
TensorFlow

TensorFlow

Intro to TensorFlow

Sam Witteveen, GDE for Machine Learning
@sam_witteveen

About me

- Google Developer Expert for Machine Learning & Deep Learning
- Red Dragon AI 

- Use Deep Learning for Conversational Computing and Natural Language Understanding

Today's Talk

- What is TensorFlow
- How its being used around the world
- How you can get started

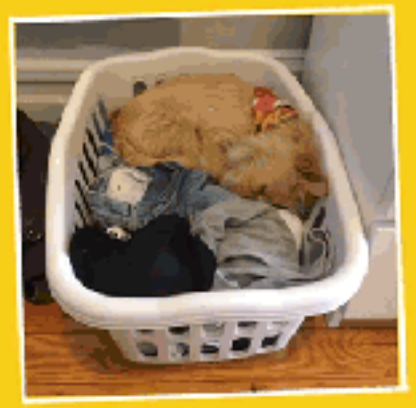
What is TensorFlow?

TensorFlow: ML for Everyone

- An open-source machine learning platform for everyone
- **Fast, flexible, and production-ready**
- Scales from research to production

IS THIS A

CAT or DOG?



CAT DOG

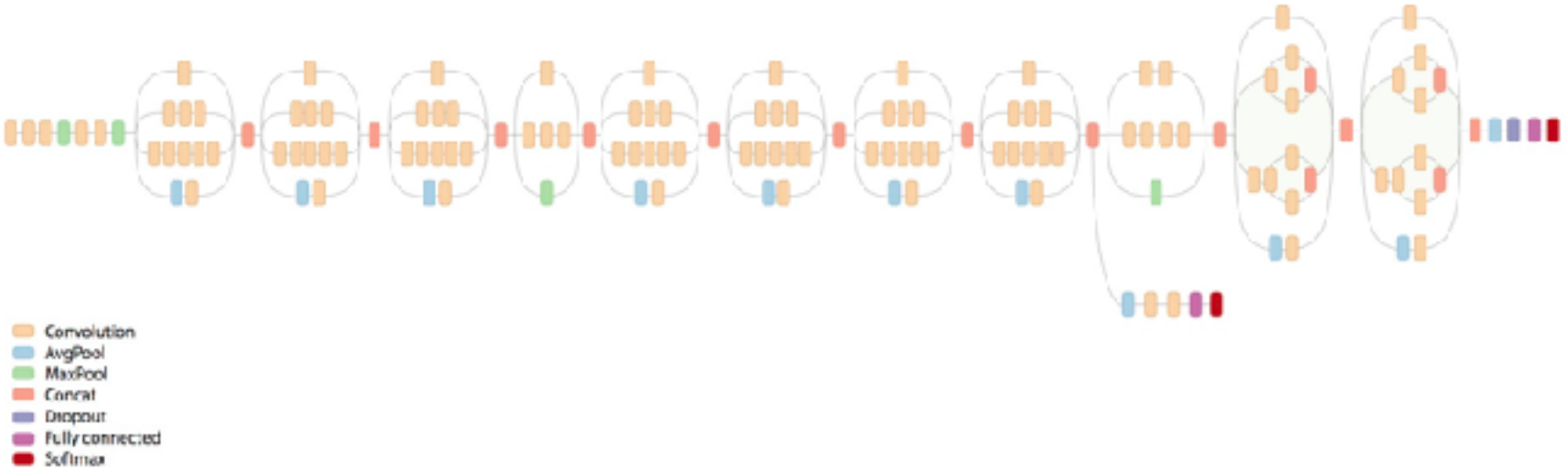
OUTPUT LAYER

ACTIVATED NEURONS

INPUT LAYER

DEEP *neural* NETWORK

Inception V3 - Deep Learning Convolutional Neural Network

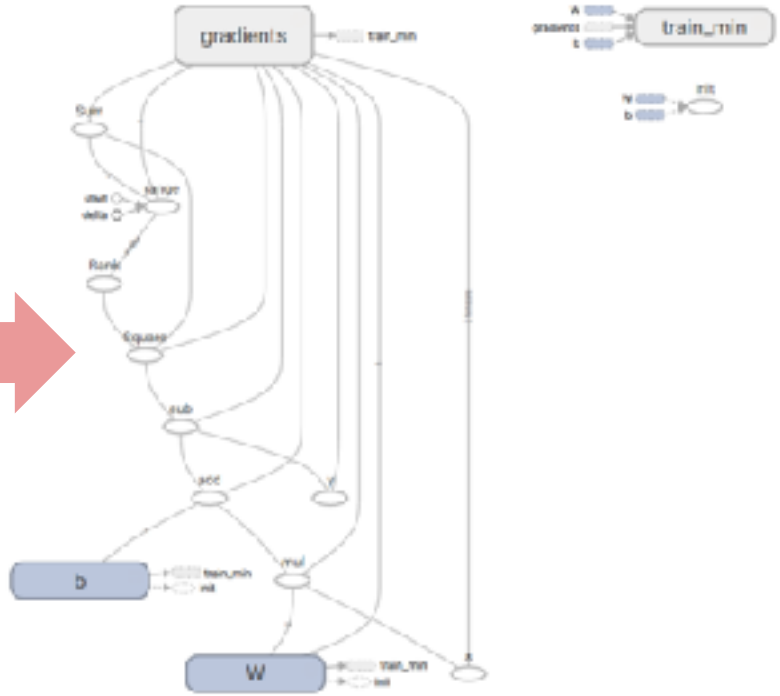


Dataflow based Computational graph

Python Program

```
import numpy as np
import tensorflow as tf

# Model parameters
W = tf.Variable([.3], tf.float32)
b = tf.Variable([-.3], tf.float32)
# Model input and output
x = tf.placeholder(tf.float32)
linear_model = W * x + b
y = tf.placeholder(tf.float32)
# loss
loss = tf.reduce_sum(tf.square(linear_model - y)) # sum of the squares
# optimizer
optimizer = tf.train.GradientDescentOptimizer(0.01)
train = optimizer.minimize(loss)
# training data
x_train = [1,2,3,4]
y_train = [0,-1,-2,-3]
# training loop
init = tf.global_variables_initializer()
sess = tf.Session()
sess.run(init) # reset values to wrong
for i in range(1000):
  sess.run(train, {x:x_train, y:y_train})

# evaluate training accuracy
curr_W, curr_b, curr_loss  = sess.run([W, b, loss], {x:x_train, y:y_train})
print("W: %s b: %s loss: %s"%(curr_W, curr_b, curr_loss))
```

TensorFlow Graph



https://www.tensorflow.org/get_started/get_started

TensorFlow Supports Many Platforms...

CPU

GPU

iOS

Android

TPU

Cloud TPU

Raspberry Pi

TensorFlow support many languages

C++

Java

python

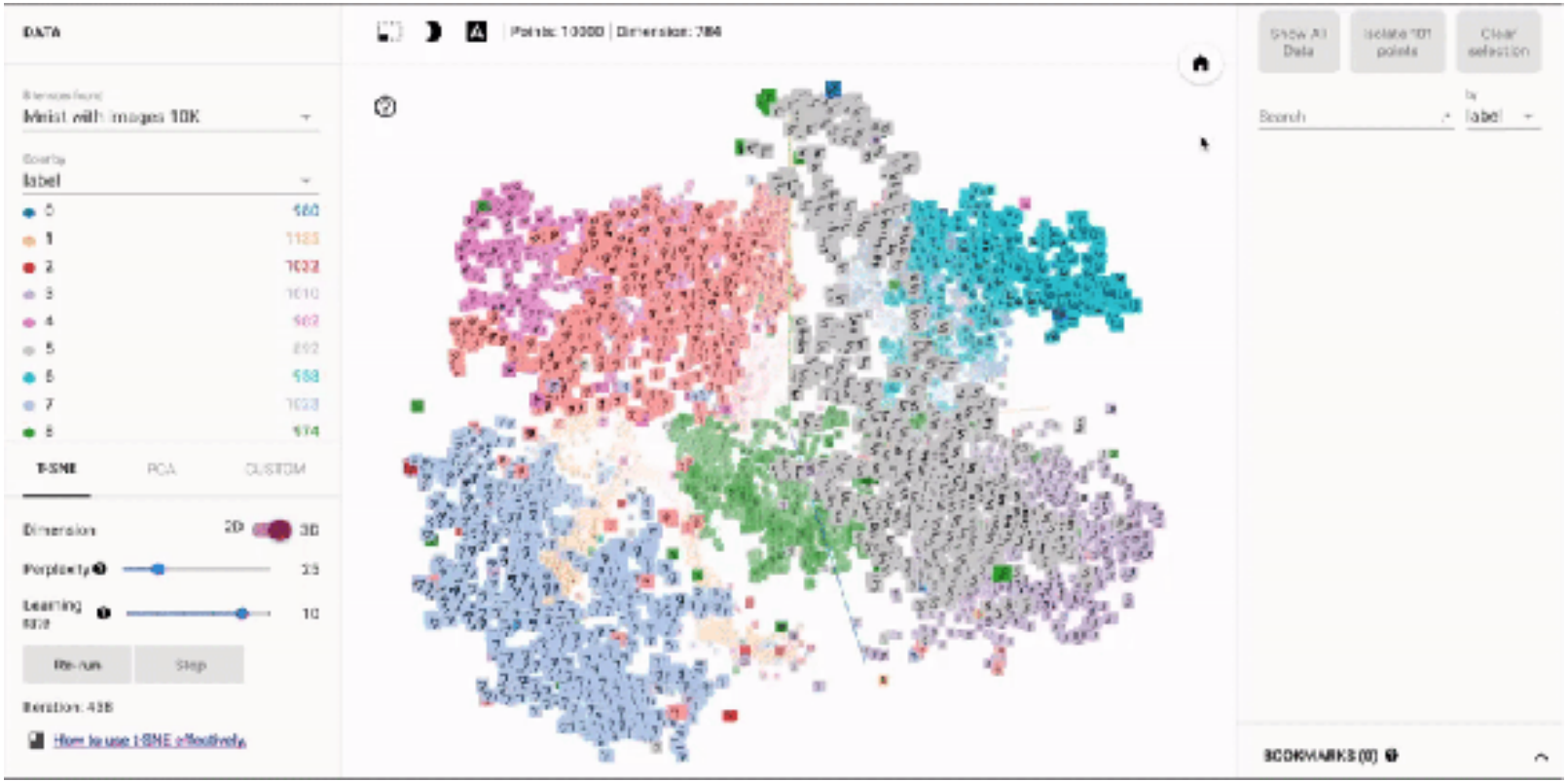
C#

julia

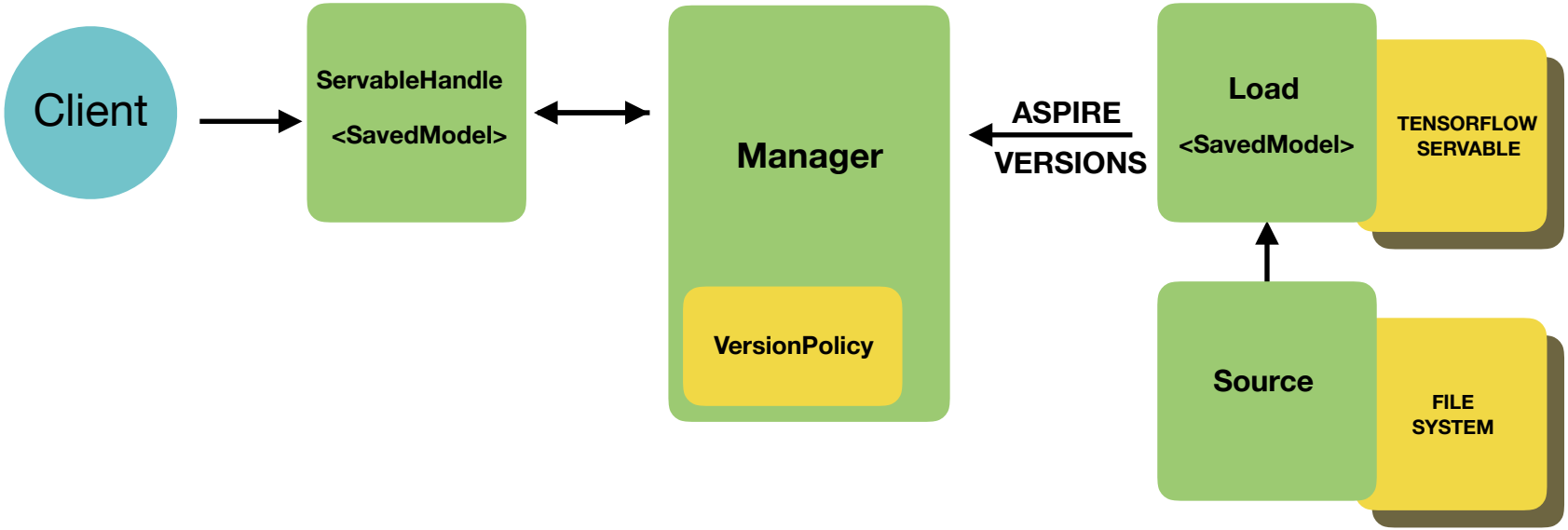
Go

Haskell

TensorFlow provides great tools like TensorBoard



TensorFlow Serving

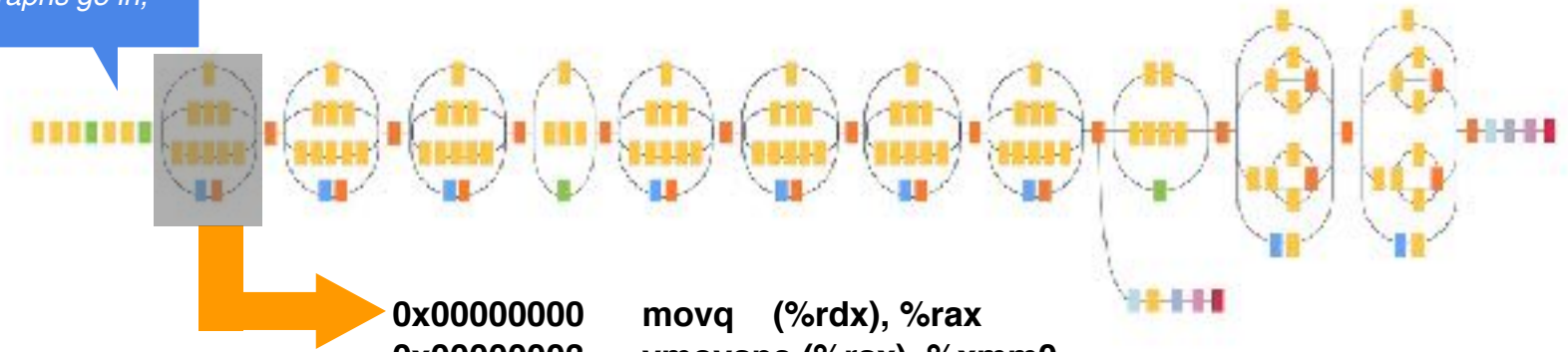


Legend:

- **Blue:** Caller-specific code
- **Green:** TensorFlow Serving core classes / APIs
- **Yellow:** Pluggable implementations

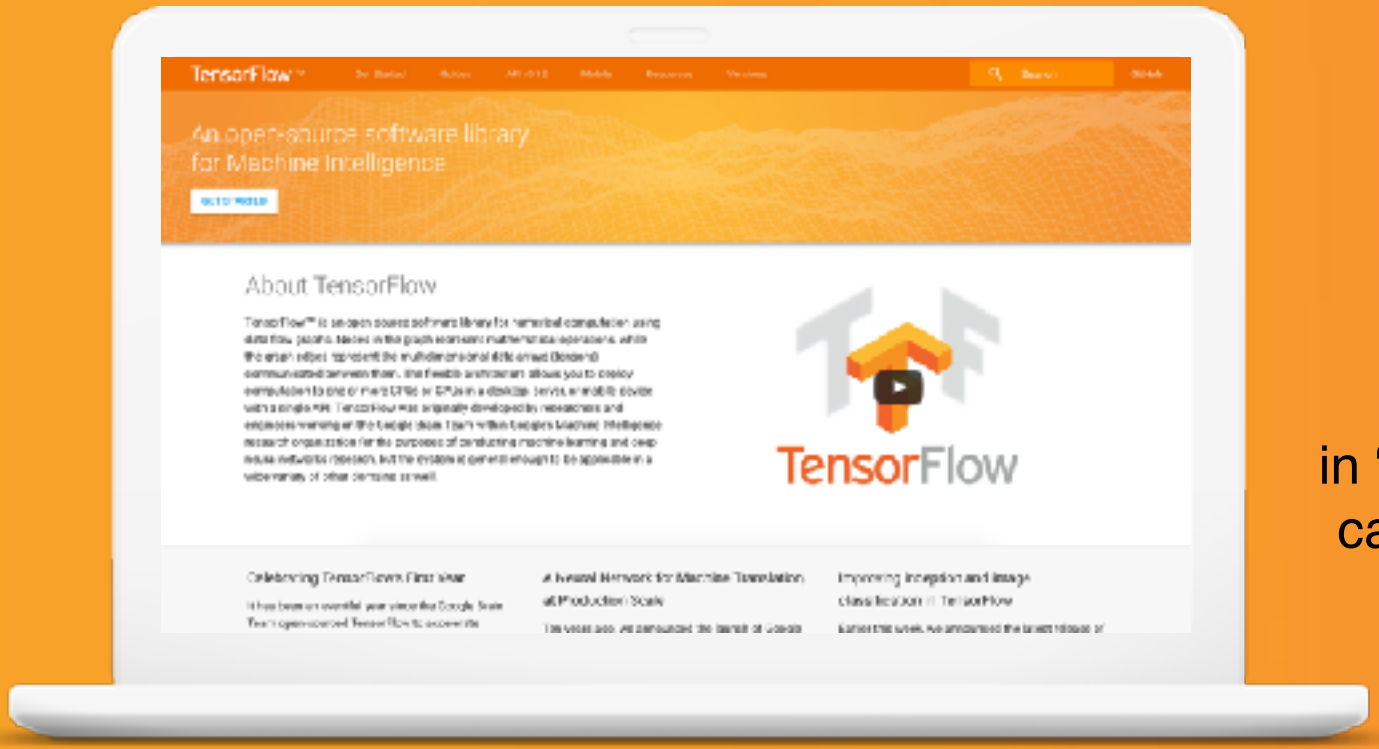
XLA: An Experimental TensorFlow Compiler

TF graphs go in,



```
0x00000000    movq    (%rdx), %rax
0x00000003    vmovaps (%rax), %xmm0
0x00000007    vmulps  %xmm0, %xmm0,
%xmm0
0x0000000b    vmovaps %xmm0, (%rdi)
                      ...
```

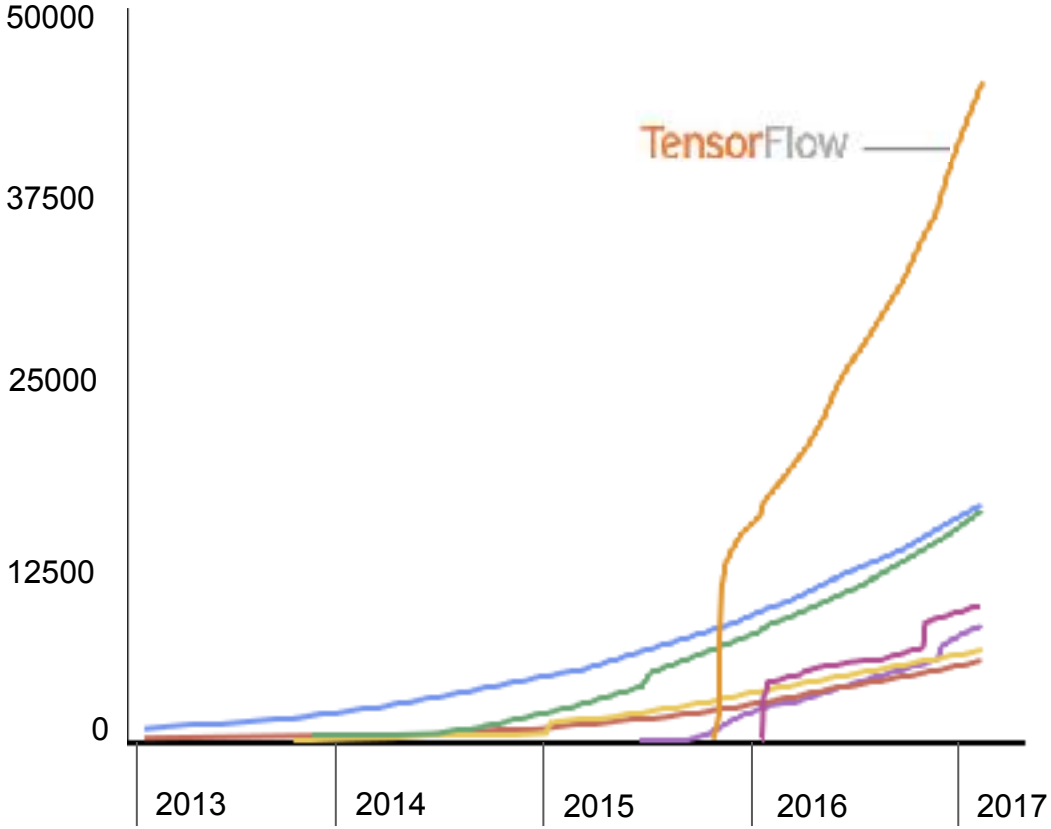
Optimized & specialized assembly comes out.



#1

repository

in “machine learning”
category on GitHub



Github Stars

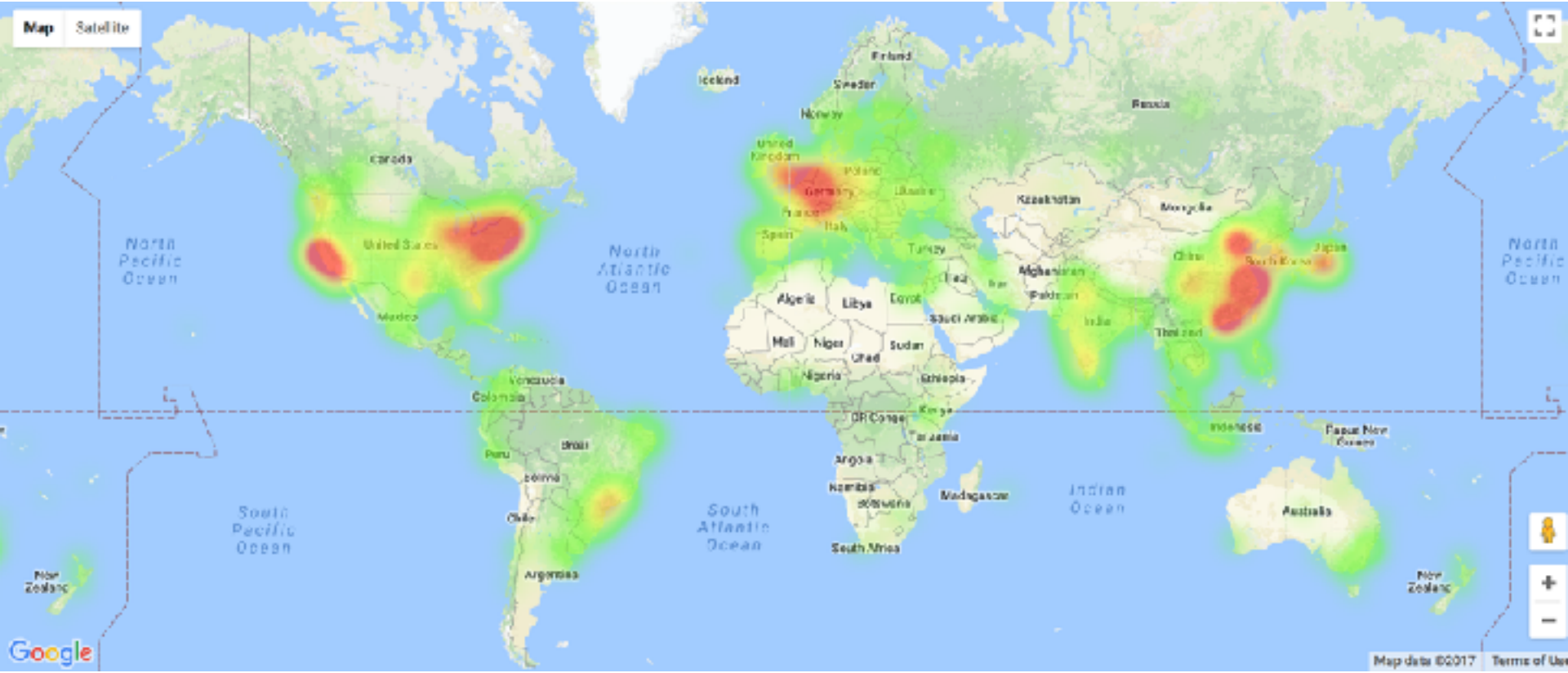
TensorFlow: A Vibrant Open-Source Community

- **Rapid development, many outside contributors**
  - 500+ non-Google contributors
  - 20,000+ commits
  - Many community created tutorials, models, translations, and projects
    - ~5,500 GitHub repositories with ‘TensorFlow’ in the title

- **Direct engagement between community and TensorFlow team**
  - 5000+ Stack Overflow questions answered
  - 5000+ GitHub issues filed and answered; 160+ new issues / week

- **Use in ML classes is growing: Toronto, Berkeley, Stanford, ...**

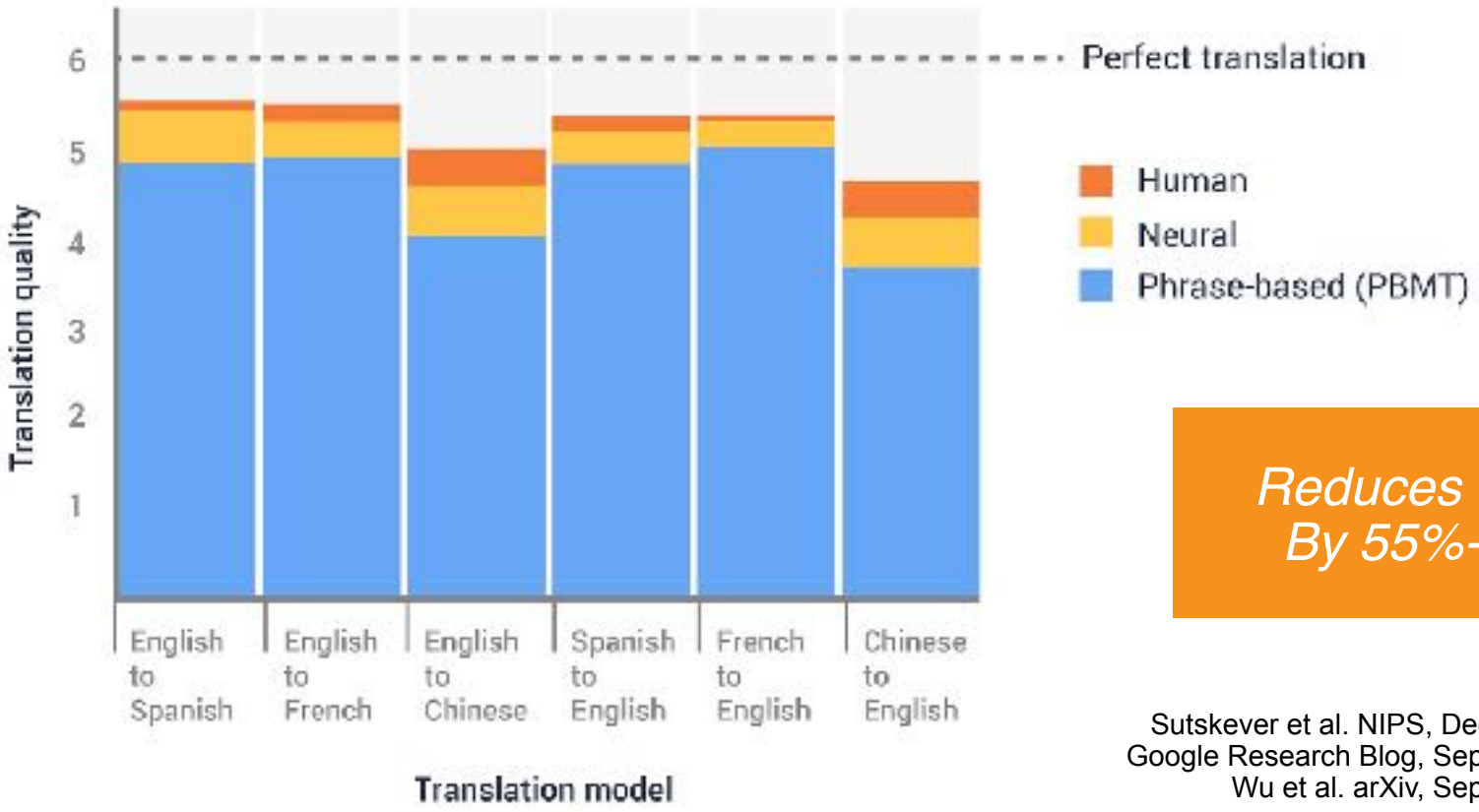
Use of TensorFlow Worldwide



Use in South East Asia

What is TensorFlow being used for?

Neural Machine Translation



Reduces Errors By 55%-85%

Sutskever et al. NIPS, Dec 2014
Google Research Blog, Sept 2016
Wu et al. arXiv, Sept 2016

Show and Tell

https://research.googleblog.com/2016/09/show-and-tell-image-captioning-open.html

Mobile Deployment

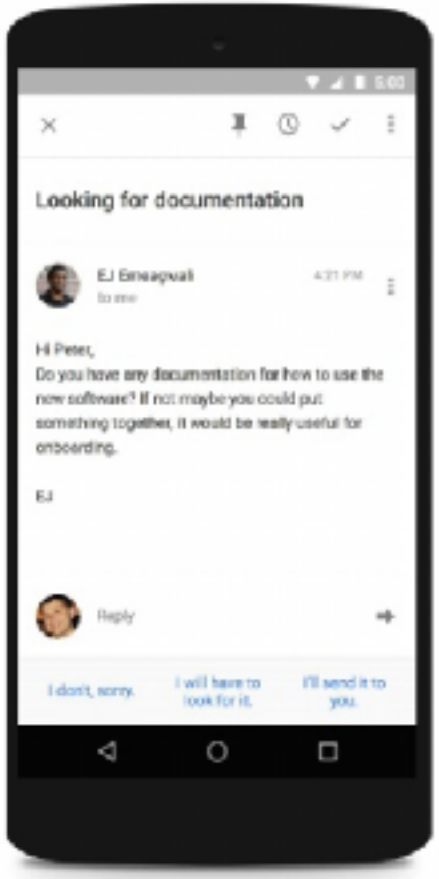
Inbox by Gmail

Smart Reply

April 1, 2009: April Fool's Day joke

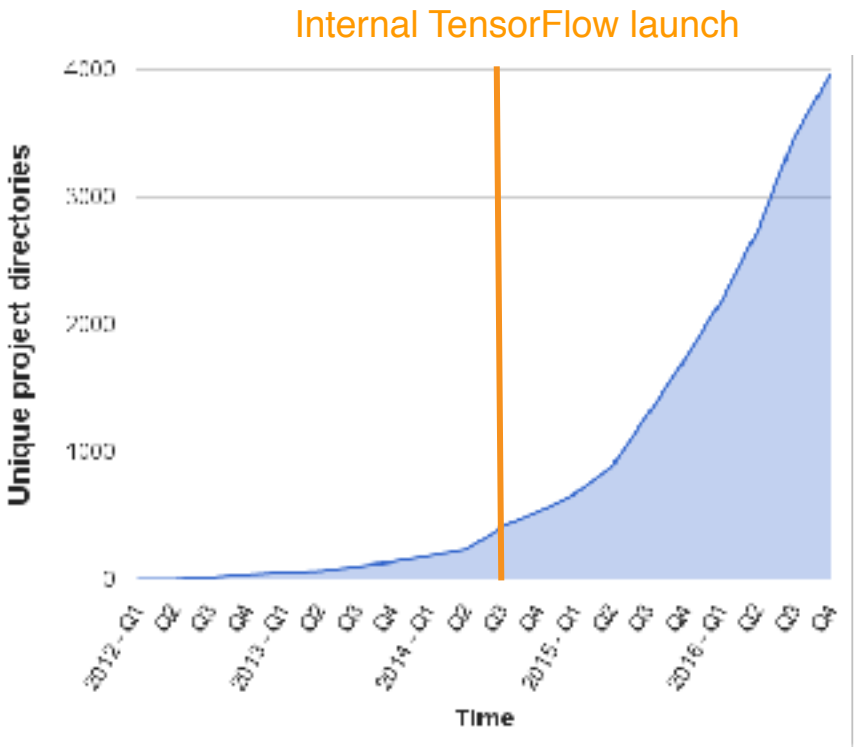
Nov 5, 2015: Launched Real Product

Feb 1, 2016: >10% of mobile Inbox replies



Research at Google

of Google directories containing model description files

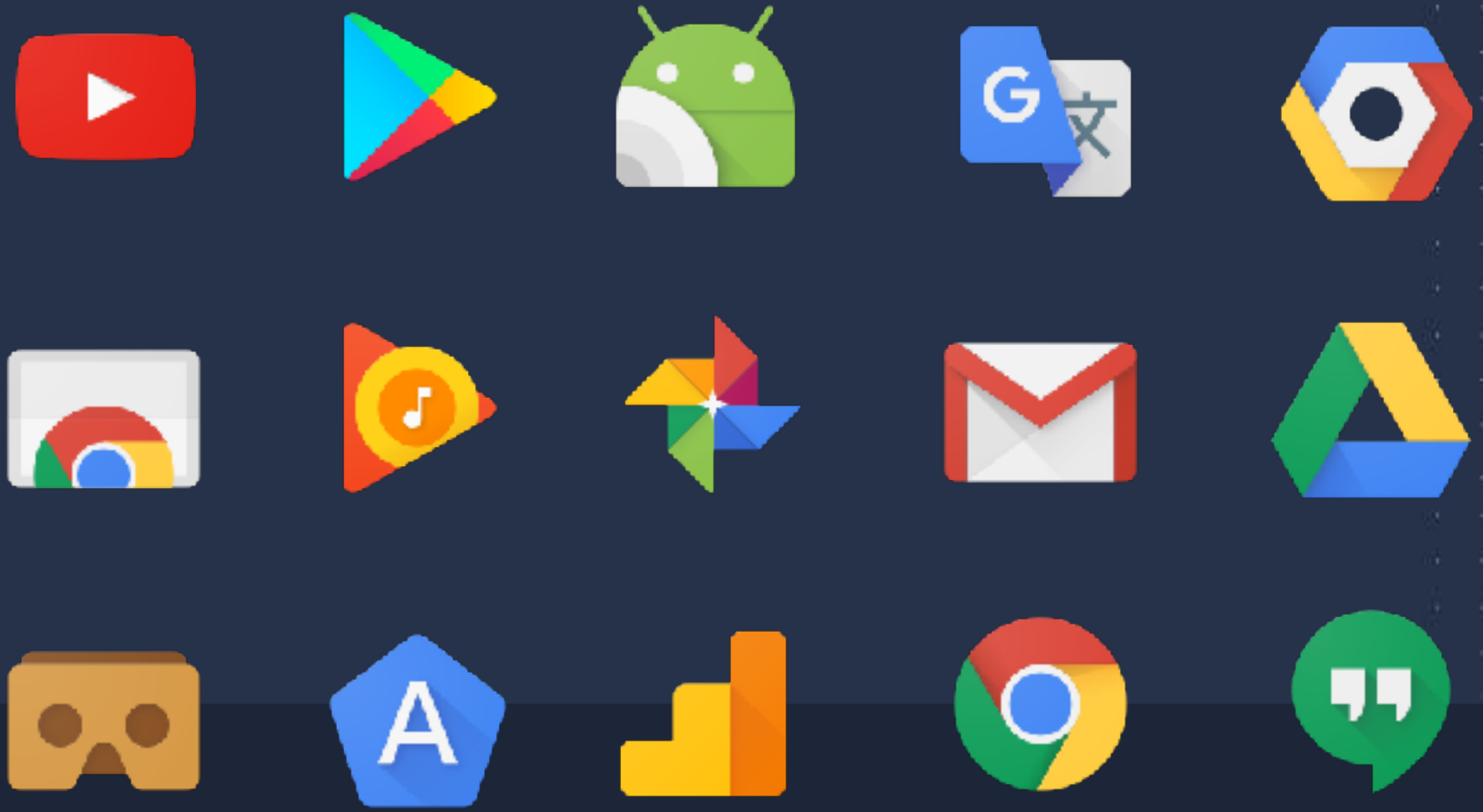


Production use in many areas:

Search
Gmail
Translate
Maps
Android
Photos
Speech
YouTube
Play
… many others ...

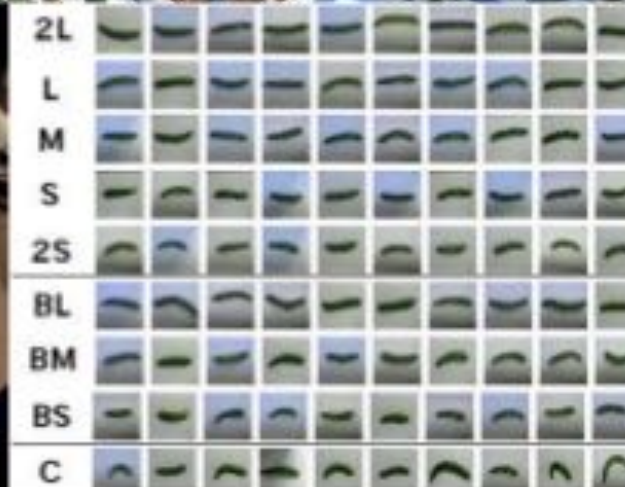
Research use for:

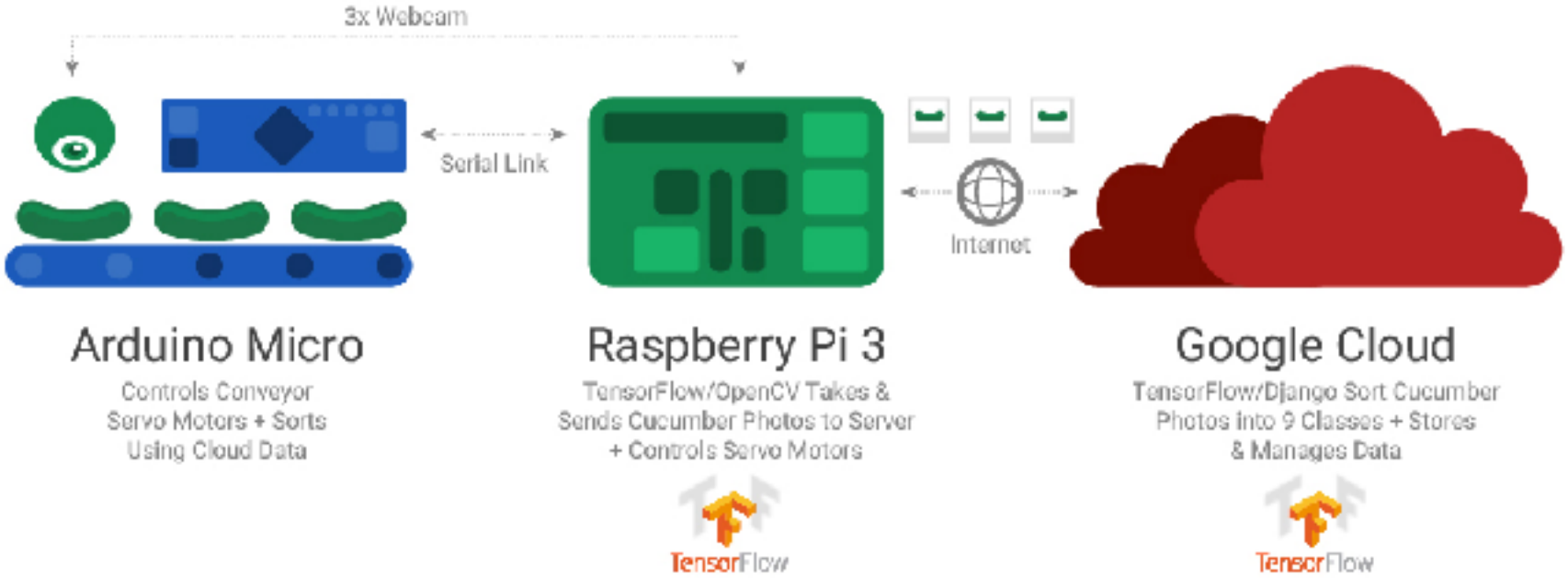
100s of projects and papers



TensorFlow is for **everyone**

TensorFlow powered Cucumber Sorter



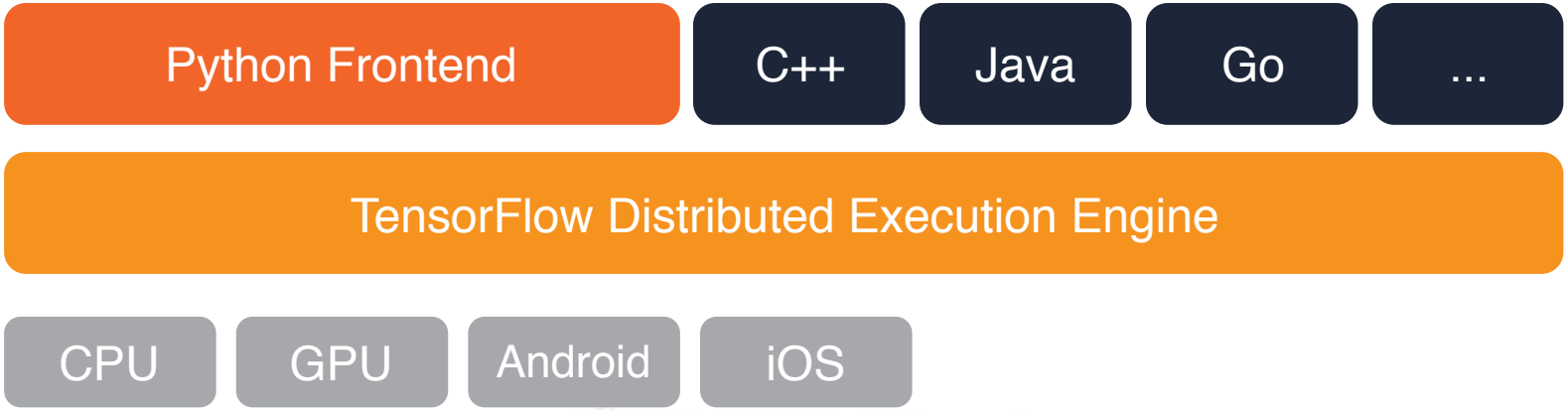


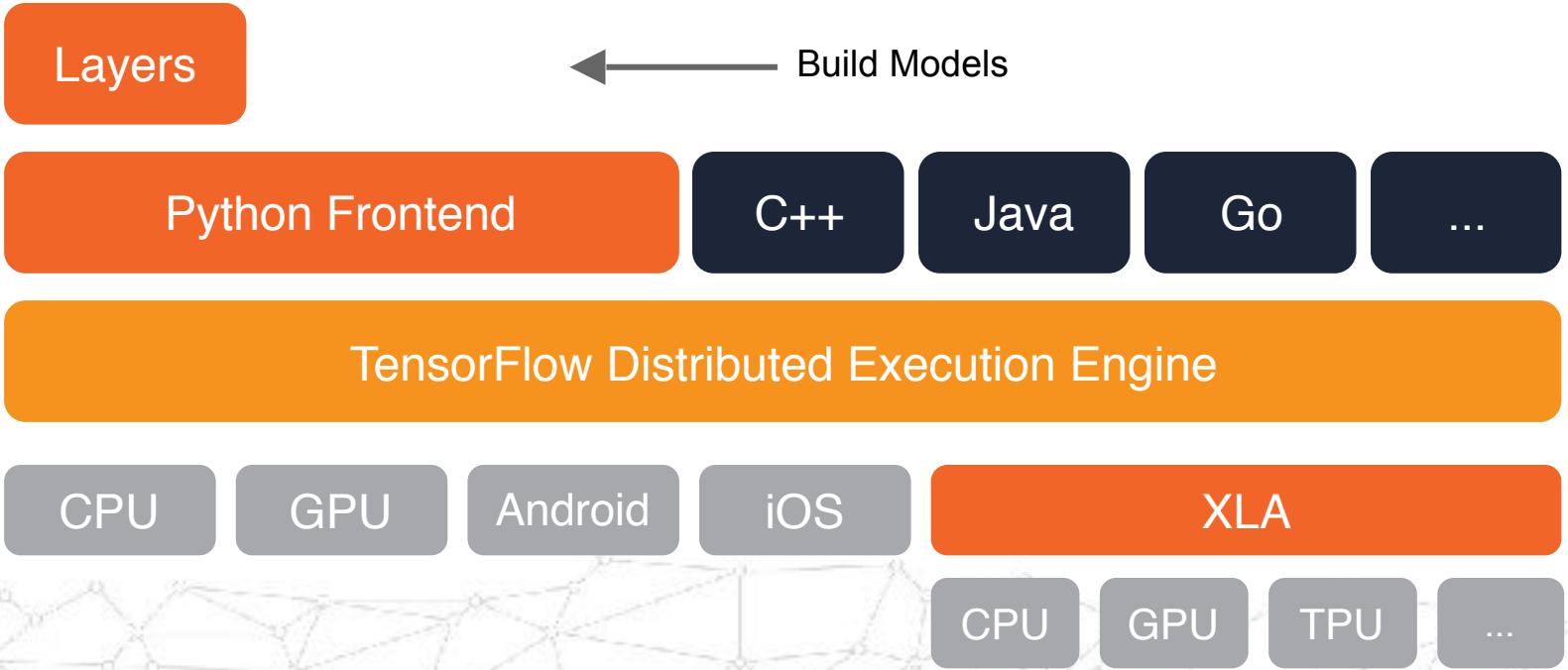
https://cloud.google.com/blog/big-data/2016/08/how-a-japanese-cucumber-farmer-is-using-deep-learning-and-tensorflow

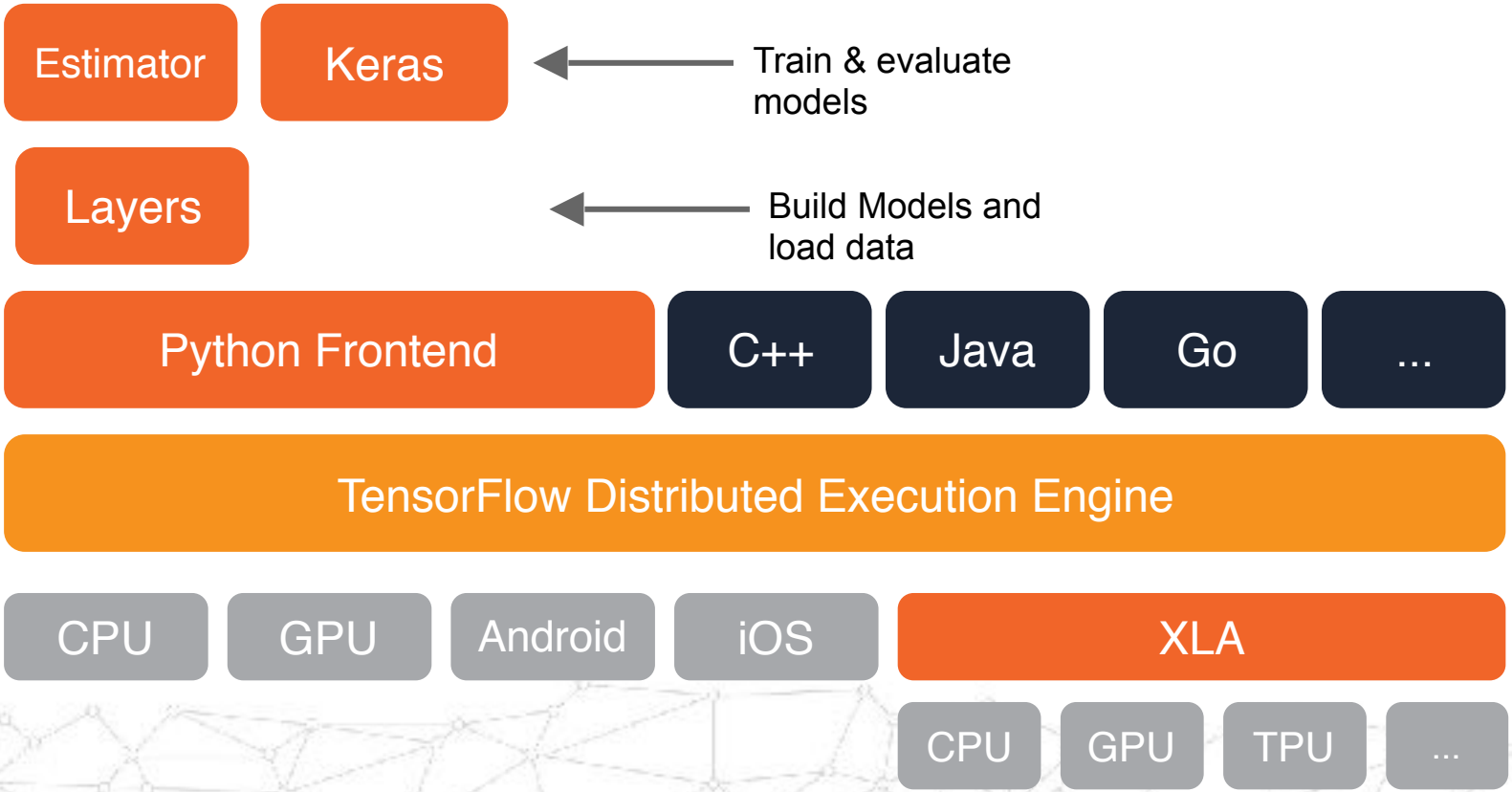
Getting Started

Key parts of the TensorFlow API

TensorFlow pre v1.0

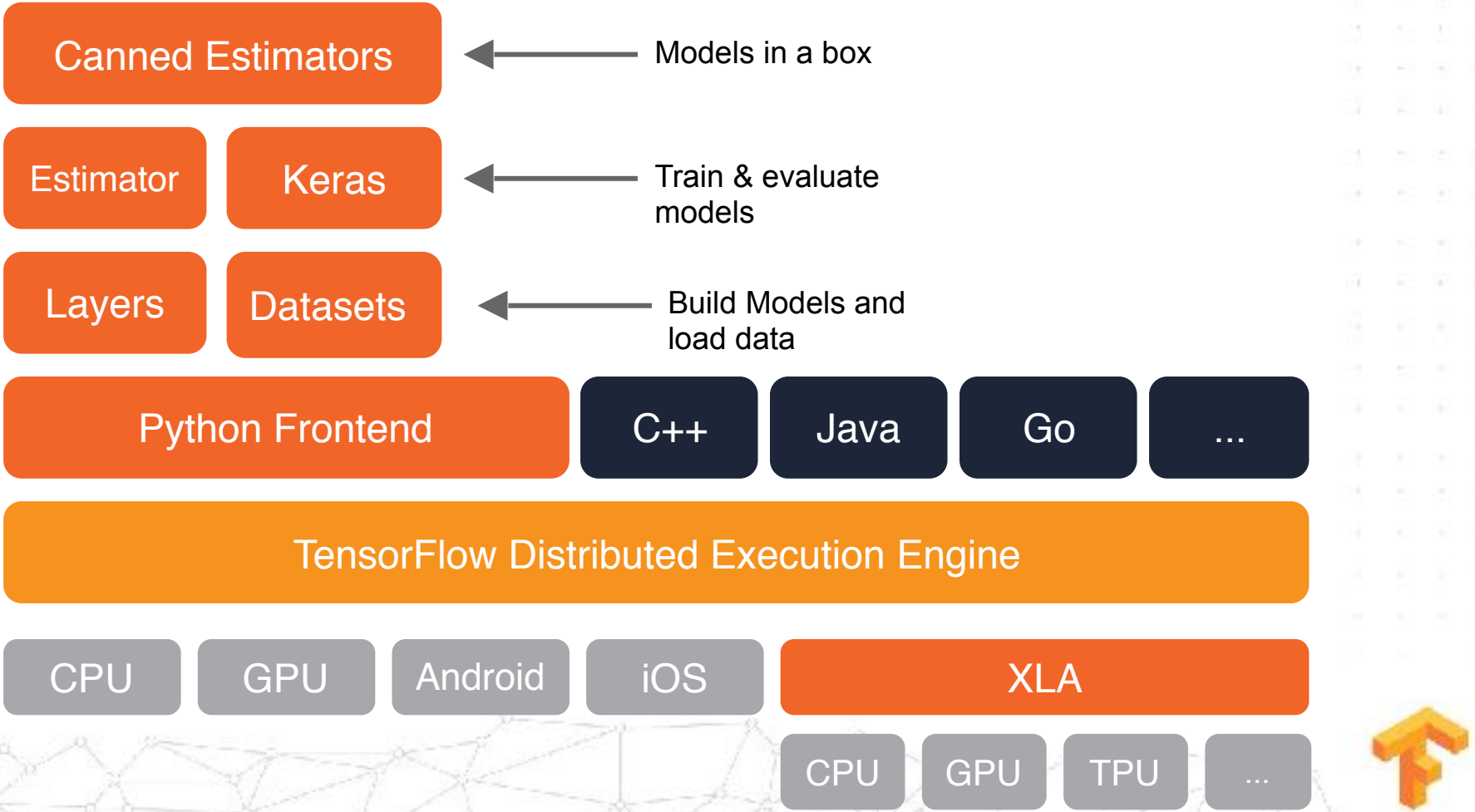

Python Frontend
C++
Java
Go
...
TensorFlow Distributed Execution Engine
CPU
GPU
Android
iOS


Layers
Build Models
Python Frontend
C++
Java
Go
...
TensorFlow Distributed Execution Engine
CPU
GPU
Android
iOS
XLA
CPU
GPU
TPU
...

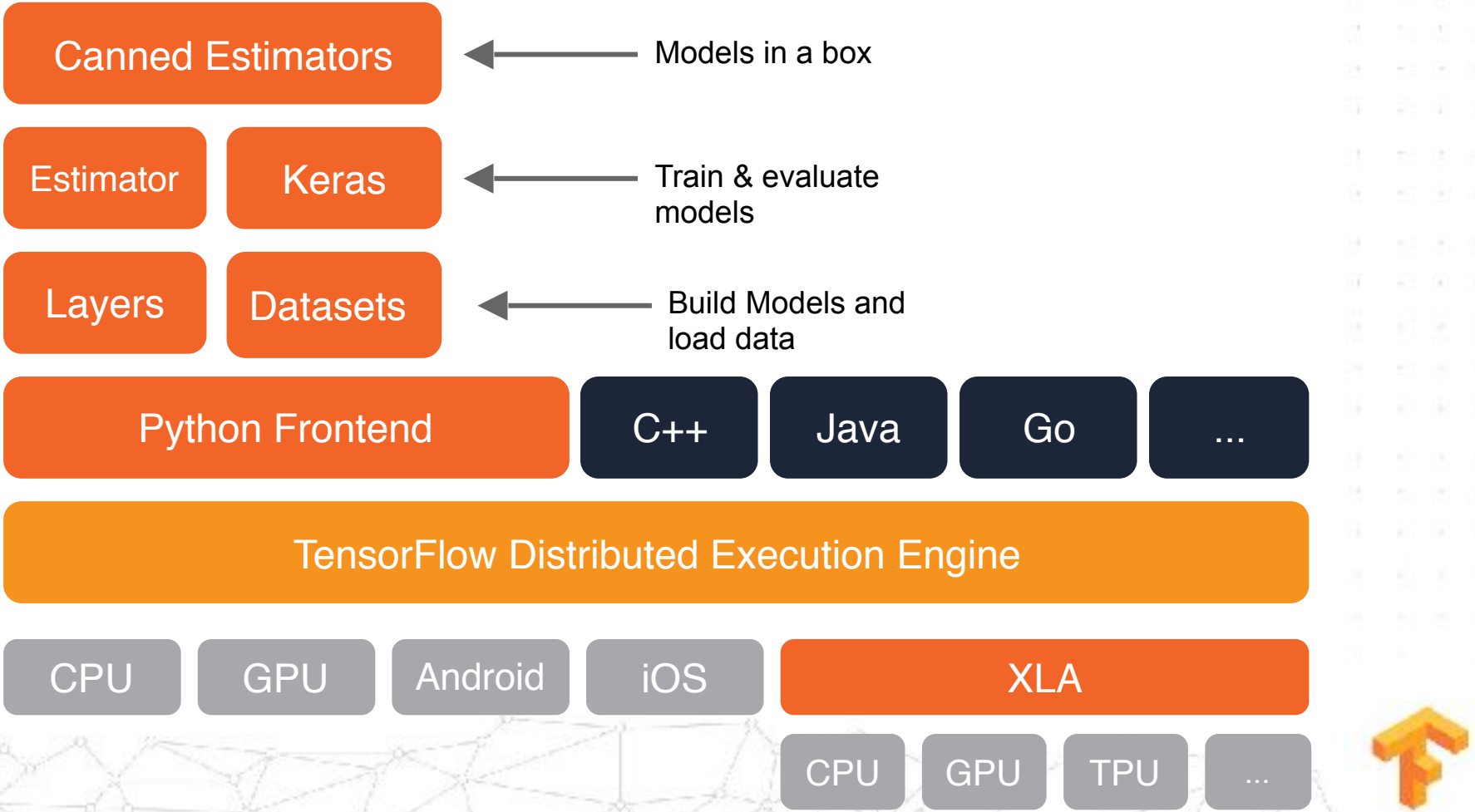

Estimator
Keras
Train & evaluate models
Layers
Build Models and load data
Python Frontend
C++
Java
Go
...
TensorFlow Distributed Execution Engine
CPU
GPU
Android
iOS
XLA
CPU
GPU
TPU
...

TensorFlow 1.4

open source machine learning


Canned Estimators
Models in a box
Estimator
Keras
Train & evaluate models
Layers
Datasets
Build Models and load data
Python Frontend
C++
Java
Go
...
TensorFlow Distributed Execution Engine
CPU
GPU
Android
iOS
XLA
CPU
GPU
TPU
...

TensorFlow v1.4


Canned Estimators
Models in a box
Estimator
Keras
Train & evaluate models
Layers
Datasets
Build Models and load data
Python Frontend
C++
Java
Go
...
TensorFlow Distributed Execution Engine
CPU
GPU
Android
iOS
XLA
CPU
GPU
TPU
...

Flexible: High level APIs

Low-Level Python API

```
import numpy as np
import tensorflow as tf

W = tf.Variable([.3], tf.float32)
b = tf.Variable([-.3], tf.float32)
x = tf.placeholder(tf.float32)
linear_model = W * x + b
y = tf.placeholder(tf.float32)
loss = tf.reduce_sum(tf.square(linear_model - y))
optimizer = tf.train.GradientDescentOptimizer(0.01)
train = optimizer.minimize(loss)
x_train = [1,2,3,4]
y_train = [0,-1,-2,-3]
init = tf.global_variables_initializer()
sess = tf.Session()
sess.run(init) # reset values to wrong
for i in range(1000):
  sess.run(train, {x:x_train, y:y_train})
curr_W, curr_b, curr_loss  = sess.run([W, b, loss], {x:x_train, y:y_train})
```

High-Level Python API

```
import tensorflow as tf
import numpy as np

features = [tf.contrib.layers.real_valued_column("x", dimension=1)]
estimator = tf.contrib.learn.LinearRegressor(feature_columns=features)
x = np.array([1., 2., 3., 4.])
y = np.array([0., -1., -2., -3.])
input_fn = tf.contrib.learn.io.numpy_input_fn({"x":x}, y, batch_size=4,
                                              num_epochs=1000)
estimator.fit(input_fn=input_fn, steps=1000)
estimator.evaluate(input_fn=input_fn)
```

https://www.tensorflow.org/get_started/get_started

Flexible: High level APIs

Low-Level Python API

```
import numpy as np
import tensorflow as tf

W
b =
x =
lin
y =
los
op
tra
x_train = [1,2,3,4]
y_train = [0,-1,-2,-3]
init = tf.global_variables_initializer()
sess = tf.Session()
sess.run(init) # reset values to wrong
for i in range(1000):
  sess.run(train, {x:x_train, y:y_train})
curr_W, curr_b, curr_loss  = sess.run([W, b, loss], {x:x_train, y:y_train})
```

15 lines

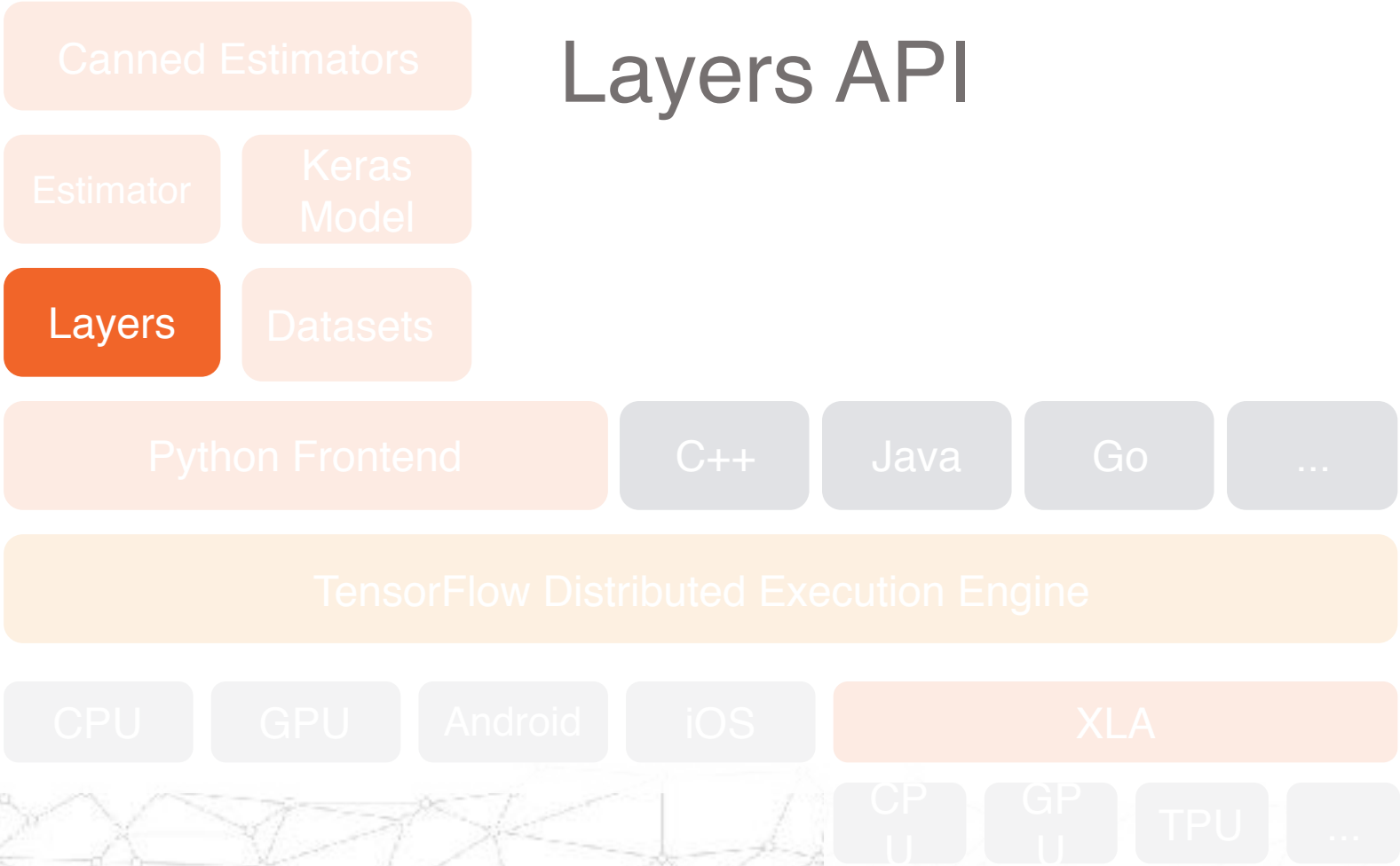
High-Level Python API

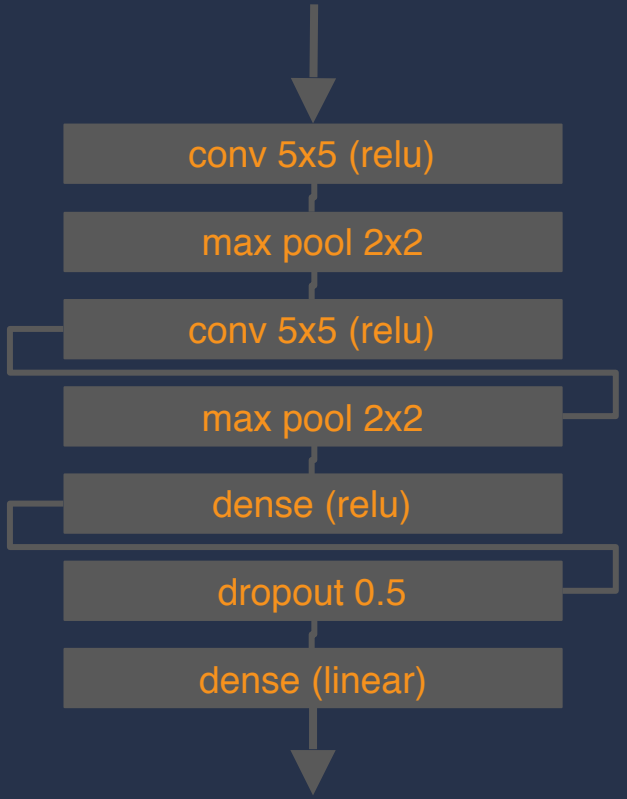
```
import tensorflow as tf
import numpy as n

features = [tf.contri
estimator = tf.contr
x = np.array([1., 2.
y = np.array([0., -1
input_fn = tf.contrib

estimator.fit(input_
estimator.evaluate(input_fn=input_fn)
```

7 lines

https://www.tensorflow.org/get_started/get_started

Layers API

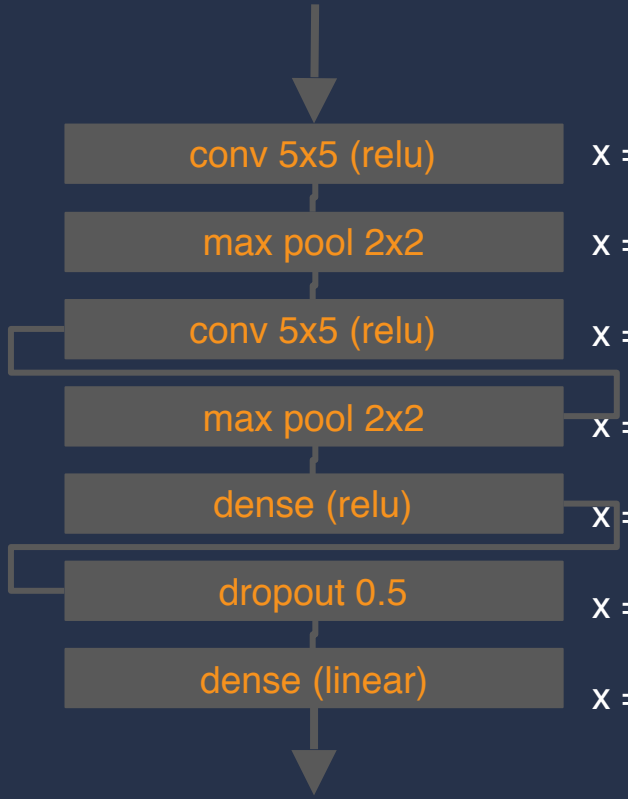

Canned Estimators
Estimator
Keras Model
Layers
Datasets
Python Frontend
C++
Java
Go
...
TensorFlow Distributed Execution Engine
CPU
GPU
Android
iOS
XLA
CPU
GPU
TPU
...


conv 5x5 (relu)
max pool 2x2
conv 5x5 (relu)
max pool 2x2
dense (relu)
dropout 0.5
dense (linear)


```
x = tf.layers.conv2d(x, kernel_size=[5,5], ...)
```

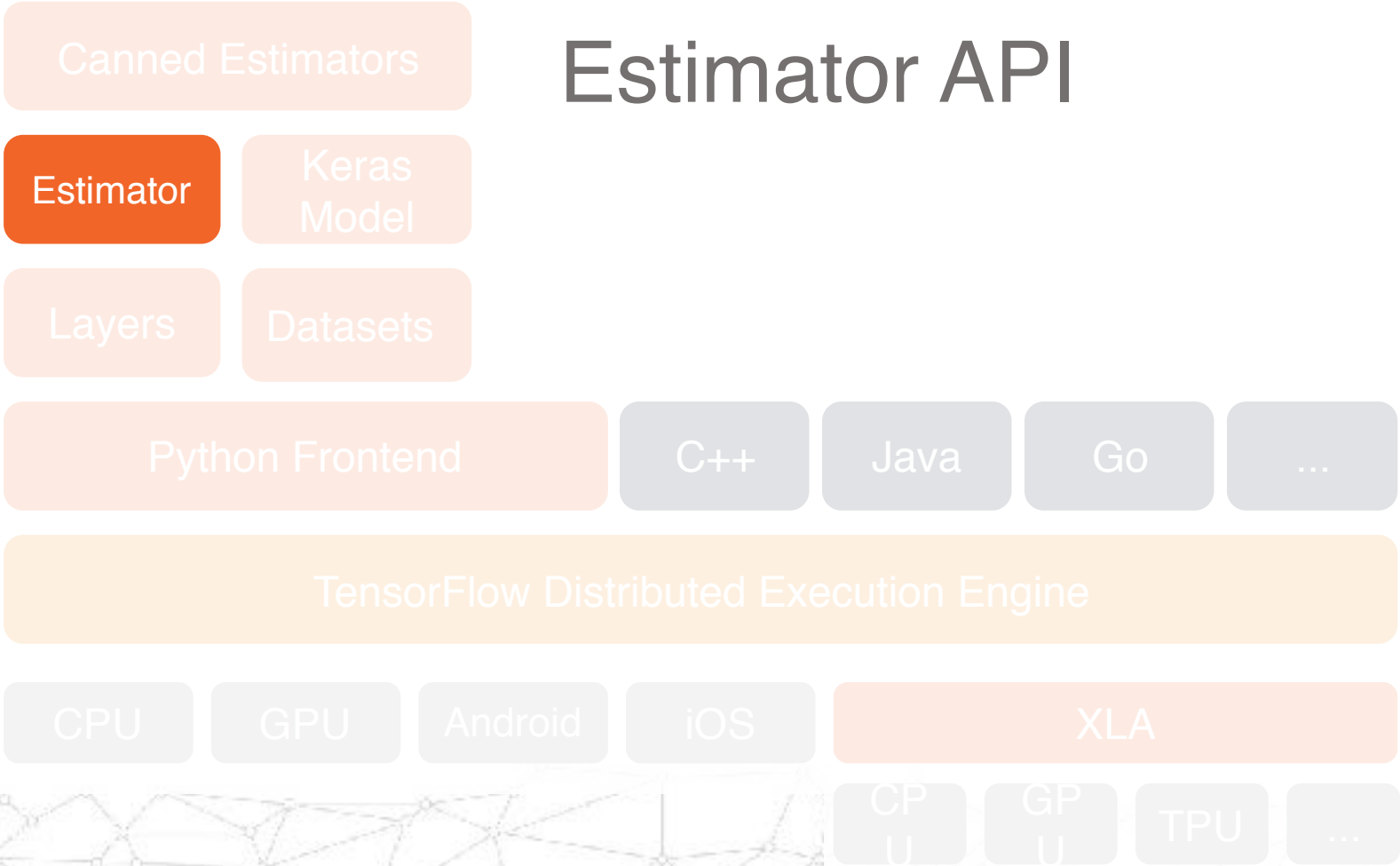

```
x = tf.layers.conv2d(x, kernel_size=[5,5], ...)
```

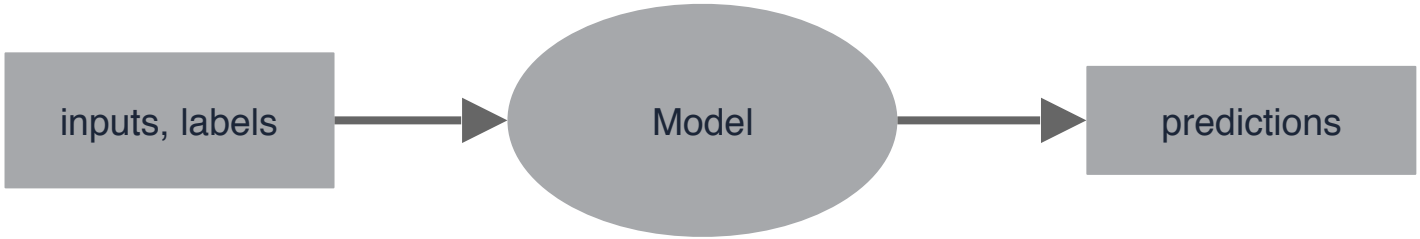
```
x = tf.layers.max_pooling2d(x, kernel_size=[2,2], ...)
```

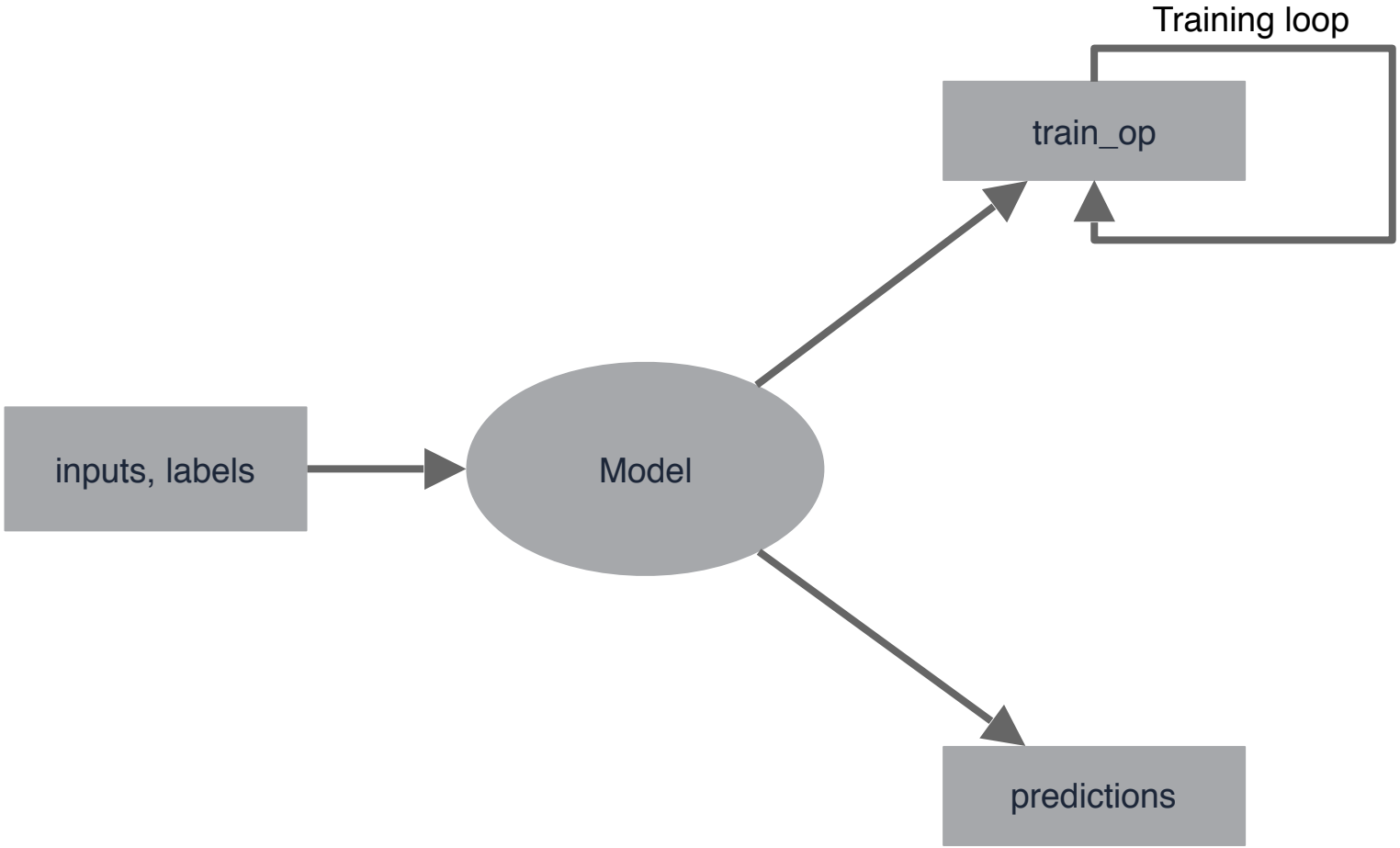


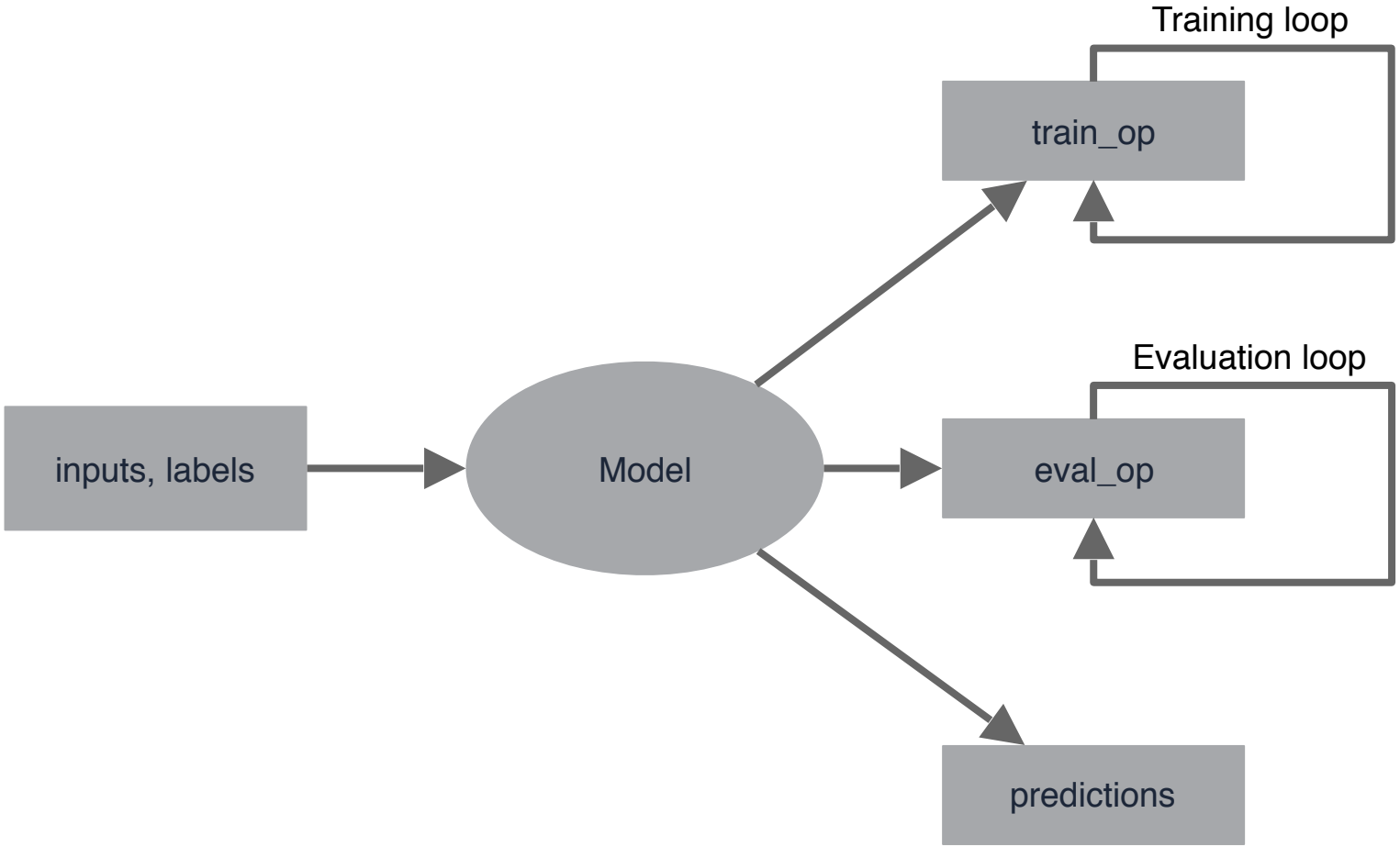

```
x = tf.layers.conv2d(x, kernel_size=[5,5], ...)
x = tf.layers.max_pooling2d(x, kernel_size=[2,2], ...)
x = tf.layers.conv2d(x, kernel_size=[5,5], ...)
x = tf.layers.max_pooling2d(x, kernel_size=[2,2], ...)
x = tf.layers.dense(x, activation_fn=tf.nn.relu)
x = tf.layers.dropout(x, 0.5)
x = tf.layers.dense(x)
```

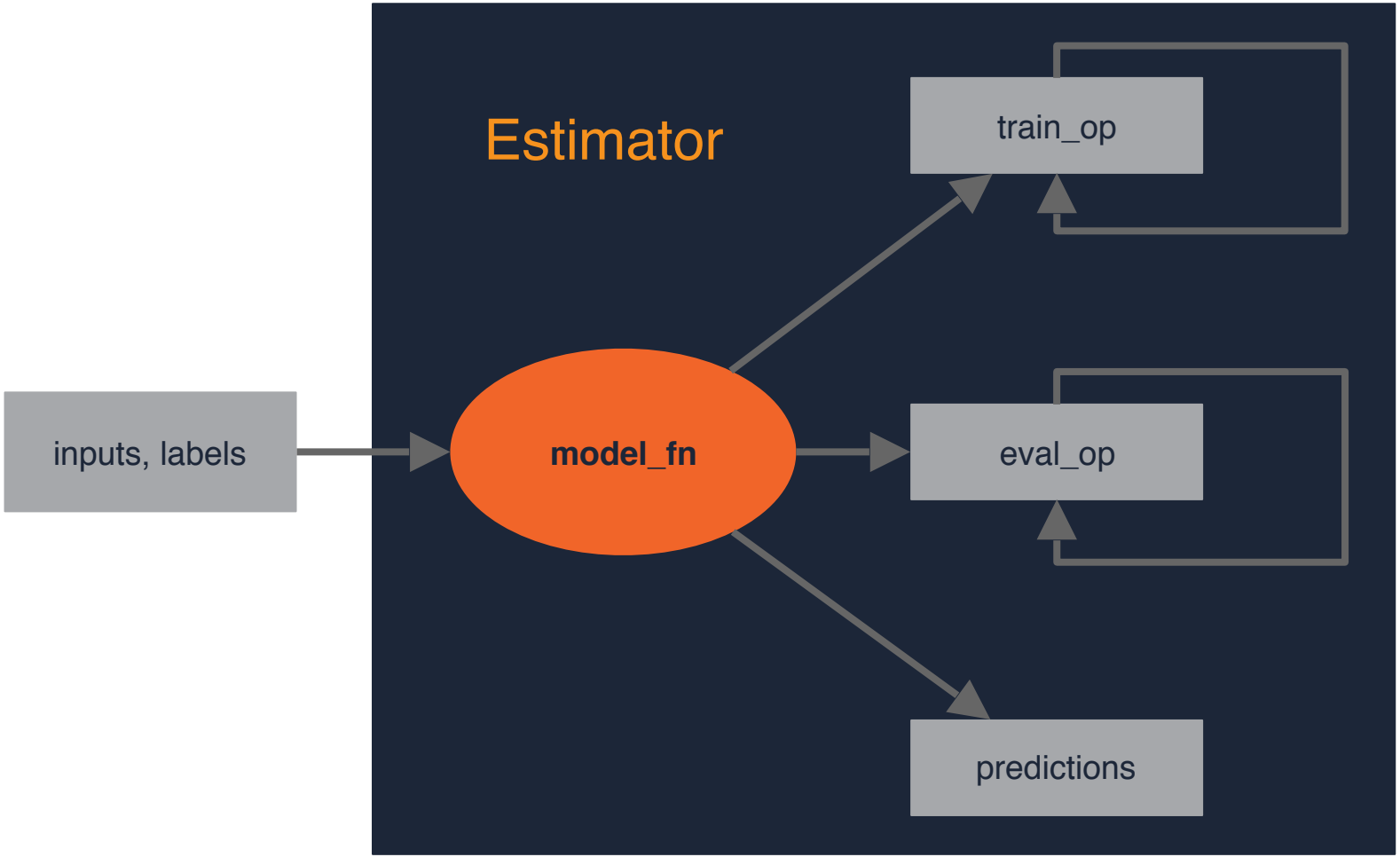
Estimator API

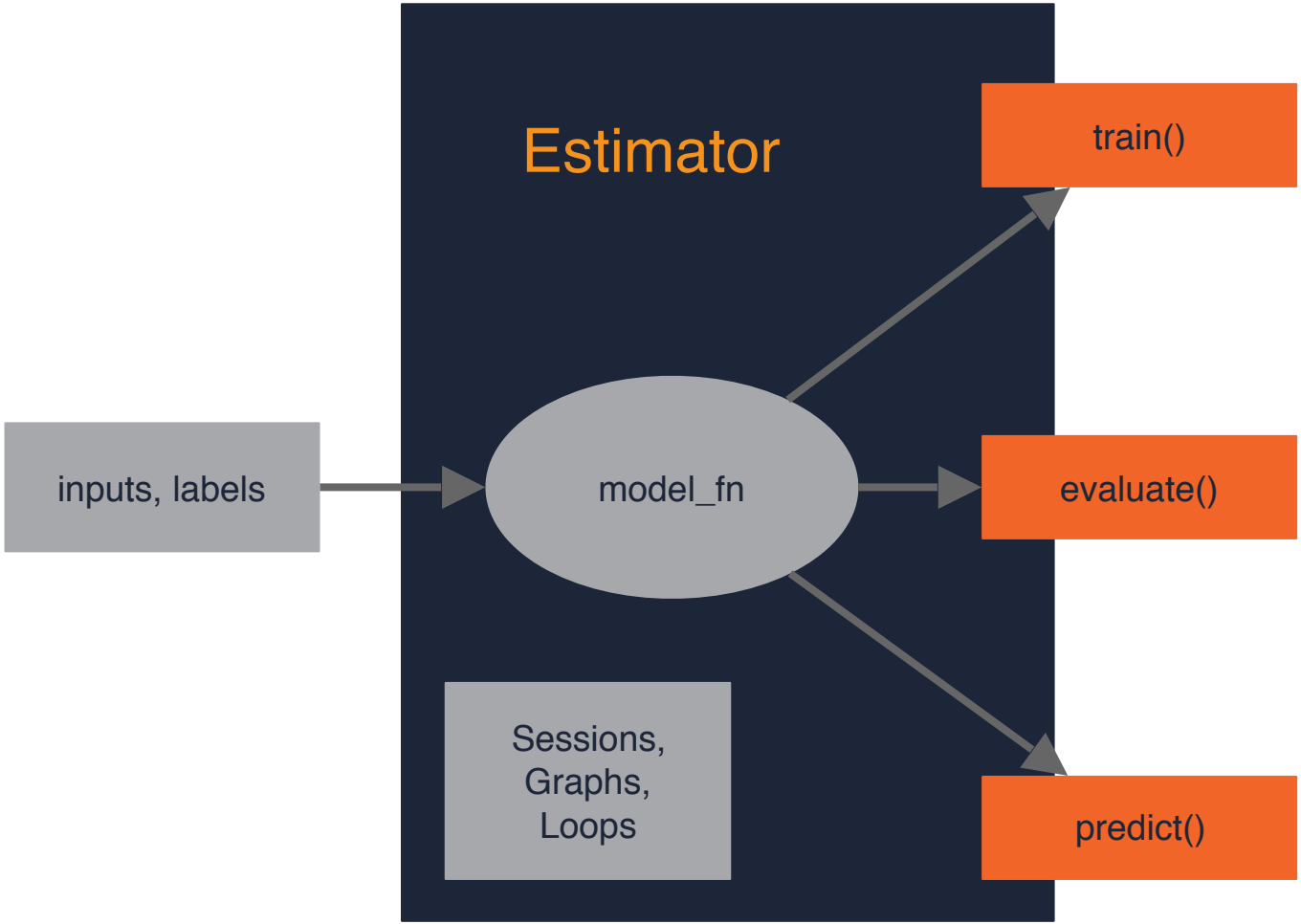


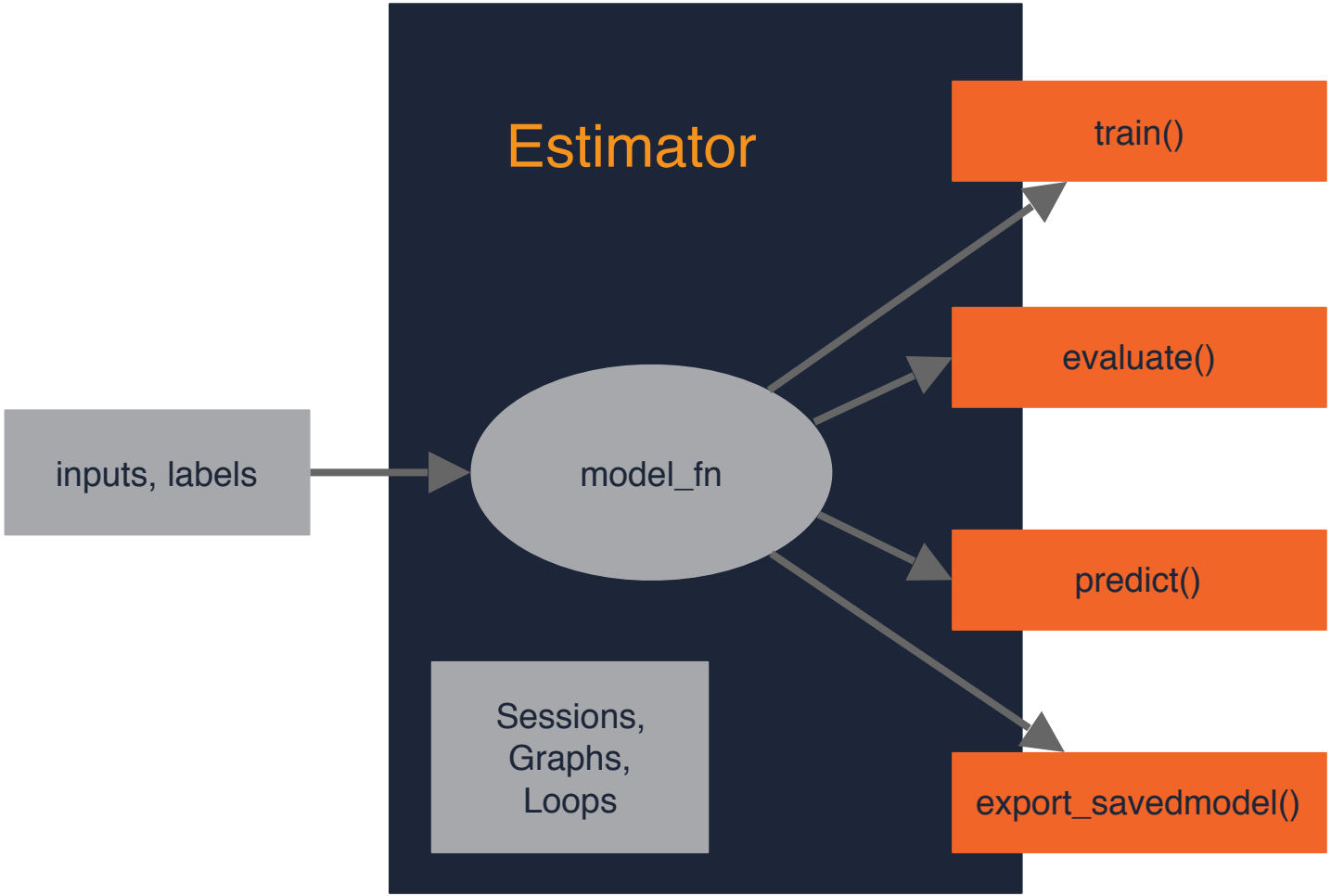

inputs, labels
Model
predictions


Training loop
train_op
inputs, labels
Model
predictions


Training loop
train_op
Evaluation loop
inputs, labels
Model
eval_op
predictions


Estimator
train_op
inputs, labels
model_fn
eval_op
predictions

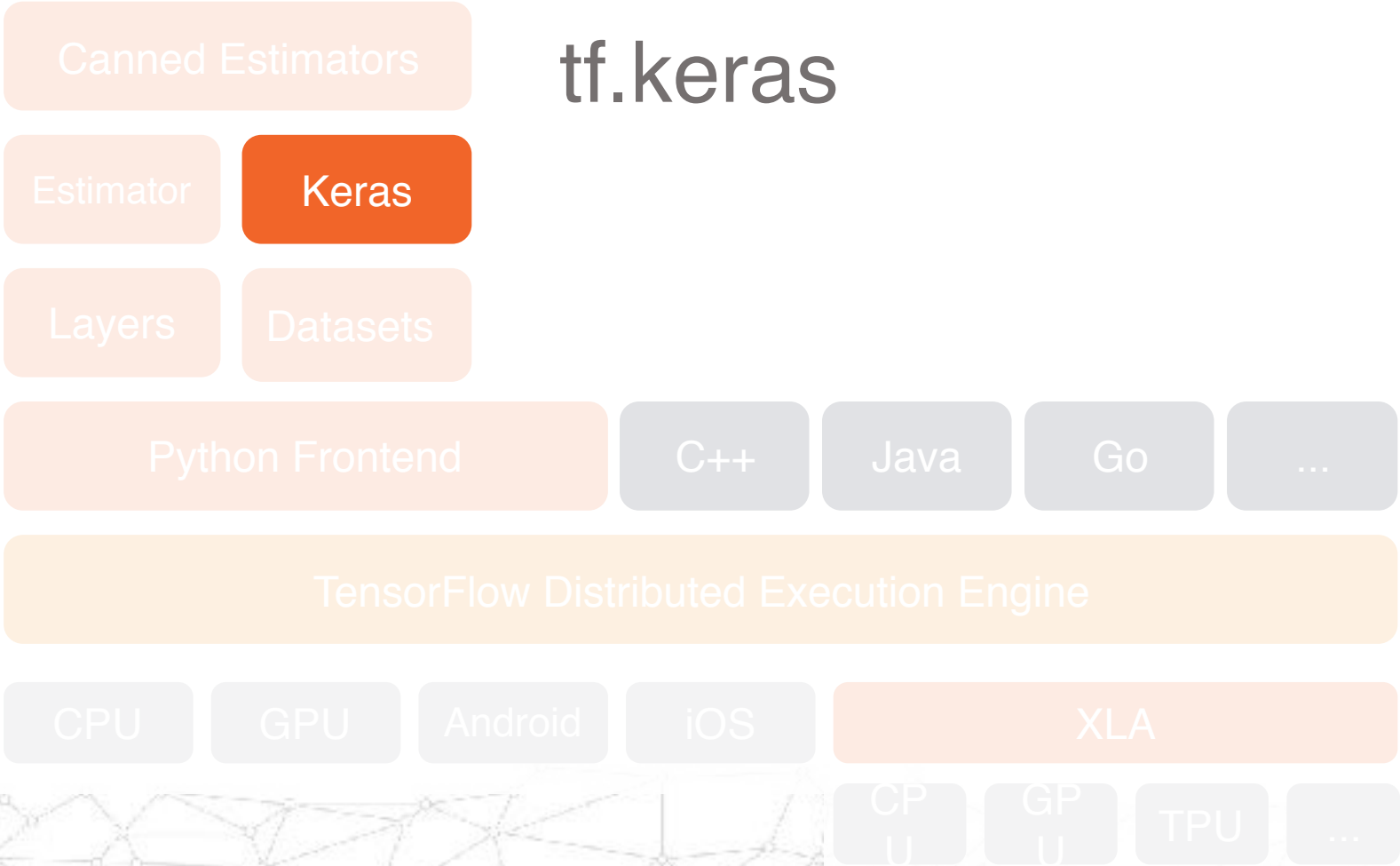

Estimator
train()
inputs, labels
model_fn
evaluate()
Sessions,
Graphs,
Loops
predict()


Estimator
train()
evaluate()
predict()
export_savedmodel()
inputs, labels
model_fn
Sessions,
Graphs,
Loops

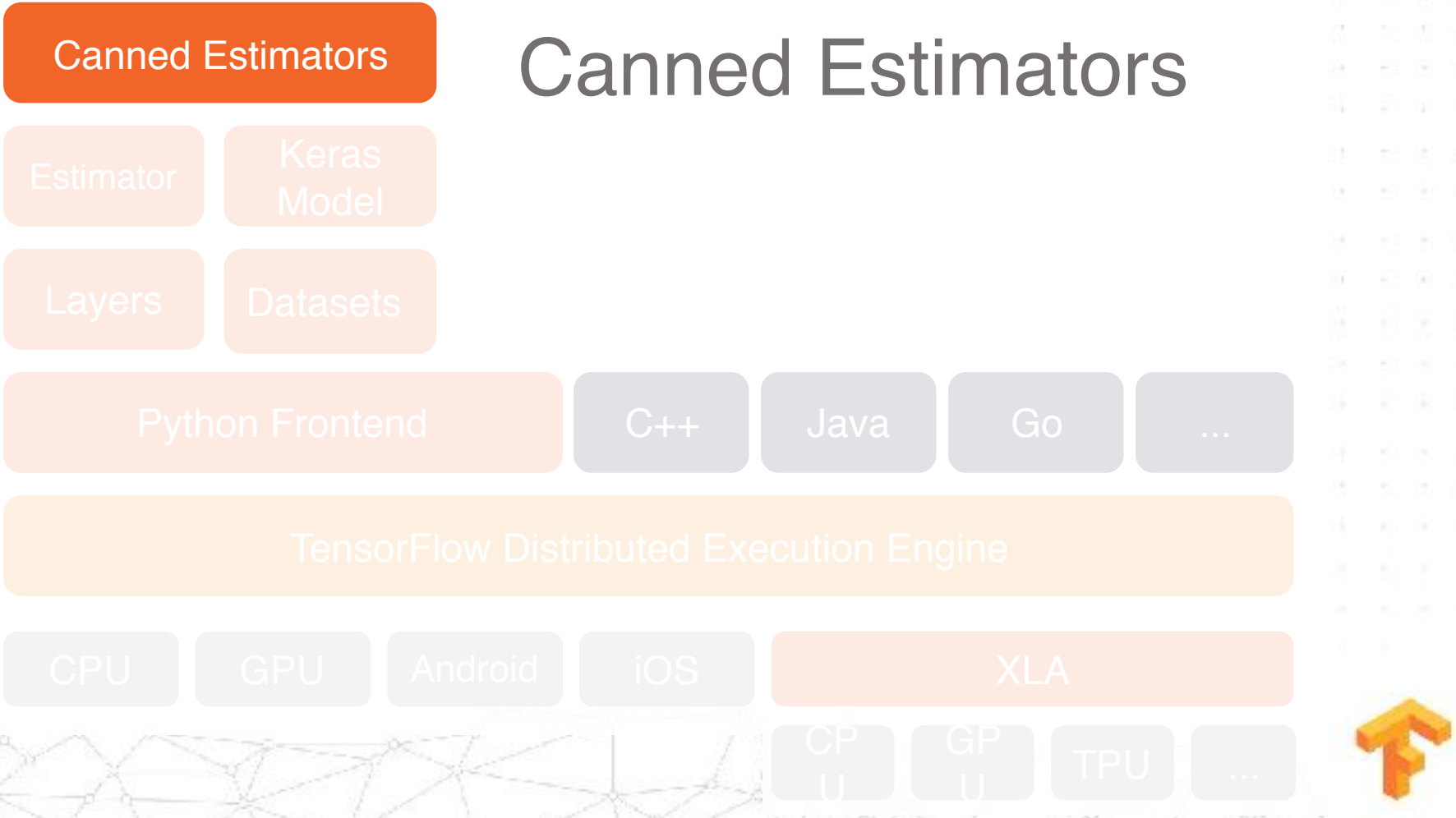
Custom Estimators

- Input_fn
- model_fn
- train
- evaluate
- export_savedmodel

tf.keras


Canned Estimators
Estimator
Keras
Layers
Datasets
Python Frontend
C++
Java
Go
...
TensorFlow Distributed Execution Engine
CPU
GPU
Android
iOS
XLA
CPU
GPU
TPU
...

```
video = tf.keras.layers.Input(shape=(None, 150, 150, 3))
cnn = tf.keras.applications.InceptionV3(weights='imagenet',
                                        include_top=False,
                                        pool='avg')

cnn.trainable = False
```


Canned Estimators
Canned Estimators
Estimator
Keras Model
Layers
Datasets
Python Frontend
C++
Java
Go
...
TensorFlow Distributed Execution Engine
CPU
GPU
Android
iOS
XLA
CPU
GPU
TPU
...

```
area = real_valued_column(“square_foot”),
rooms = real_valued_column(“num_rooms”),
zip_code = sparse_column_with_integerized_feature(“zip_code”, 100000)


regressor = DNNRegressor(
    feature_columns=[area, rooms, embedding_column(zip_code, 8)],
    hidden_units=[1024, 512, 256])

regressor.train(train_input_fn)

regressor.evaluate(eval_input_fn)
```

TensorFlow v1.4

Code time!!

http://playground.tensorflow.org