Fall 2018

Deep Learning for Computer Vision, Speech, and Language

Liangliang Cao, Xiaodong Cui, Kapil Thadani

https://columbia6894.github.io/

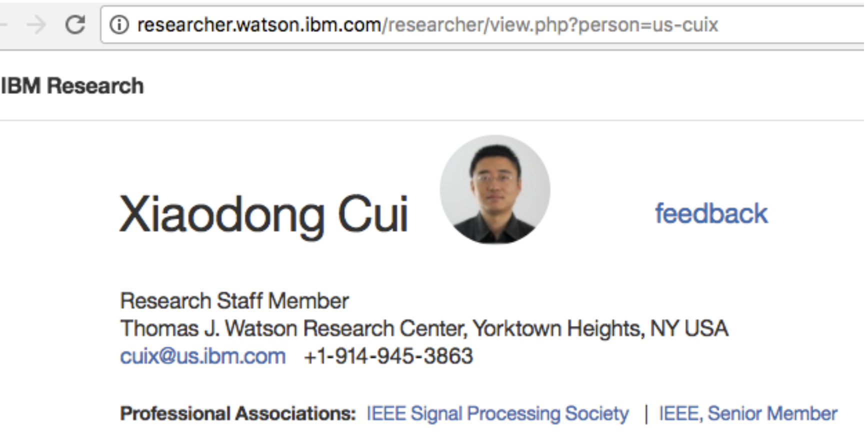
Outline

- *How to register?*
- *Who we are*
- *Grading*
  - *Homework*
  - *Projects*
- *Course schedule and resource*
- *Some demo of deep learning*
- *Programming basics*

How to register this class?

- Unfortunately instructors do NOT have access to the waiting list
  - What we can do is to welcome sit-ins or independent study with instructors
- Each department selected 20 students
  - EE, CS and Data Science co-sponsored 60 students total
- Please drop early if you cannot finish HW#1 or do not follow
  - Talk to your department if you want to be listed in the waiting list

Lectures



researcher.watson.ibm.com/researcher/view.php?person=us-cuix

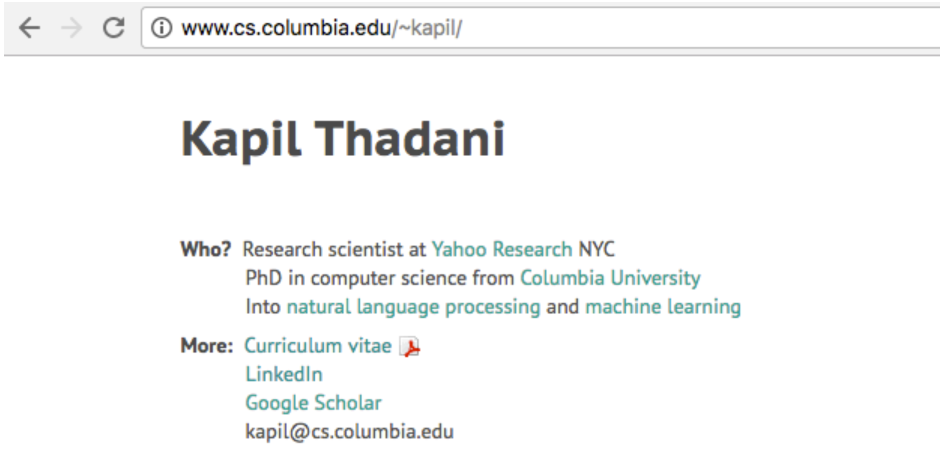
IBM Research

Xiaodong Cui

feedback

Research Staff Member
Thomas J. Watson Research Center, Yorktown Heights, NY USA
cuix@us.ibm.com +1-914-945-3863

Professional Associations: IEEE Signal Processing Society | IEEE, Senior Member

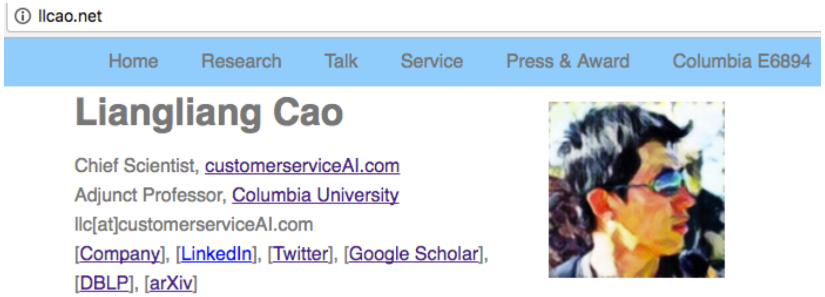


www.cs.columbia.edu/~kapil/

Kapil Thadani

Who? Research scientist at Yahoo Research NYC
PhD in computer science from Columbia University
Into natural language processing and machine learning

More: Curriculum vitae
LinkedIn
Google Scholar
kapil@cs.columbia.edu



llcao.net

Liangliang Cao

Chief Scientist, customerserviceAI.com
Adjunct Professor, Columbia University
llc[at]customerserviceAI.com
[Company], [LinkedIn], [Twitter], [Google Scholar], [DBLP], [arXiv]

A few guest lecturers announced soon

Teaching assistants

- Rajath Kumar

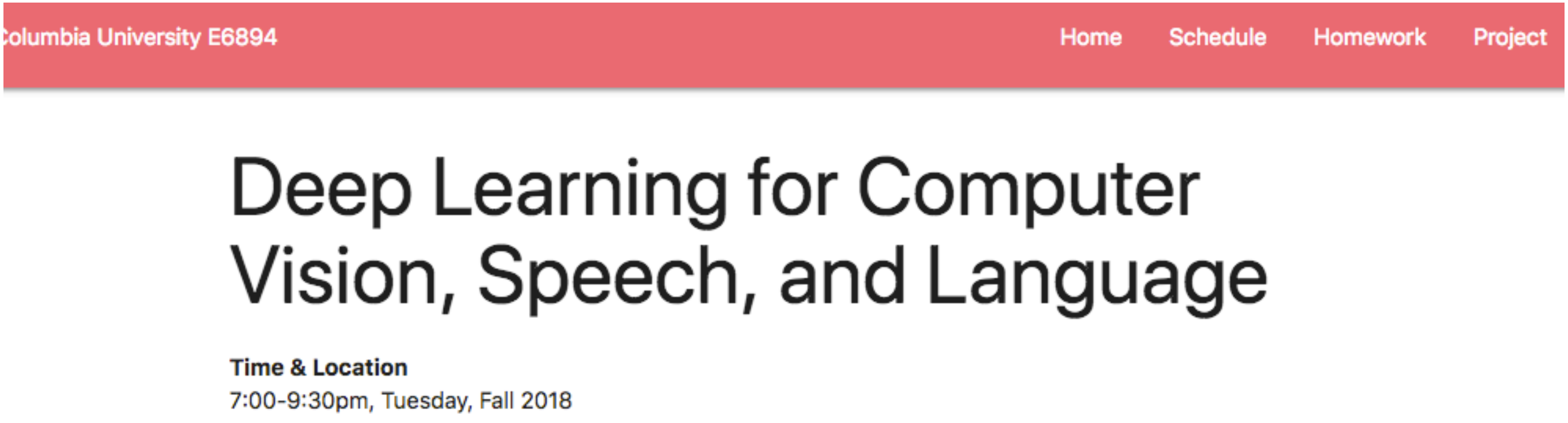
rm3497@columbia.edu

- Qiao Zhang

qz2301@columbia.edu

Website

- Slides and materials will be available on the website https://columbia6894.github.io/



Deep Learning for Computer Vision, Speech, and Language

Time & Location

7:00-9:30pm, Tuesday, Fall 2018

Purpose of This Class

- For Columbia graduates, teach "what can we do with deep neural networks"
- Why multi-modalities?
  - Speech, Vision, and NLP are most popular fields for deep learning
  - By comparing three fields, you may feel deep networks are no longer "black-box magic"
  - We hope you can generalize these success to multi-modal problem or a new domain
- How?
  - Course, Homework and Projects

Grading

- 40% project
  - In previous class the best team published paper in top/premium conferences

- 40% homework
  - HW1 is important
  - Present one paper on the important research breakthough

- 20% paper presentation and course attendance

Examples of Successful Projects

- Hassan Akbari and Himani Arora, Speech from Lip Videos, ICASSP 2018
- Nikolai Yakovenko, Poker-CNN, AAAI 2016
- Chris Cleveland, PIXM (startup) 2016
- John Bowler and Mo Zhou, Axon Segmentation, WACV 2015
- Yin Cui, Y. Xiang, and K. Rong, Galaxy Image Retrieval, WACV 2014

Course requirements

- Knowledgeable about NLP and/or speech and/or vision and/or machine learning
- Fluent in Python.
- Know Tensorflow (or pyTorch) or you can learn it quickly
- Willing to work with GPUs.

Why Python?

- Free (*not like Matlab!*)
- Much easier to use than CUDA C/C++
- THE choice for scientific computing and cloud service
- If you do not know python, please consider to drop coz it will be too hard to follow the class.

How to access GPU?

- Build one
  - If you have a (relative new) desktop, you should add a GPU card with $1000 (eg. NVidia GTX1080Ti or Titan XP)

- Use Google cloud
  - $300 free credit for ever email
  - Free credit via Columbia CRF (coming soon)

Course schedule

1. Overview (class 1-4)
   - Course overview and Tensorflow basic
   - Review of NN and Optimization
   - Review of NLP basic
   - Review of CNN

2. Deep learning for Speech, Language, and Vision

   Each class focuses one topic with

   a) Lectures by the instructor/guest speaker

   b) One homework per topic

Student presentation

Details to be announced in the next class

Reference: last year's procedure:

- Form a team with two students
- Select one paper (from the list suggested by the instructors)
- Prepare a 20 mins presentation, at least 15 pages slides
- Demos/source code analysis are welcome

Final project

- Team work: 2-3 students per group
- Goal:
  - Develop the state-of-the-art deep learning techniques.
  - Try to solve real problems with the knowledge you learned
- Format:
  - 4 pages double column (e.g., in ICASSP format)
  - or 8 pages single column (e.g., in NIPS format)
- Evaluation
  - Students' vote: Idol Award
  - Instructor's pick: AI conference quality

(I only write recommendation letters for students with conference-quality projects)

Which toolkit shall I use for project

- Tensorflow (huge society, by Google)
  - **Keras** (high level interface)
  - Good for development and deployment

- Other choices:
  - PyTorch (Good for sequential research)
  - MxNet

Mastering the tools

- Use Python Notebook (Jupyter)
  - http://jupyter.org/try
  - colab.research.google.com
  - Submit homework with results in python notebook!
- Use Git for team project

  - Create a personal account on github.com

  - Understand git commands

Power of Deep Networks

- AlphaGo Zero by David Silver
- Google Cloud Vision API
- Visual Memory QA
- Super SloMo
- Watson Text to Speech and Watson Speech to Text

Short Break

https://columbia6894.github.io/

Programming: Tensorflow, Keras and Homework

Liangliang Cao

https://columbia6894.github.io/

Install Tensorflow

- Documentation
  - https://www.tensorflow.org/
  - We suggest to use virtual env
- Straightforward installation
  - On cloud, you need setup a project
  - On your local machine, type "pip install tensorflow" and install other related libraries
- In this class, we'll use Codelab to demonstrate some basic concept
  - https://colab.research.google.com/

Which is True for Tensorflow?

- A python framework of computing math expression

- Designed for large scale data

- Designed for the specific purpose of deep learning

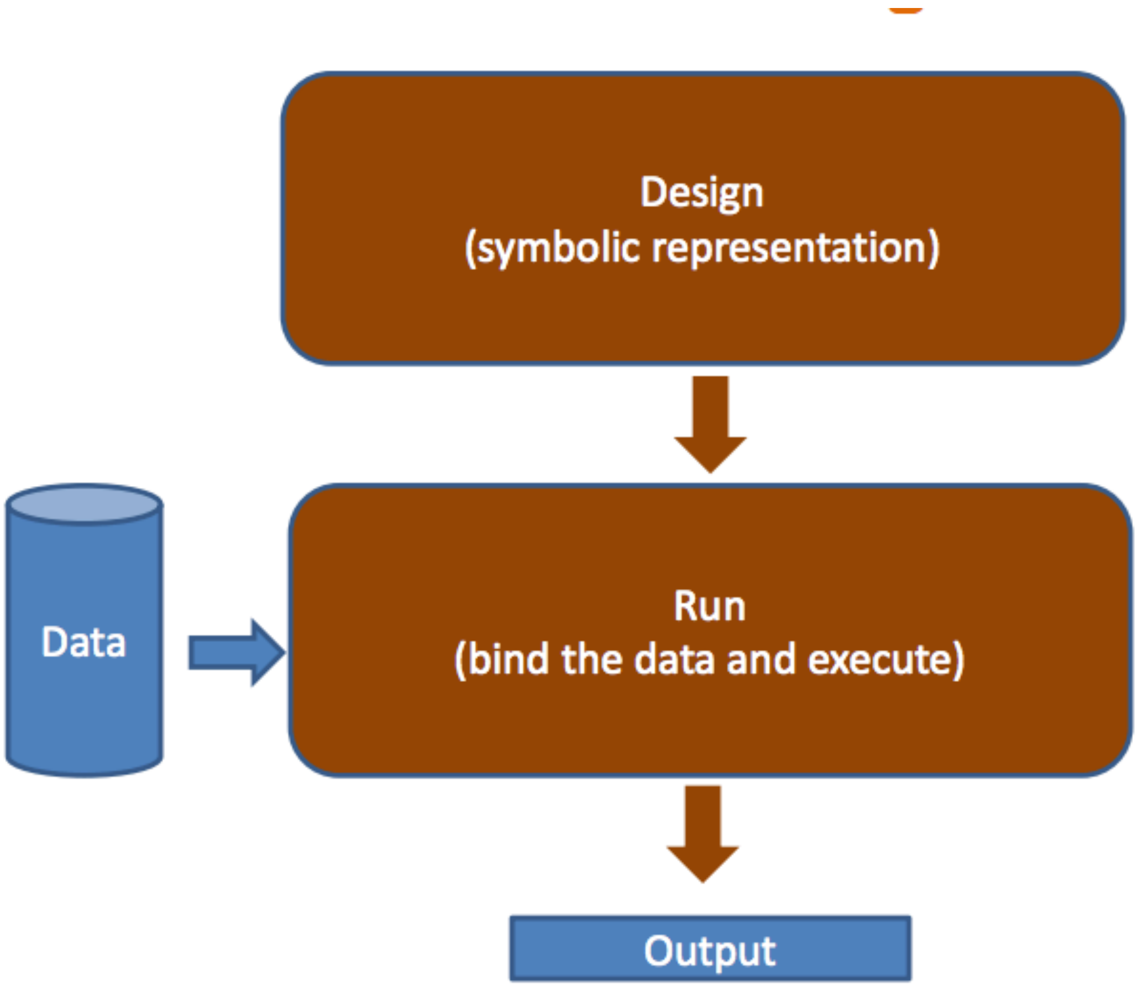
Which is True for Tensorflow?

- A python framework of computing math expression
  - Similar with Theano

- Designed for large scale data
  - Excellent engineering

- ~~Designed for the specific purpose of deep learning~~
  - Tensorflow's low level APIs are for general purpose.

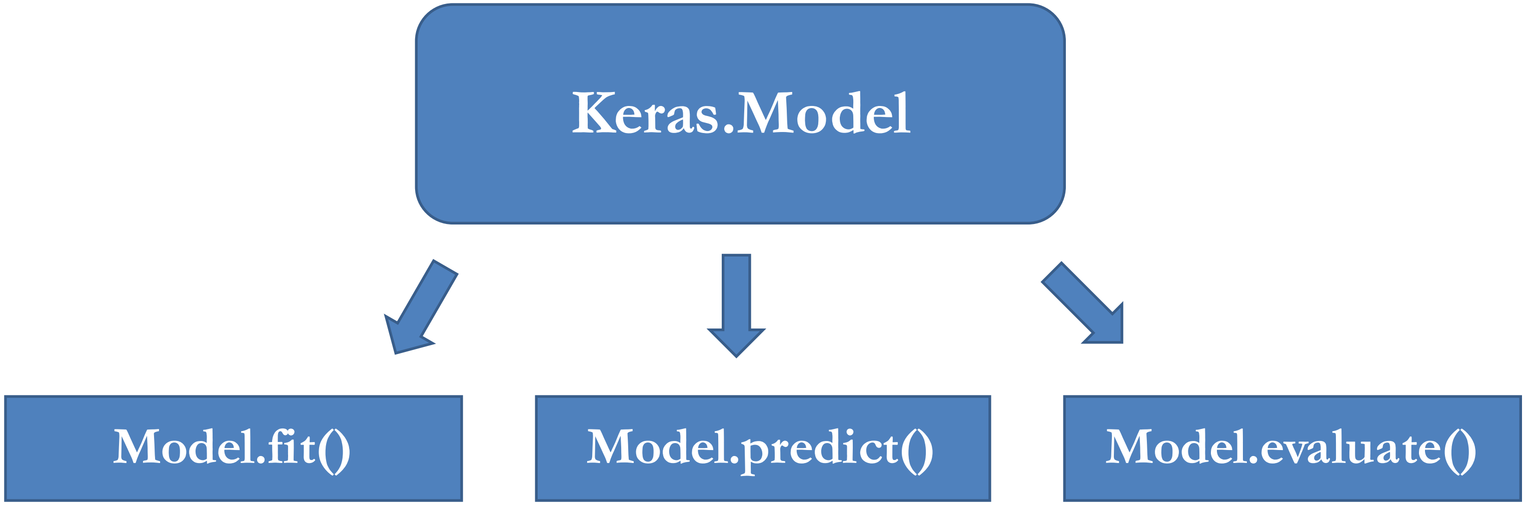
Tensorflow’s Design



Keras Design

Keras is a wrapper of Tensorflow

from tensorflow.python import keras



Keras.Model

Model.fit()

Model.predict()

Model.evaluate()

Reference: https://keras.io/

A Simple Tensorflow Example

```
import tensorflow as tf
a = tf.add([2,1], [1,2])
print(a)
```

What is the output?

A Simple Tensorflow Example

```
import tensorflow as tf
a = tf.add([1,2], [2,1])
print(a)
```

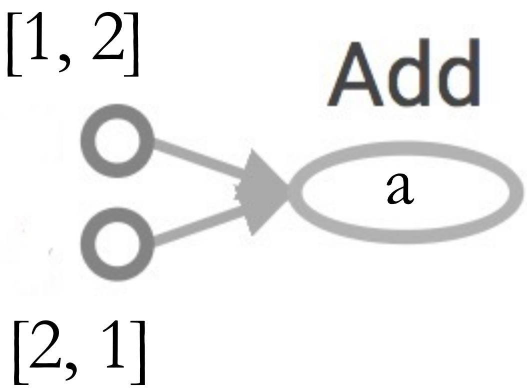
What is the output?

a) [3, 3]
b) Tensor("Add:0", shape=(2, 1), dtype=int32)
c) None of above

A Simple Tensorflow Example

```
import tensorflow as tf
a = tf.add([1,2], [2,1])
print(a)
```

What is the output?



[1, 2]

Add

a

[2, 1]

A tf graph includes symbolic objects

A tf session allocates memory to evaluate symbolic objects

A Simple Tensorflow Example

```
import tensorflow as tf
a = tf.add([1,2], [2,1])
print(a)
with tf.Session()  as sess:
      print sess.run(a)
```

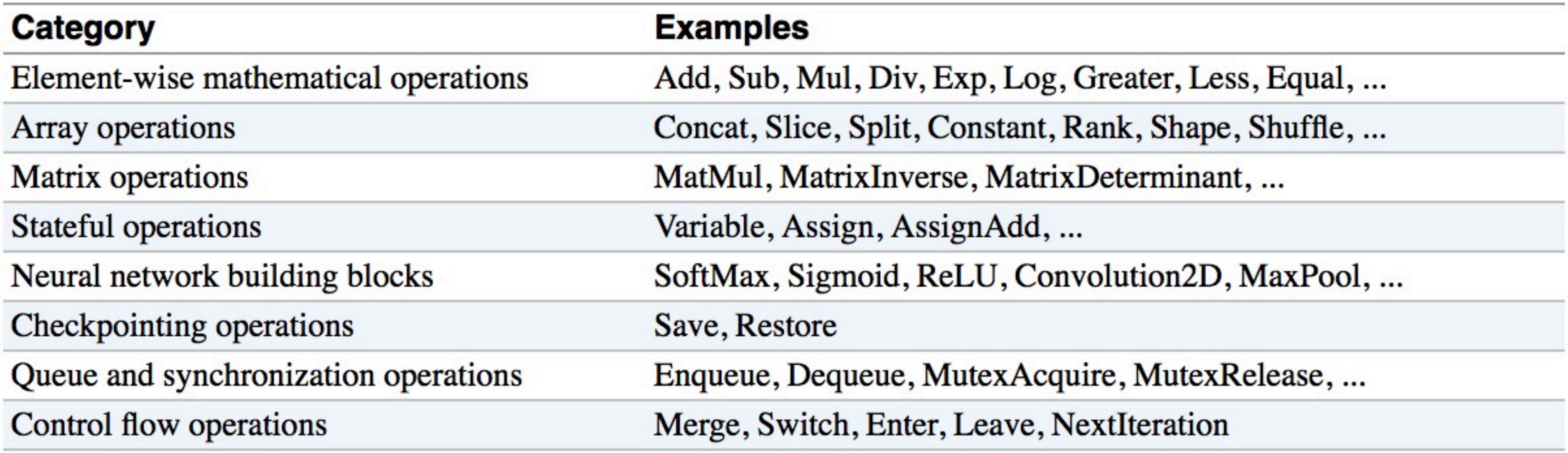
What is the new output?

Why Graph and Session?

- Optimize computation. The graph model will be optimized before evaluating

- Facilitate distributed computation, spread the work across multiple CPUs, GPUs, or TPUs.

Tensorflow Low-level APIs



Category	Examples
Element-wise mathematical operations	Add, Sub, Mul, Div, Exp, Log, Greater, Less, Equal, ...
Array operations	Concat, Slice, Split, Constant, Rank, Shape, Shuffle, ...
Matrix operations	MatMul, MatrixInverse, MatrixDeterminant, ...
Stateful operations	Variable, Assign, AssignAdd, ...
Neural network building blocks	SoftMax, Sigmoid, ReLU, Convolution2D, MaxPool, ...
Checkpointing operations	Save, Restore
Queue and synchronization operations	Enqueue, Dequeue, MutexAcquire, MutexRelease, ...
Control flow operations	Merge, Switch, Enter, Leave, NextIteration

Refer to Chip Huyen’s course on Tensorflow

A Magic Function

```
import tensorflow as tf
x = tf.placeholder(tf.float32)
y =  2*x + x*x
g = tf.gradients(x + y, [x, y])
with tf.Session()  as sess:
        print sess.run(g, feed_dict={x:1.0})
```

Tf.gradient allows automat gradient calculation.

Super useful for optimization (next class)

Another Example

```
import tensorflow as tf
X = tf.placeholder(tf.float32, name='X')
Y = tf.placeholder(tf.float32, name='Y')

w = tf.get_variable('weights', initializer=tf.constant(0.0))
b = tf.get_variable('bias', initializer=tf.constant(0.0))

Y_predicted = w * X + b
loss = tf.square(Y - Y_predicted, name='loss')
optimizer = tf.train.GradientDescentOptimizer(learning_rate=0.001).minimize(loss)
with tf.Session() as sess:
    sess.run(tf.global_variables_initializer())
    for i in range(100): # run 100 epochs
        for x, y in data:
            sess.run(optimizer, feed_dict={X: x, Y:y})
    w_out, b_out = sess.run([w, b])
```

From Tensorflow to Keras

```
import tensorflow as tf

from tensorflow.python.keras.models  import Sequential
from tensorflow.python.keras.layers  import Dense, Activation

model = Sequential()
model.add(Dense(10,  input_dim=100, activation='softmax'))
model.compile(optimizer='sgd',  loss='categorical_crossentropy',  metrics=['accuracy'])
history = model.fit(X_train, Y_train, 128, nb_epoch=5,  validation_data=(X_test,  Y_test))
score = model.evaluate(X_test,  Y_test)
```

From Tensorflow to Keras

```
import tensorflow as tf

from tensorflow.python.keras.models  import Sequential
from tensorflow.python.keras.layers  import Dense, Activation

model = Sequential()
model.add(Dense(10,  input_dim=100, activation='softmax'))
model.compile(optimizer='sgd',  loss='categorical_crossentropy',  metrics=['accuracy'])
history = model.fit(X_train, Y_train, 128, nb_epoch=5,  validation_data=(X_test,  Y_test))
score = model.evaluate(X_test,  Y_test)
```

Keras is simpler to use for classification/regression problems.

Keras has a lot of wrapper functions for network building

Keras does not provide many low level APIs for large scale data

Take-home Work

Required:

- Install Jupiter Notebook
- Install Tensorflow and Keras
- Work on homework #1

  https://columbia6894.github.io/homework.html

Suggested:

- Create a github account
- Read Tensorflow and Keras tutorials