Intro to Keras

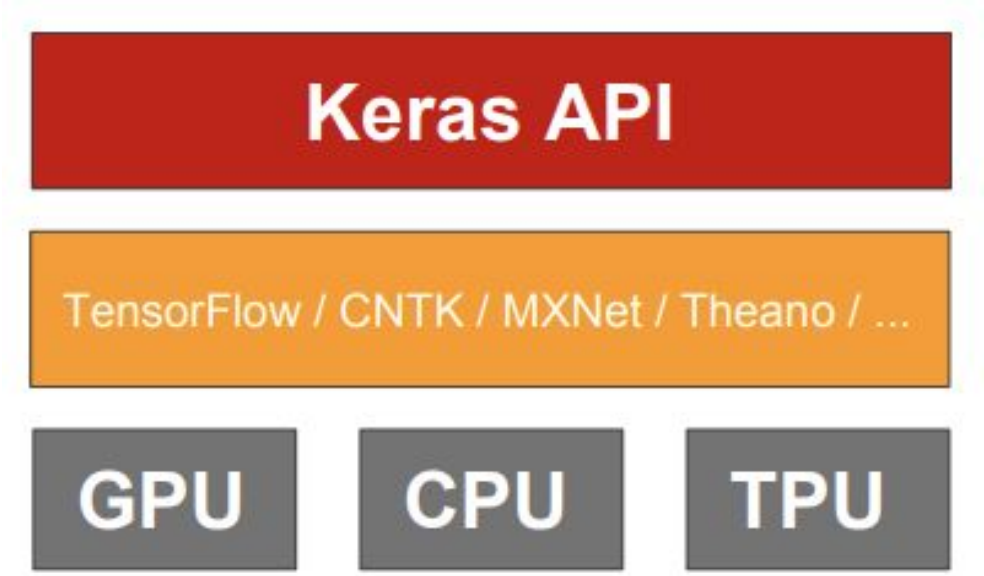
Outline

- What is Keras?
- How to use Keras
- Distributed computing with Keras
- Eager execution

What is Keras?

API or Abstraction for specifying and training

- Officially part of tensorflow project
- Optimized for tensorflow computations



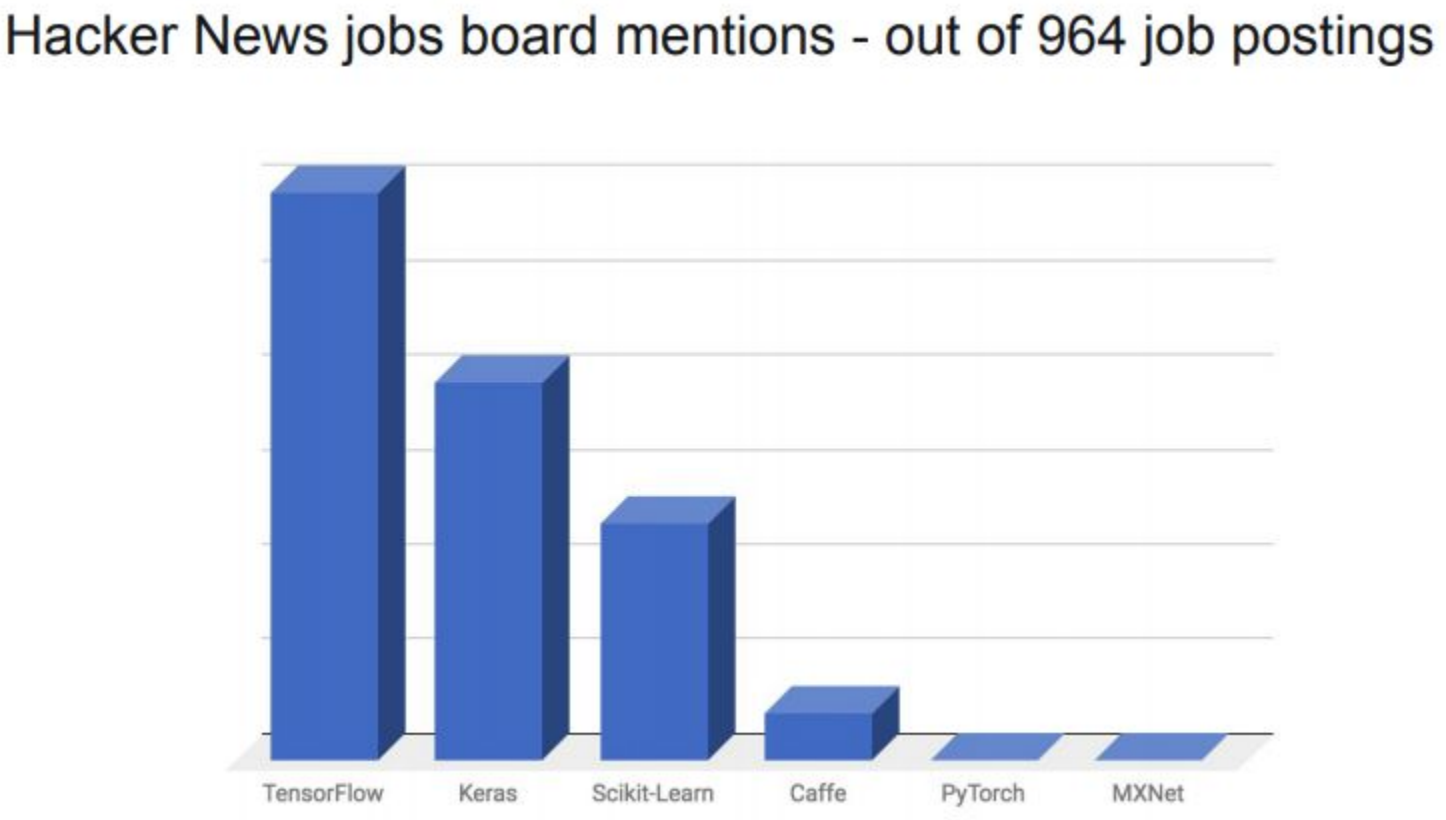
Explore the Repo

https://github.com/keras-team/keras

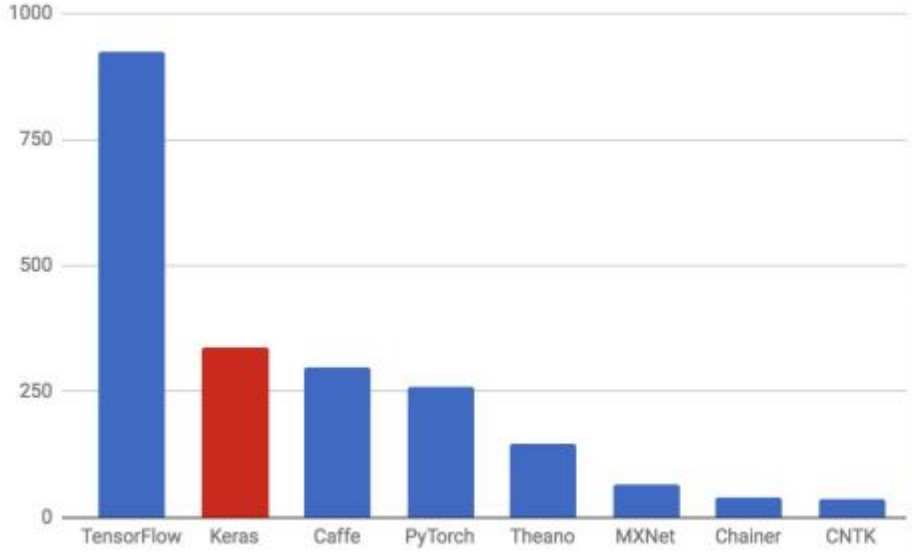
Why Why Why?

- Programmer focused
- Widespread Use
- Multiple backends supported
- Easy to productionalize/deploy models

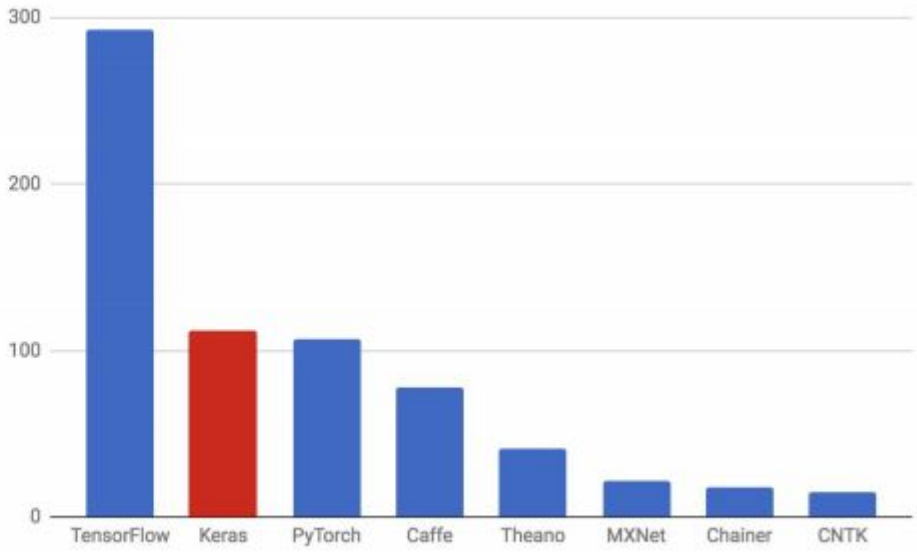
Startup-land traction



Research traction



arXiv mentions as of 2018/03/07 (past 3 months)



arXiv mentions as of 2018/03/07 (past 1 month)

User Experience

- Keras is an API designed for Humans, not machines
- Keras is easy to learn and easy to use
- The ease of use does not come with reduced performance or flexibility

Multiple Backends - Multiple Platforms

- Develop
  - Python
  - R
- Run same code with
  - Tensorflow
  - CNTK
  - Theano
  - MXnet
  - ...
- Run code on
  - CPU
  - Nvidia GPU
  - AMD GPU
  - TPU

Productionalize / Deploy

- TF Serving
- Google Cloud ML Engine
- In browser GPU acceleration (Webkeras, Keras.js, WebDNN, ...)
- Android (TF, TF lite), iPhone (native CoreML support)
- Raspberry Pi
- JVM

Go build cool AR apps with Keras + TF + CoreML + ARKit

Three API Styles

- Sequential Model
  - Dead simple
  - Only for single input single output, sequential layer stacks
  - Good for ~70% of use cases
- The functional API
  - Like playing with lego bricks
  - Multi-input, multi-output, arbitrary static graphs topologies
  - Good for ~95% of use cases
- Model subclassing
  - Maximum flexibility
  - Larger potential error surface

Sequential API

```
import keras
from keras import layers

model = keras.Sequential()

model.add(layers.Dense(20, activation='relu', input_shape=(9,)))
model.add(layers.Dense(20, activation='relu'))
model.add(layers.Dense(1))

model.compile(loss='mean_squared_error', optimizer='adam')

model.fit(X, y, epochs=10, batch_size=2)
```

The Functional API

```
import keras
from keras import layers

inputs = keras.Input(shape=(9,))
x = layers.Dense(20, activation='relu')(inputs)
x = layers.Dense(20, activation='relu')(x)

outputs = layers.Dense(1)(x)

model = keras.Model(inputs, outputs)
model.compile(loss='mean_squared_error', optimizer='adam')
model.fit(X, y, epochs=10, batch_size=2)
```

Model Subclassing

```
class MyModel(tf.keras.Model):

  def __init__(self):
    super(MyModel, self).__init__(name='my_model')
    self.dense_1 = layers.Dense(20, activation='relu', input_shape=(9,))
    self.dense_2 = layers.Dense(1)

  def call(self, inputs):
    x = self.dense_1(inputs)
    return self.dense_2(x)


model = MyModel()
model.compile(loss='mean_squared_error', optimizer='adam')
model.fit(X, y, batch_size=2, epochs=5)
```

Problems in Class