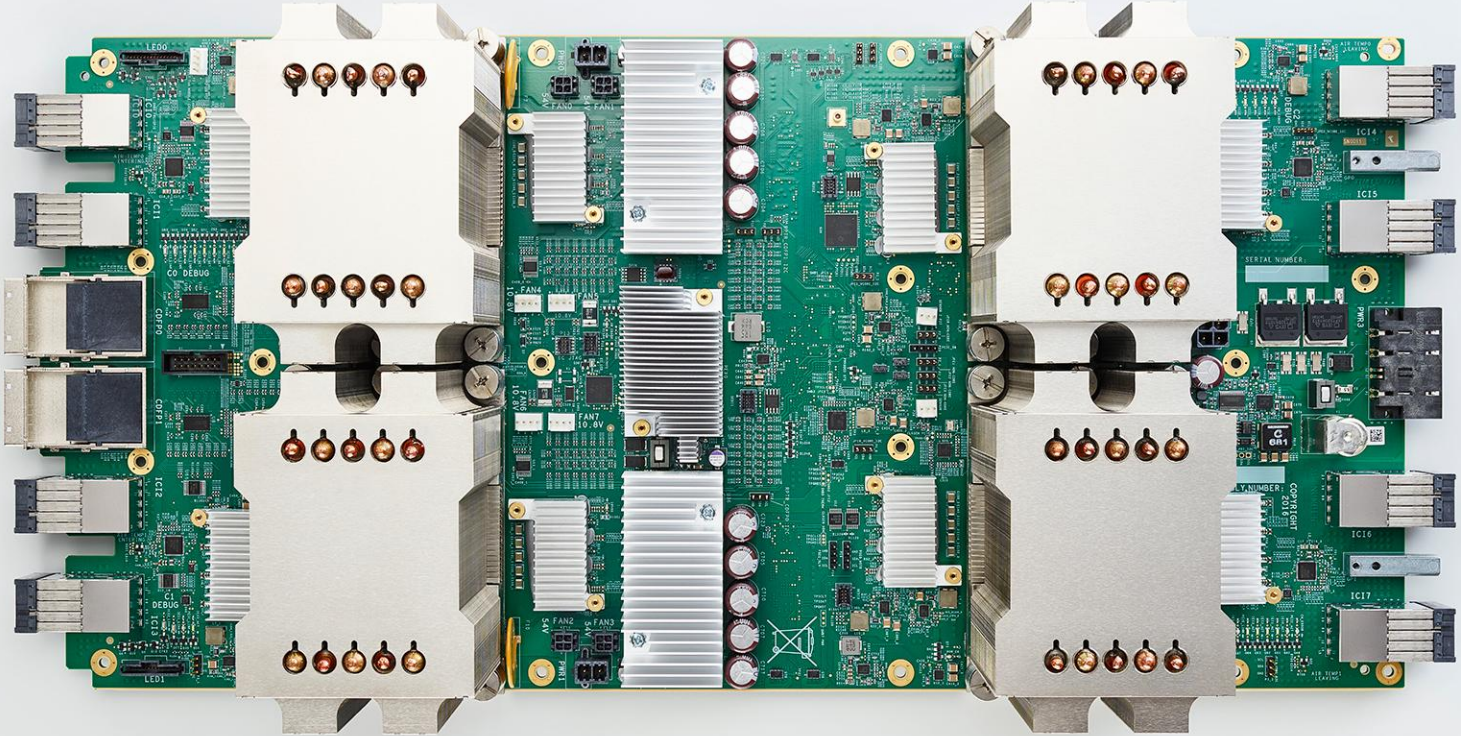
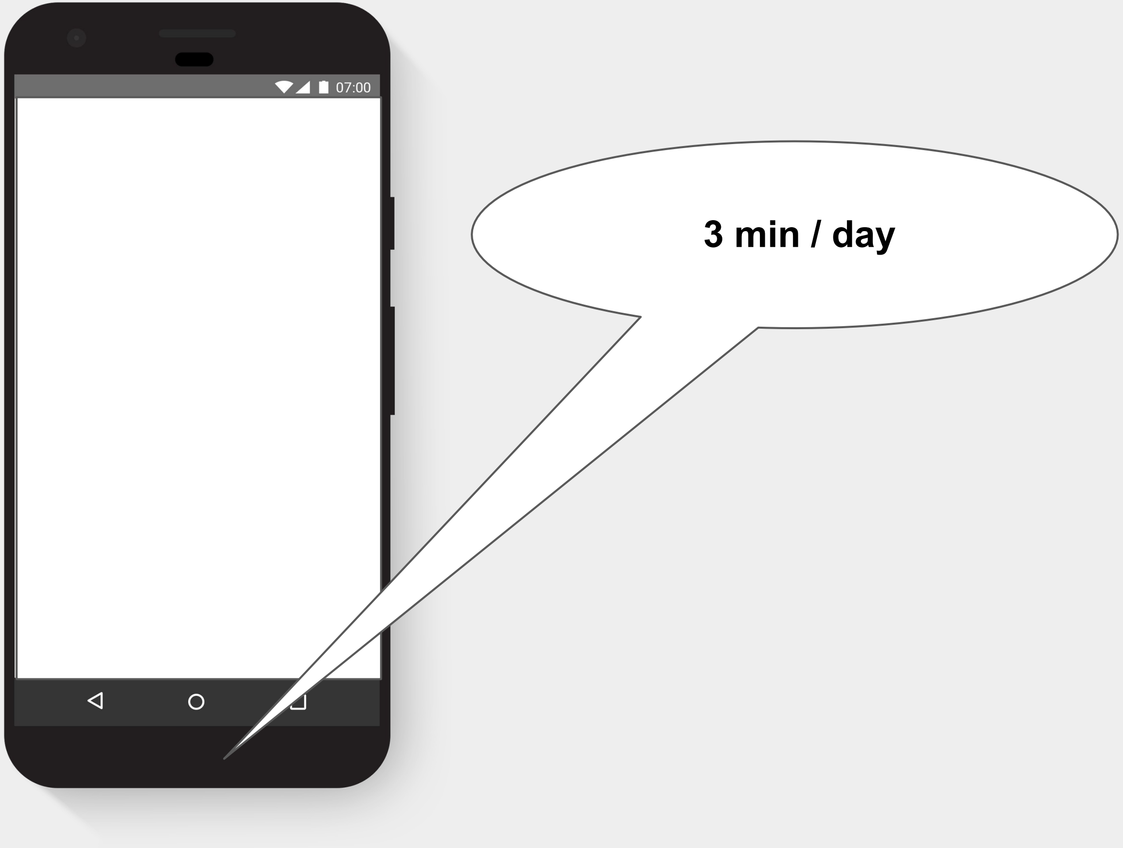
QCon SAN FRANCISCO

TensorFlow: Pushing the ML Boundaries

Magnus Hyttsten
@MagnusHyttsten
Google

Tensor Processing Unit (TPU) v2

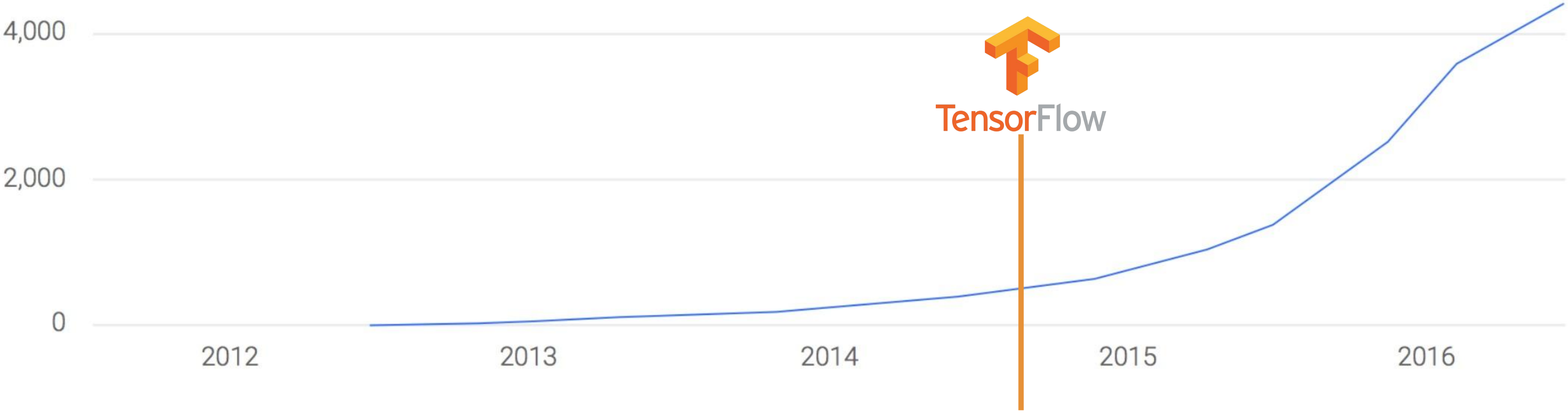



07:00
3 min / day

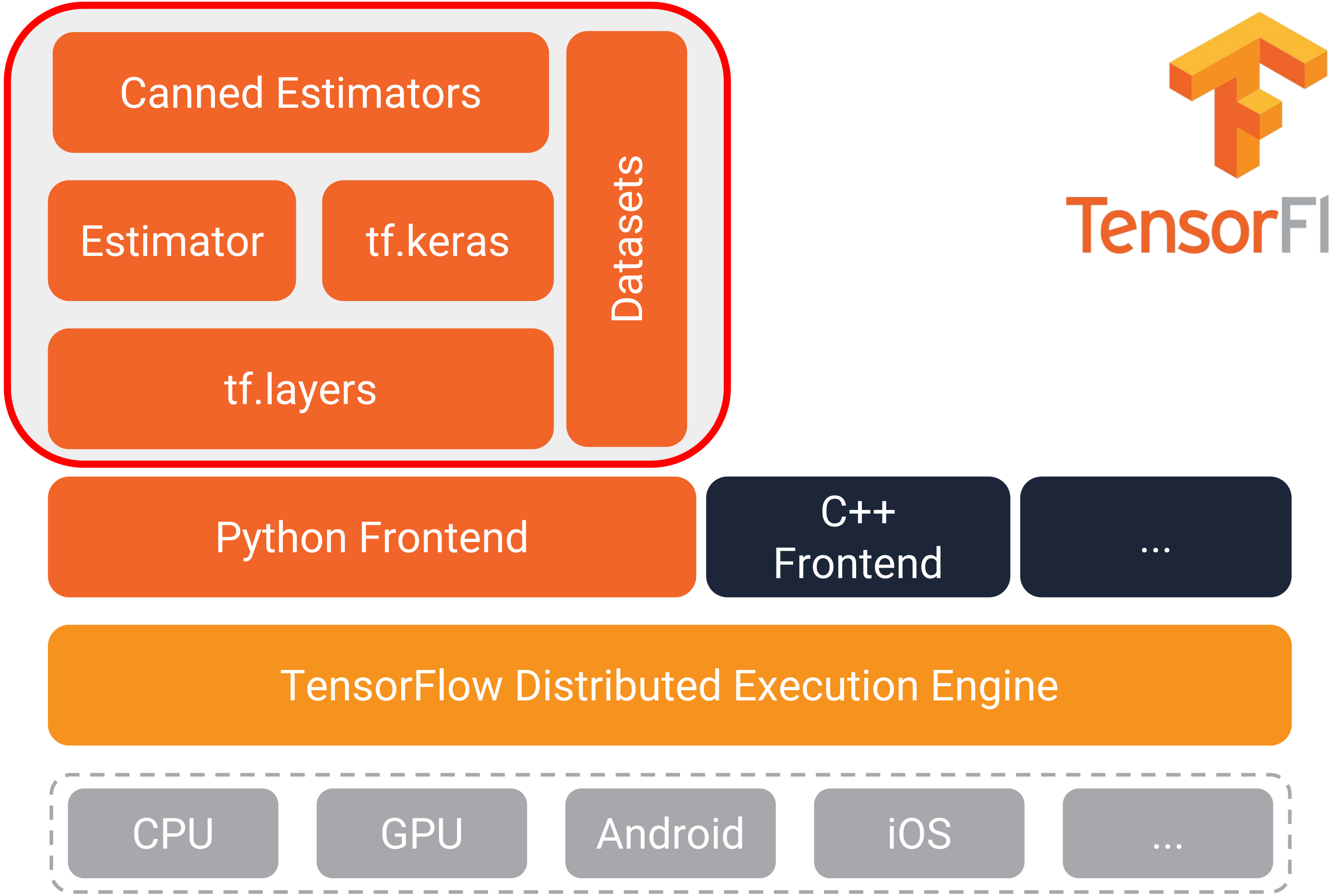
Guinea Pig

Meet Robin

Growing use of deep learning @ Google



Used across products


Canned Estimators
Estimator
tf.keras
tf.layers
Datasets
Python Frontend
C++ Frontend
...
TensorFlow Distributed Execution Engine
CPU
GPU
Android
iOS
...

TensorFlow

Keras - Code at goo.gl/TjQPfS

```python
from tensorflow.contrib.keras.python import keras
# couple more imports ...

model = Sequential()
model.add(Dense(512, activation='relu', input_shape=(784,)))
model.add(Dense(256, activation='relu')) # sizes arbitrary
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(10, activation='softmax'))
```

Code at goo.gl/TjQPfS

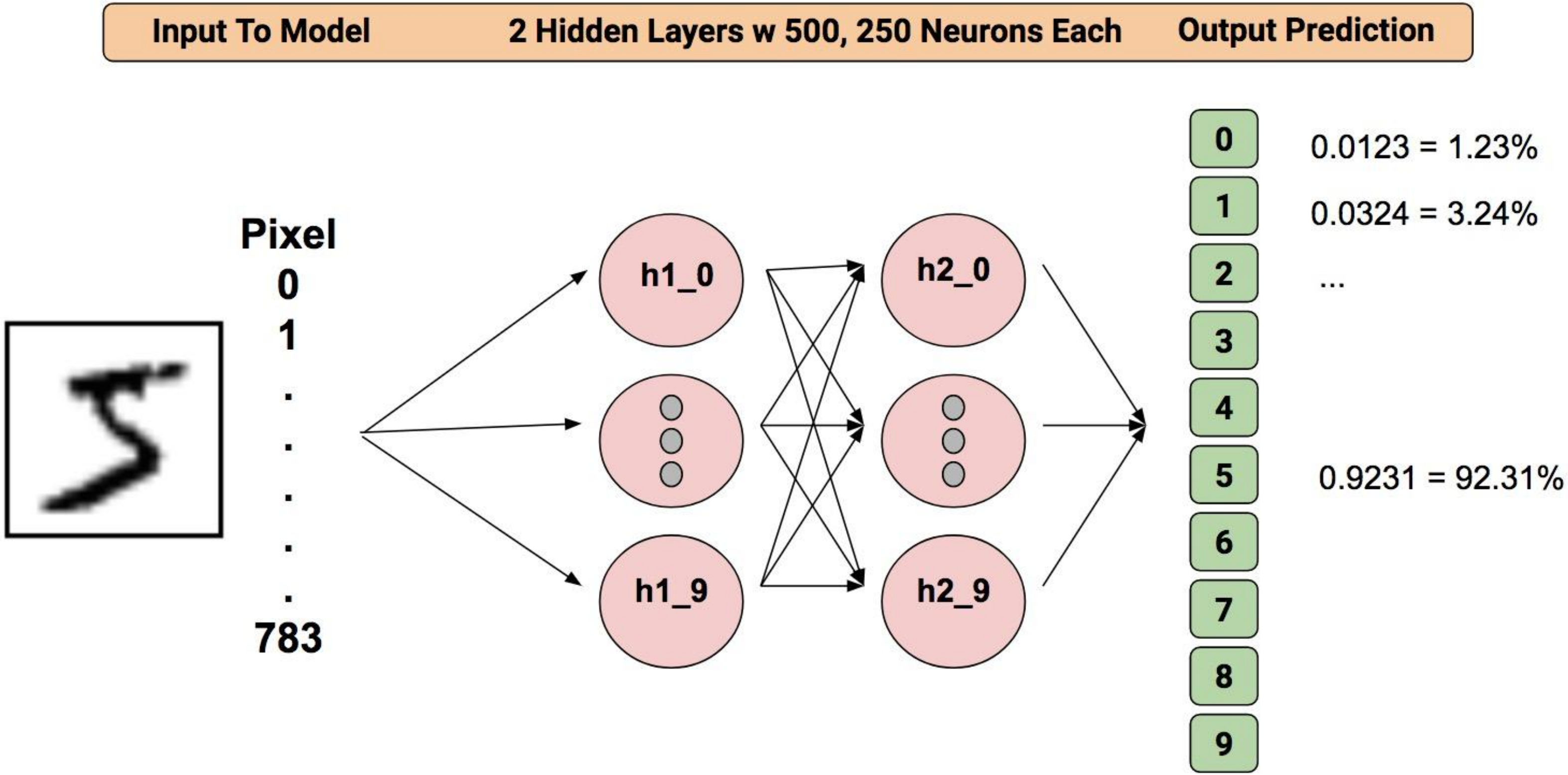
```
# terms not as scary as they sound
model.compile(loss='categorical_crossentropy',
              optimizer=RMSprop(),
              metrics=['accuracy'])

model.fit(x_train, y_train, batch_size=128, epochs=20)

score = model.evaluate(x_test, y_test)

print('Test accuracy:', score)
```

Datasets

```python
def my_input_fn(file_path, perform_shuffle=False, repeat_count=1):
    def decode_csv(line):
        parsed_line = tf.decode_csv(line, [[0.], [0.], [0.], [0.], [0]])
        label = parsed_line[-1:] # Last element is the label
        del parsed_line[-1] # Delete last element
        features = parsed_line # Everything (but last element) are the features
        d = dict(zip(feature_names, features)), label
        return d

    dataset = (tf.contrib.data.TextLineDataset(file_path) # Read text file
        .skip(1) # Skip header row
        .map(decode_csv)) # Transform each elem by applying decode_csv fn
    if perform_shuffle:
        # Randomizes input using a window of 256 elements (read into memory)
        dataset = dataset.shuffle(buffer_size=256)
    dataset = dataset.repeat(repeat_count) # Repeats dataset this # times
    dataset = dataset.batch(32)  # Batch size to use
    iterator = dataset.make_one_shot_iterator()
    batch_features, batch_labels = iterator.get_next()
    return batch_features, batch_labels
```

Datasets

```
classifier = tf.estimator.DNNClassifier(
   feature_columns=feature_columns, # The input features to our model
   hidden_units=[10, 10], # Two layers, each with 10 neurons
   n_classes=3,
   model_dir=PATH) # Path to where checkpoints etc are stored


classifier.train(
   input_fn=lambda: my_input_fn(FILE_TRAIN, True, 8))


evaluate_result = estimator.evaluate(
   input_fn=lambda: my_input_fn(FILE_TEST, False, 4)
print("Evaluation results")
for key in evaluate_result:
   print("   {}, was: {}".format(key, evaluate_result[key]))
```



Input To Model | 2 Hidden Layers w 500, 250 Neurons Each | Output Prediction

0.0123 = 1.23%

0.0324 = 3.24%

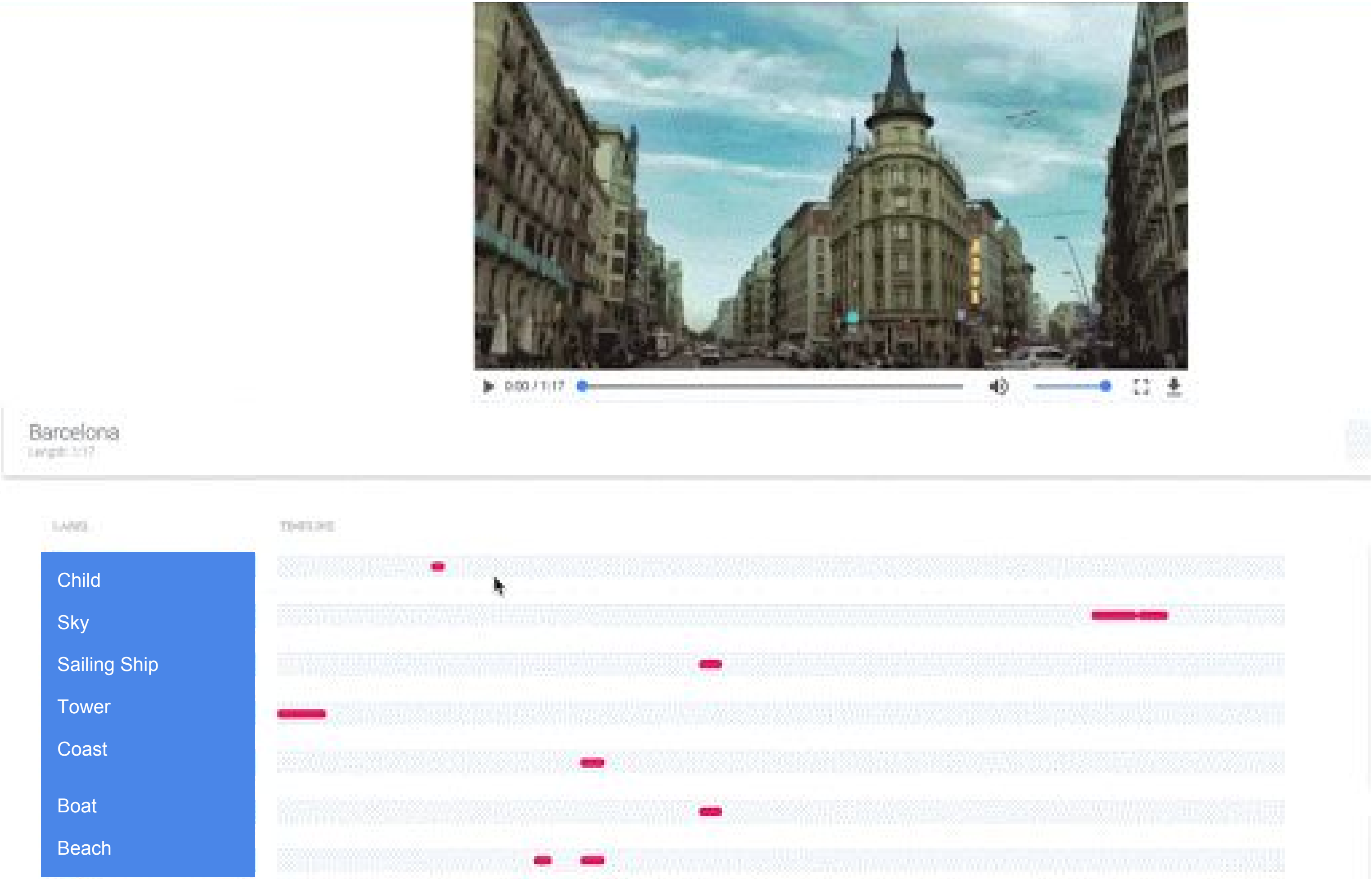
...

0.9231 = 92.31%

All summing to 100%

Getting Started

https://goo.gl/Ujm2Ep



Barcelona

Length: 1:17

LABEL

TIMELINE

Child

Sky

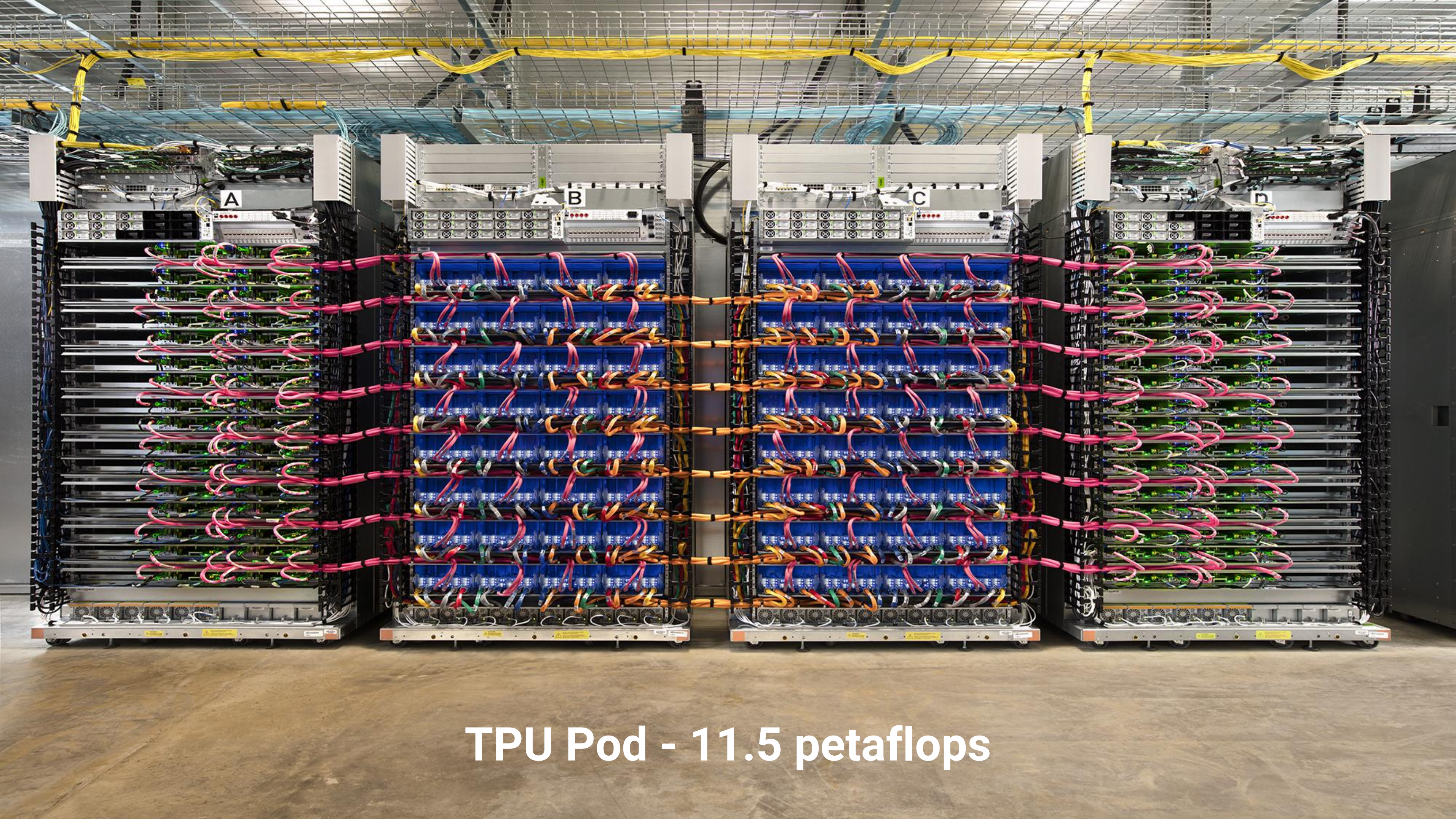
Sailing Ship

Tower

Coast

Boat

Beach



TPU Pod - 11.5 petaflops

QCon
SAN FRANCISCO

Thank You

g.co/tpusignup

Magnus Hyttsten
@MagnusHyttsten

Google