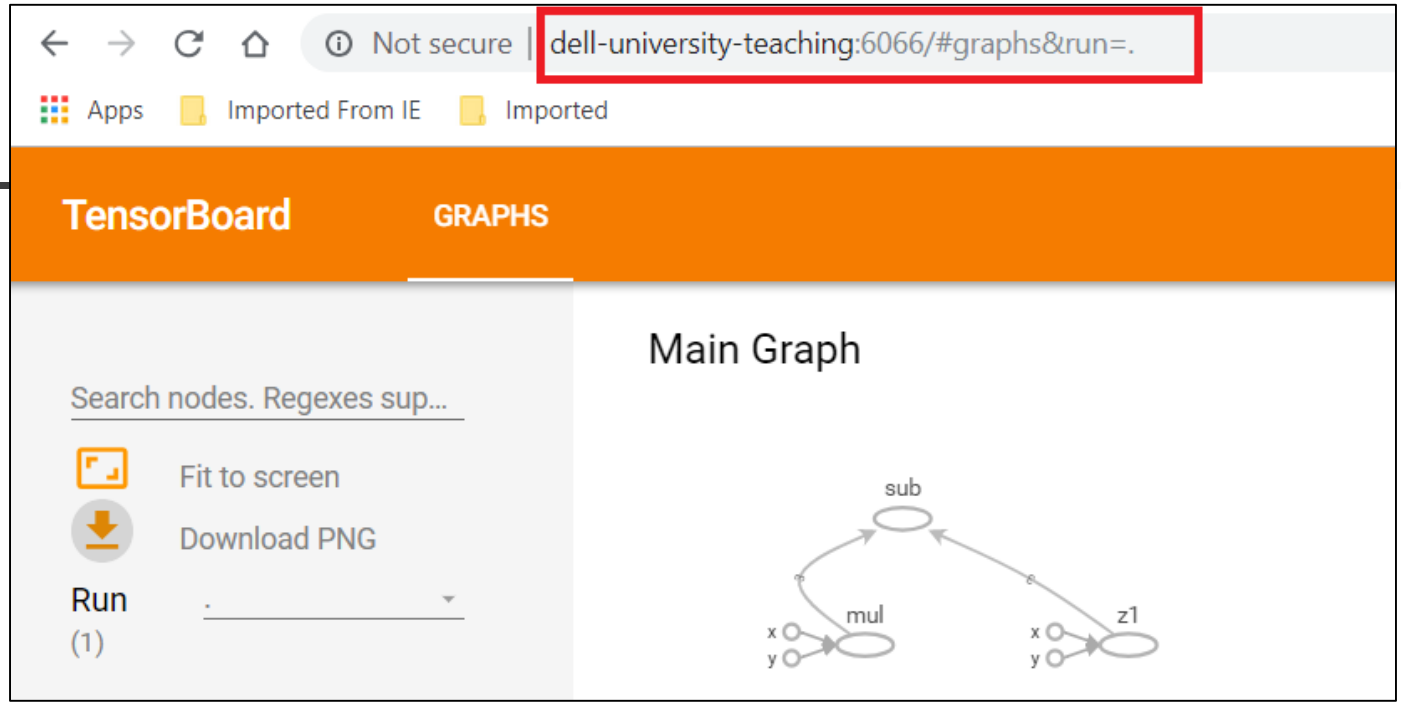


Run your model on cell phone

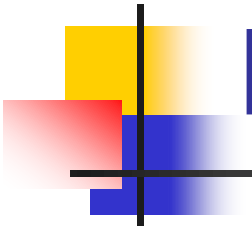
TensorFlow Abstraction Layers
API Hierarchy

	API	
1	tf.estimator	High level API
2	tf.layers, tf.losses, tf.metrics	Custom NN Model components • tf.layers: ReLu activation function, create new hidden layer • tf.metrics: RMSE • tf.losses: Entropy with Logit
3	Core TensorFlow (Python)	Python API: Numeric processing code Add, subtract, multiply, divide matrix Creating variables, Tensors, getting the shape
4	Core TensorFlow (C++)	C++ API Writing custom app
5	CPU + GPU + TPU + Android	TensorFlow for different hardware

DAG of the Computation




```
import tensorflow as tf

x = tf.constant([3,5,7],name="x")
y = tf.constant([1,2,3],name="y")
z1 = tf.add(x,y,name="z1")
z2 = x*y
z3 = z2 - z1

with tf.Session() as sess:
    with tf.summary.FileWriter("summaries", sess.graph) as writer:
        a1, a3 = sess.run([z1,z3])
```



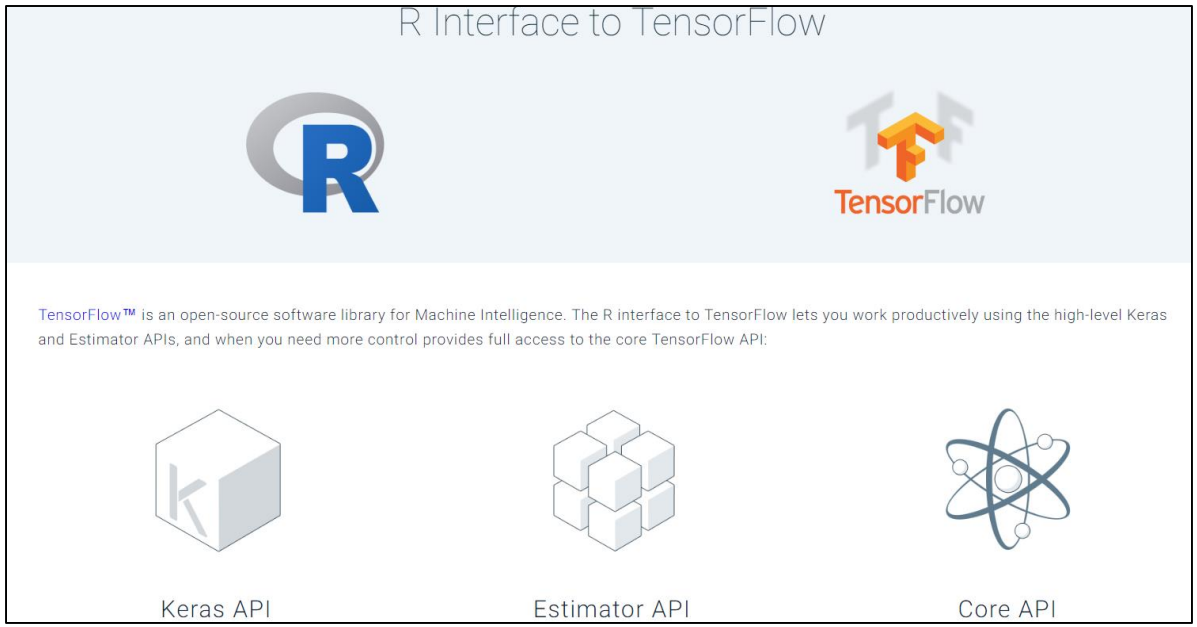
Placeholder

- Placeholders allow you to feed in values into a graph

```
import tensorflow as tf

a = tf.placeholder("float",None)

b = a*4

print(a)
Tensor("Placeholder:0", dtype=float32)

with tf.Session() as session:
    print(session.run(b,feed_dict={a:[1,2,3]}))


[  4.   8.  12.]
```

Installing TensorFlow in R

- https://tensorflow.rstudio.com
- Core API
- Keras API
- Estimator API



Install: TensorFlow Core API

```
#######################################################
# Install TensorFlow in R + RStudio
#
#######################################################
#install.packages("devtools")
library(devtools)

#######################################################
devtools::install_github("rstudio/tensorflow")

library(tensorflow)

# The following command will take approximately 5 minutes or more to install TensorFlow
install_tensorflow()

#######################################################

> sess = tf$Session()
> hello = tf$constant('Hello TensorFlow')
> sess$run(hello)
b'Hello TensorFlow’
> #######################################################
> # Version number of TensorFlow
> #
> tensorflow::tf_config()
TensorFlow v1.10.0 (C:\Users\ash\AppData\Local\conda\conda\envs\r-tensorflow\lib\site-
packages\tensorflow\__init__.p)
Python v3.6 (C:\Users\ash\AppData\Local\conda\conda\envs\r-tensorflow\python.exe)
>
```

Install: TensorFlow with Keras API

INSTALLATION

First, install the keras R package from GitHub as follows:

```
devtools::install_github("rstudio/keras")
```

The Keras R interface uses the TensorFlow backend engine by default. To install both the core Keras library as well as the TensorFlow backend use the `install_keras()` function:

```
library(keras)
install_keras()
```

This will provide you with default CPU-based installations of Keras and TensorFlow. If you want a more customized installation, e.g. if you want to take advantage of NVIDIA GPUs, see the documentation for `install_keras()`.

Install: TensorFlow with Estimator API

INSTALLATION

To use **tfestimators**, you need to install both the R package as well as TensorFlow itself.

First, install the tfestimators R package from CRAN as follows:

```
install.packages("tfestimators")
```

Then, use the `install_tensorflow()` function to install TensorFlow:

```
install_tensorflow()
```

This will provide you with a default installation of TensorFlow suitable for getting started. See the article on installation to learn about more advanced options, including installing a version of TensorFlow that takes advantage of NVIDIA GPUs if you have the correct CUDA libraries installed.

The following canned estimators are currently available:

Estimator	Description
`linear_regressor()`	Linear regressor model.
`linear_classifier()`	Linear classifier model.
`dnn_regressor()`	DNN Regression.
`dnn_classifier()`	DNN Classification.
`dnn_linear_combined_regressor()`	DNN Linear Combined Regression.
`dnn_linear_combined_classifier()`	DNN Linear Combined Classification.

R Code with TensorFlow

- Install TensorFlow + Keras + Estimator
- Basic TensorFlow Operations wit R
- Regression
  - R 'lm' function
  - R + TensorFlow
  - R + Keras + TensorFlow
  - R + Estimator + TensorFlow

Preview of my next Talk
OC R User's Group: April 30, 2019

- Title: Deep Learning Optimization Techniques
  - Gradient Descent
  - Stochastic Gradient Descent
  - Momentum
  - Nesterov Momentum
  - AdaGrad
  - RMSProp
  - Adam: Adaptive Moments