TensorFlow

Introduction to TensorFlow 2.0

Brad Miro - @bradmiro

Google

October 2019

TensorFlow

Deep Learning

Intro to TensorFlow

TensorFlow @ Google

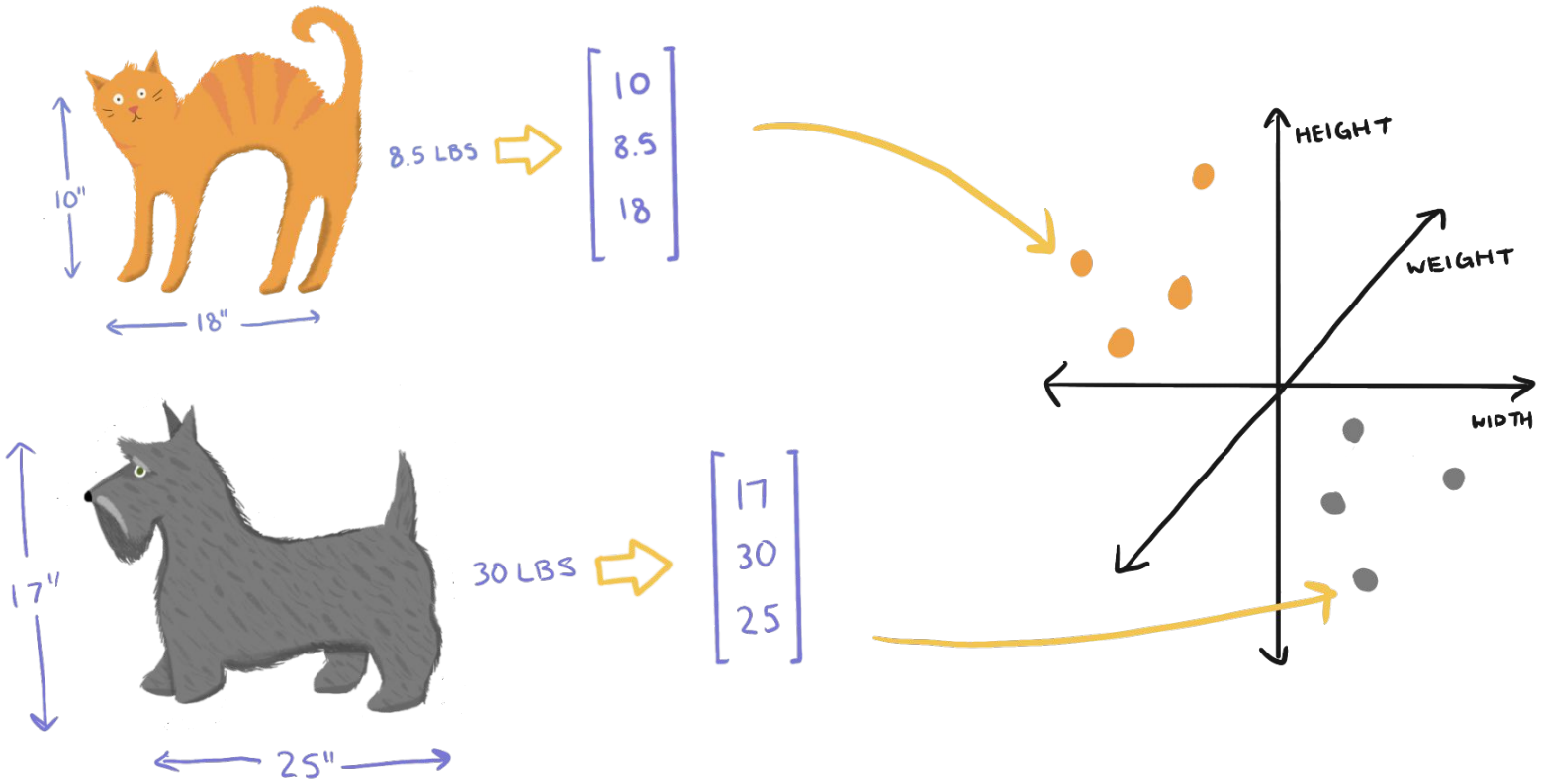
2.0 and Examples

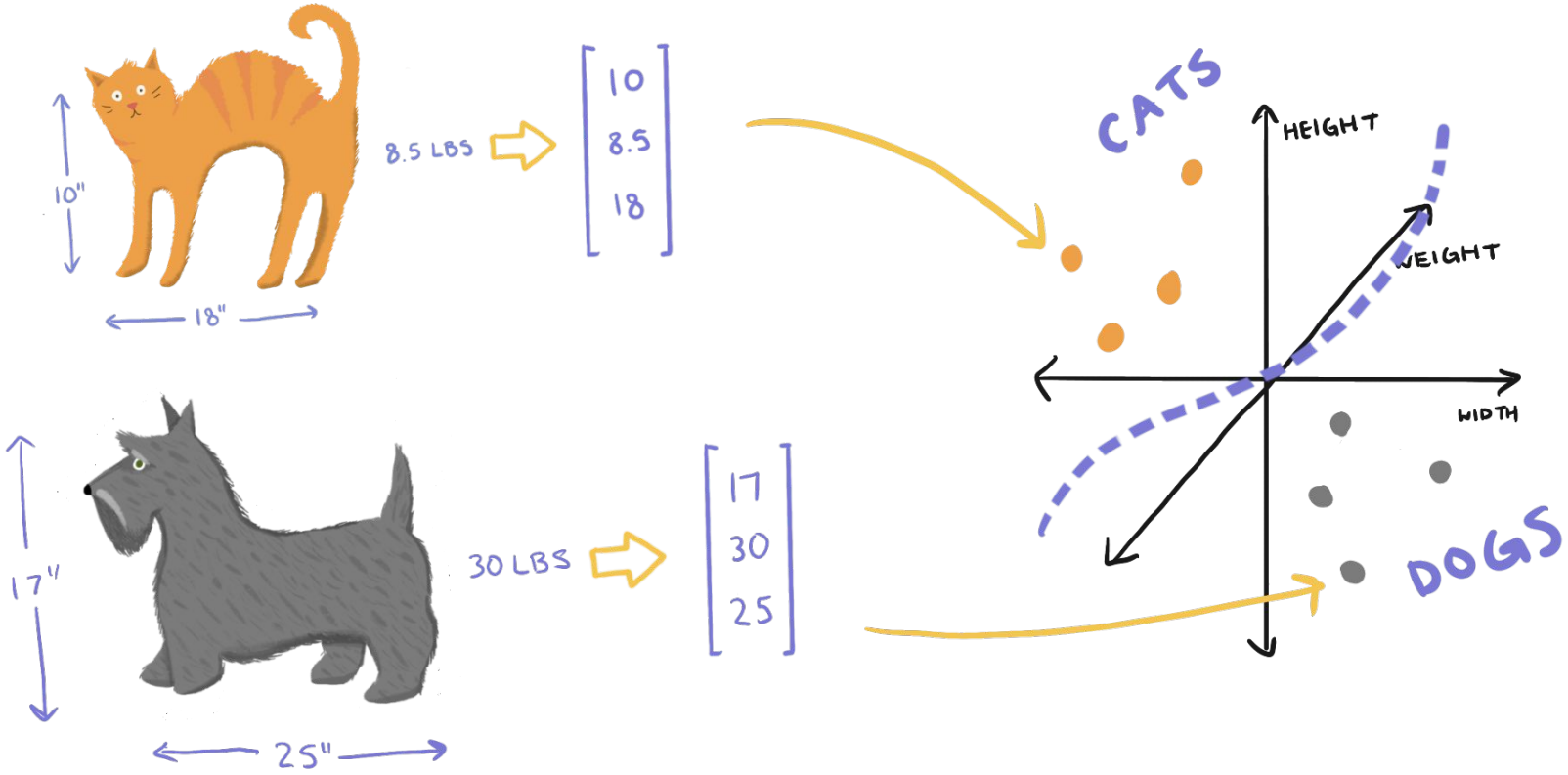
Getting Started

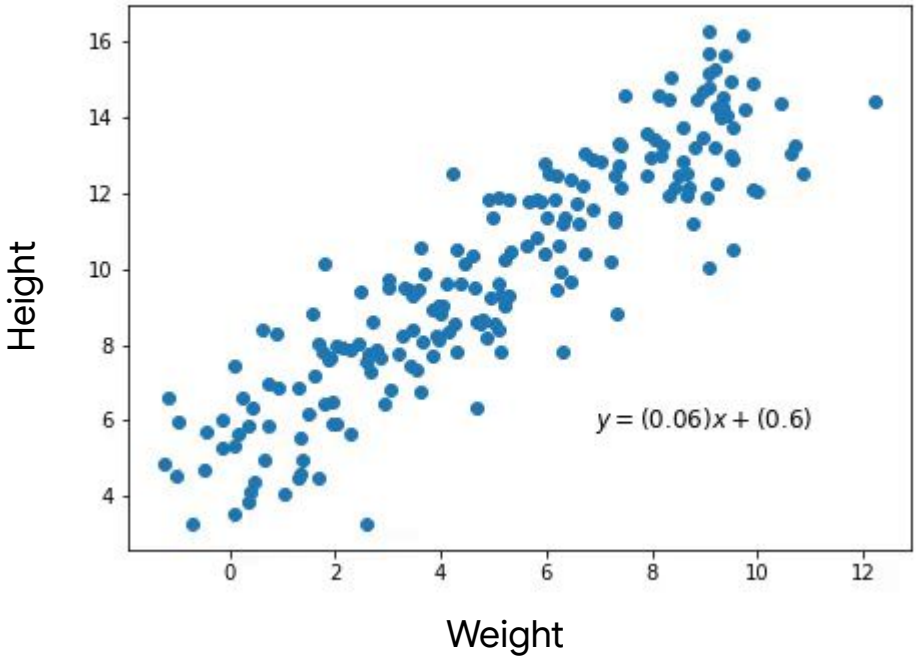
TensorFlow

Deep Learning

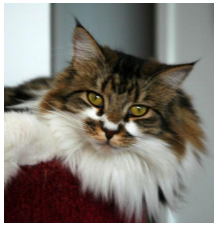
Doodles courtesy of @dalequark


10"
18"
8.5 LBS
10
8.5
18
17"
25"
30 LBS
17
30
25
HEIGHT
WEIGHT
WIDTH

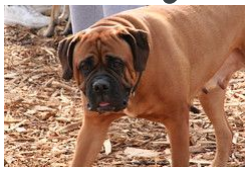

10"
18"
8.5 LBS
10
8.5
18
17"
25"
30 LBS
17
30
25
CATS
DOGS
HEIGHT
WEIGHT
WIDTH

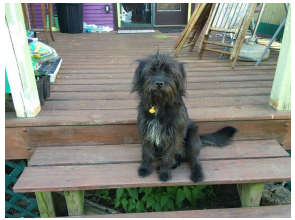

Height
Weight
$y = (0.06)x + (0.6)$
16
14
12
10
8
6
4
0
2
4
6
8
10
12

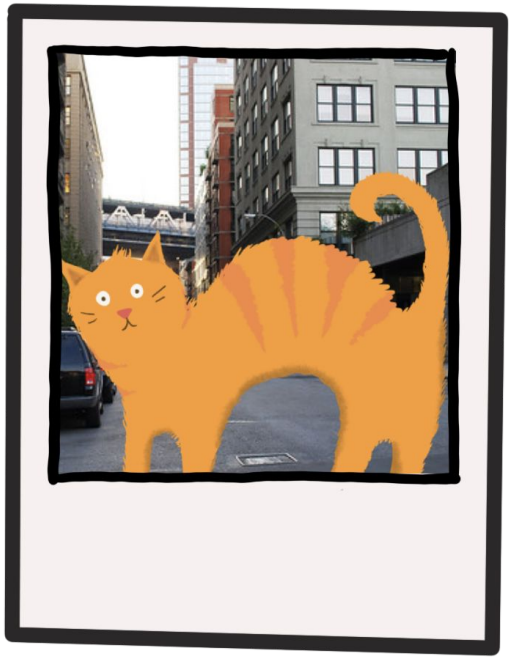
Examples of cats

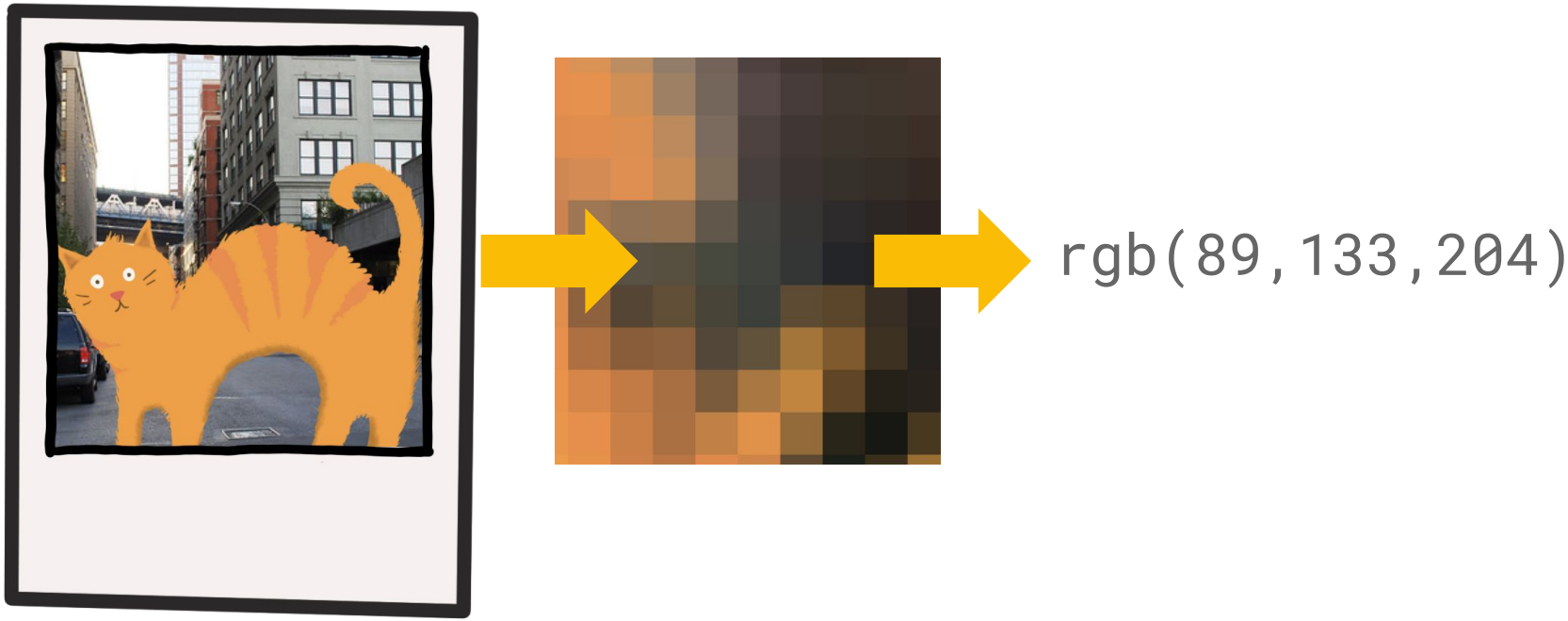


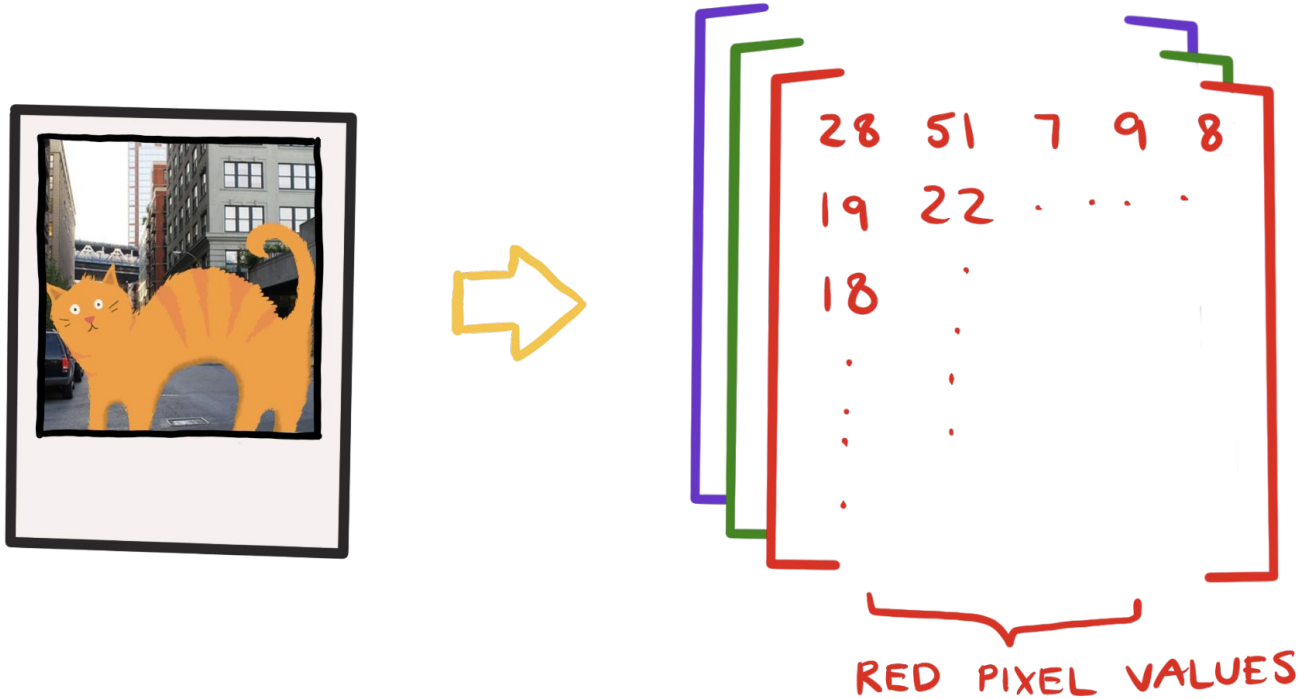
Examples of dogs

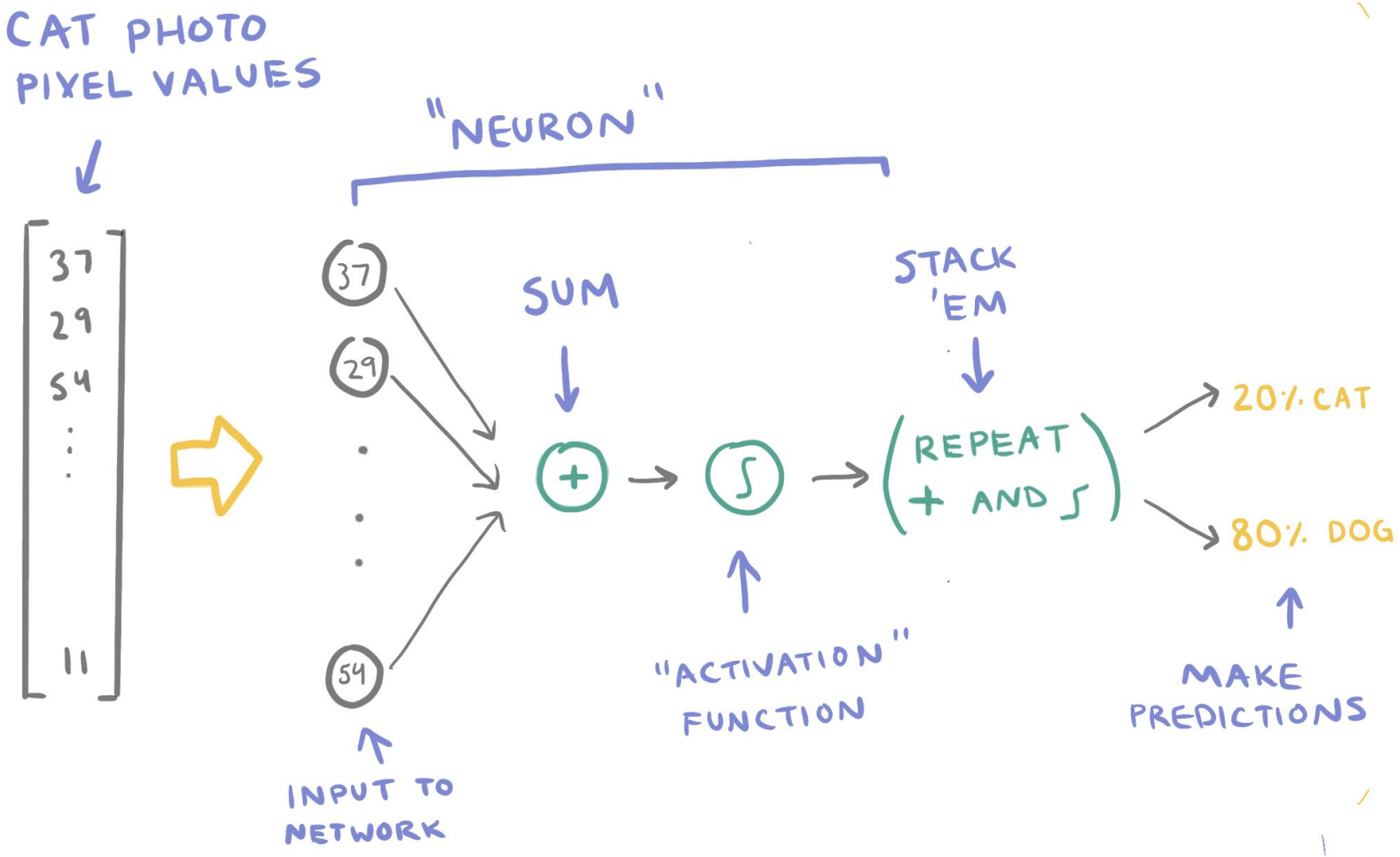


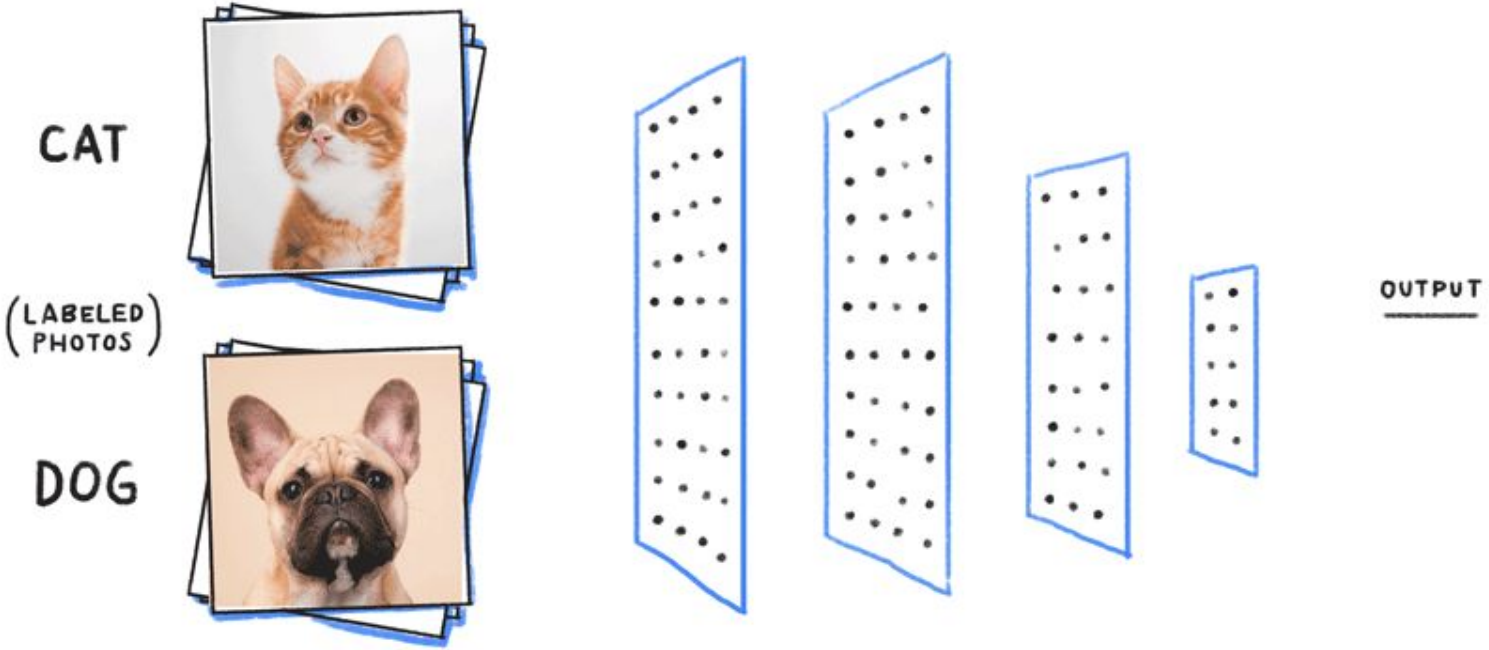





rgb(89,133,204)

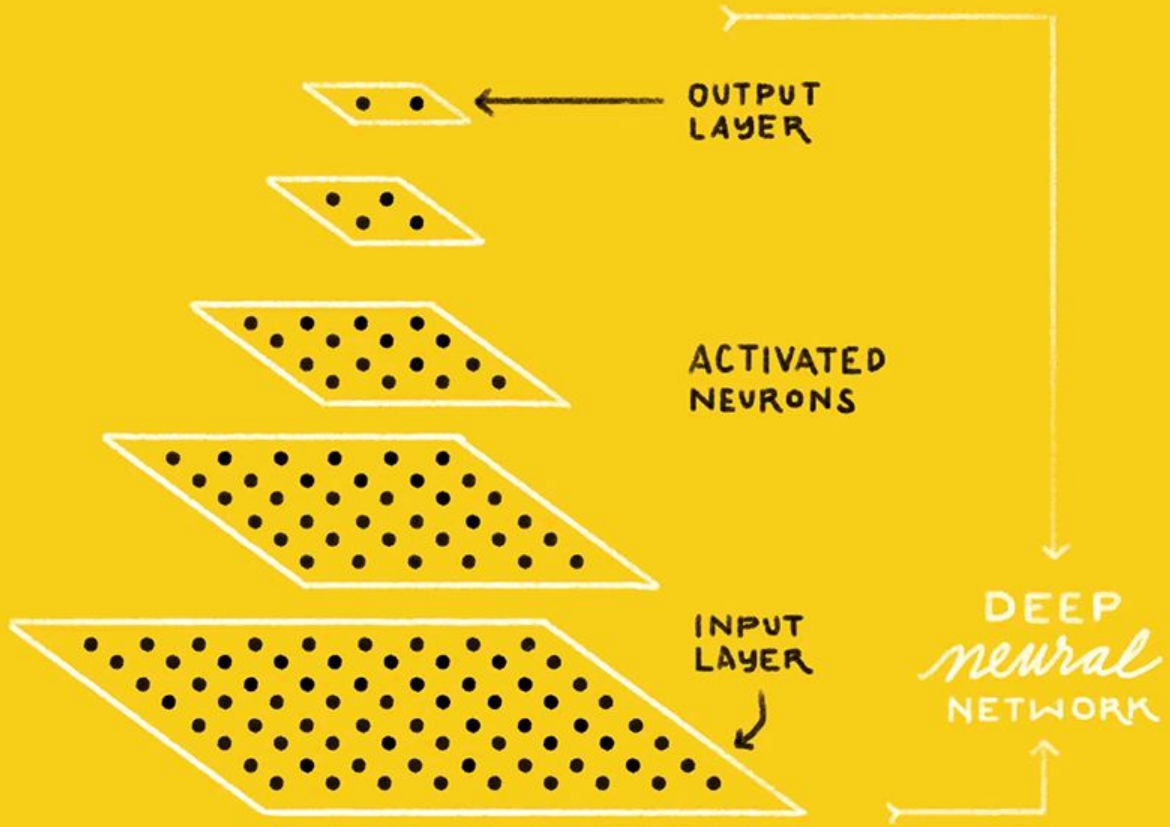

28 51 7 9 8
19 22
18
RED PIXEL VALUES

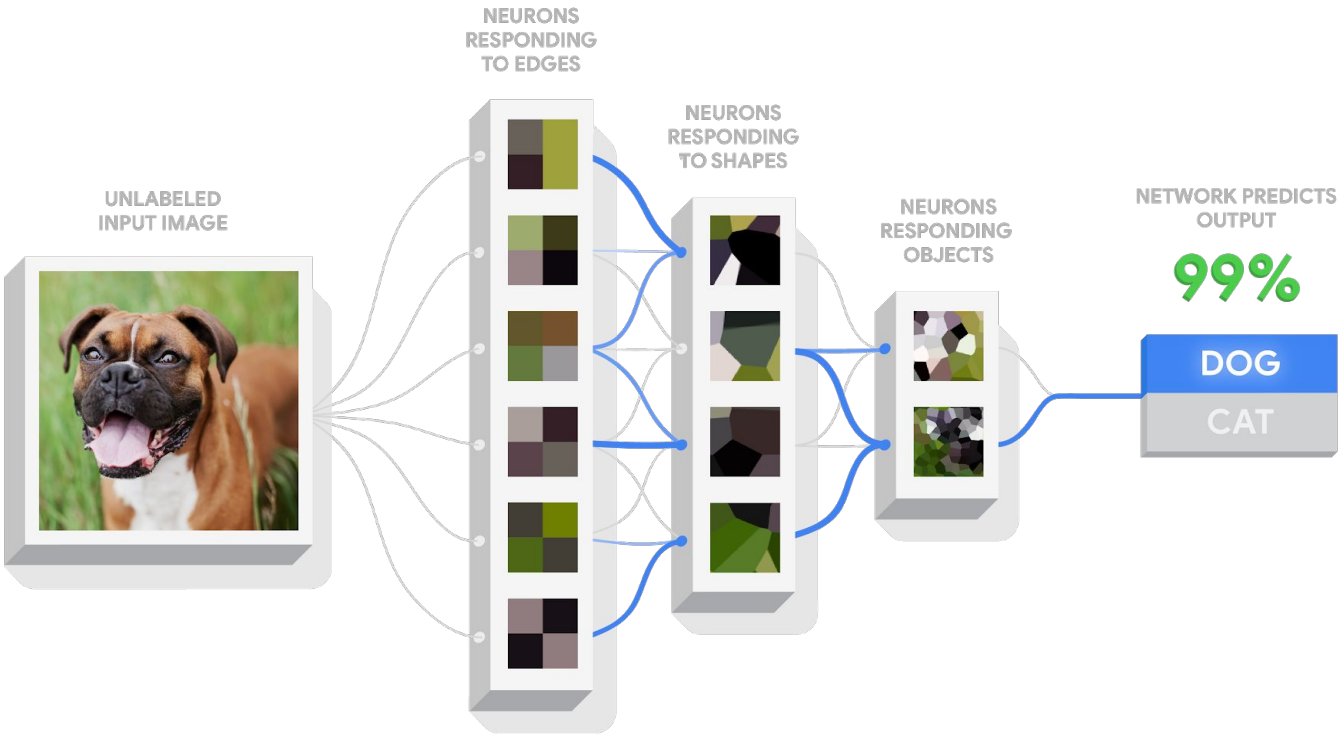

CAT PHOTO
PIXEL VALUES
37
29
54
11
"NEURON"
37
29
54
SUM
+
∫
STACK
'EM
REPEAT
+ AND ∫
20% CAT
80% DOG
INPUT TO
NETWORK
"ACTIVATION"
FUNCTION
MAKE
PREDICTIONS


CAT
(LABELED PHOTOS)
DOG
OUTPUT

IS THIS A

CAT or DOG?




UNLABELED
INPUT IMAGE
NEURONS
RESPONDING
TO EDGES
NEURONS
RESPONDING
TO SHAPES
NEURONS
RESPONDING
OBJECTS
NETWORK PREDICTS
OUTPUT
99%
DOG
CAT

Use Deep Learning When...

You have **lots** of data (~ 10k+ examples)

The problem is “complex” - speech, vision, natural language

The data is unstructured

You need the absolute “best” model

Don’t Use Deep Learning When...

You **don’t** have a large dataset

You are performing sufficiently well with traditional ML methods

Your data is structured and you possess the proper domain knowledge

Your model should be explainable

TensorFlow

TensorFlow

Open source deep learning library

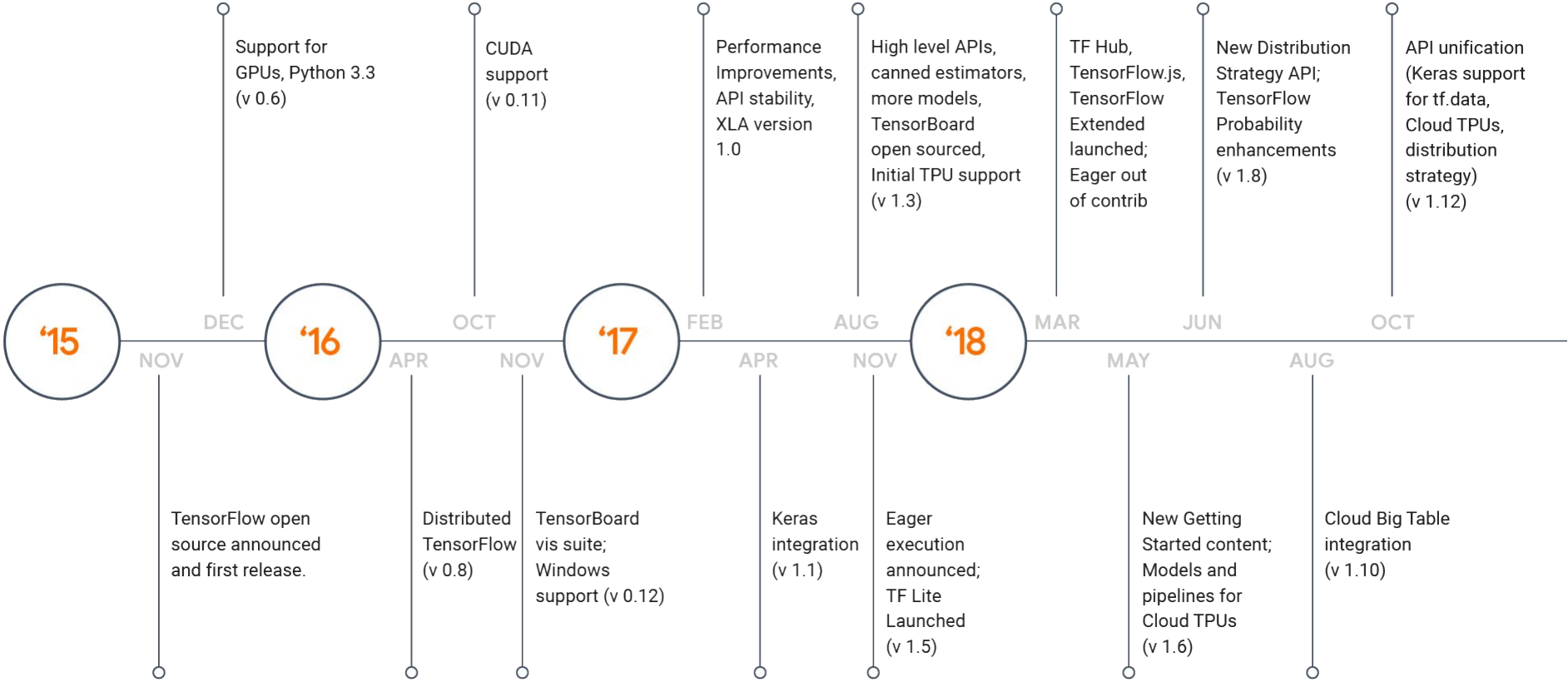
Utilities to help you write neural networks

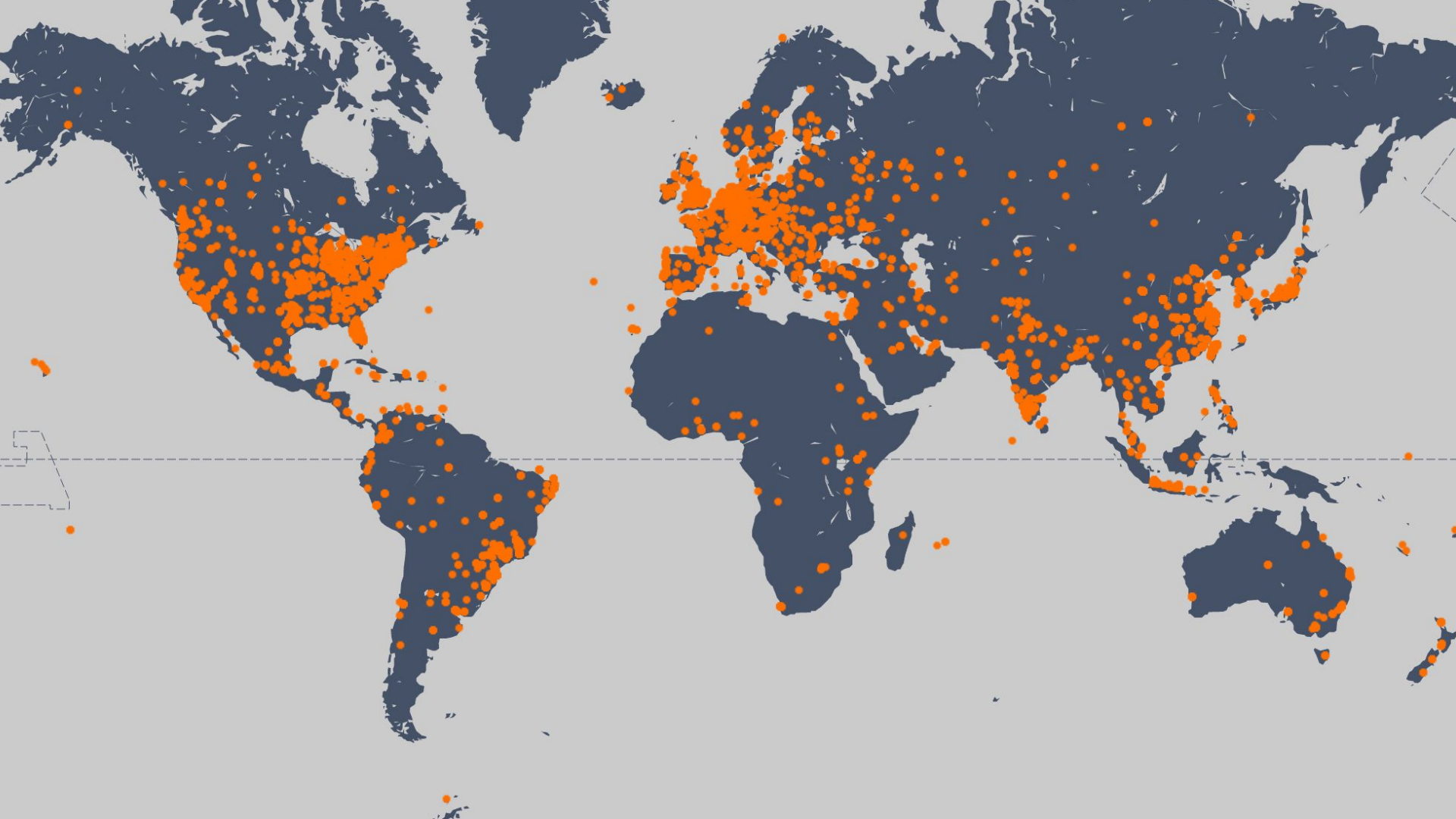
GPU / TPU support

Released by Google in 2015

>2200 Contributors

2.0 released September 2019


'15
NOV
TensorFlow open source announced and first release.
DEC
Support for GPUs, Python 3.3 (v 0.6)
'16
APR
Distributed TensorFlow (v 0.8)
OCT
CUDA support (v 0.11)
NOV
TensorBoard vis suite; Windows support (v 0.12)
'17
FEB
Performance Improvements, API stability, XLA version 1.0
APR
Keras integration (v 1.1)
AUG
High level APIs, canned estimators, more models, TensorBoard open sourced, Initial TPU support (v 1.3)
NOV
Eager execution announced; TF Lite Launched (v 1.5)
'18
MAR
TF Hub, TensorFlow.js, TensorFlow Extended launched; Eager out of contrib
MAY
New Getting Started content; Models and pipelines for Cloud TPUs (v 1.6)
JUN
New Distribution Strategy API; TensorFlow Probability enhancements (v 1.8)
AUG
Cloud Big Table integration (v 1.10)
OCT
API unification (Keras support for tf.data, Cloud TPUs, distribution strategy) (v 1.12)



41,000,000+	69,000+	12,000+	2,200+
downloads	commits	pull requests	contributors

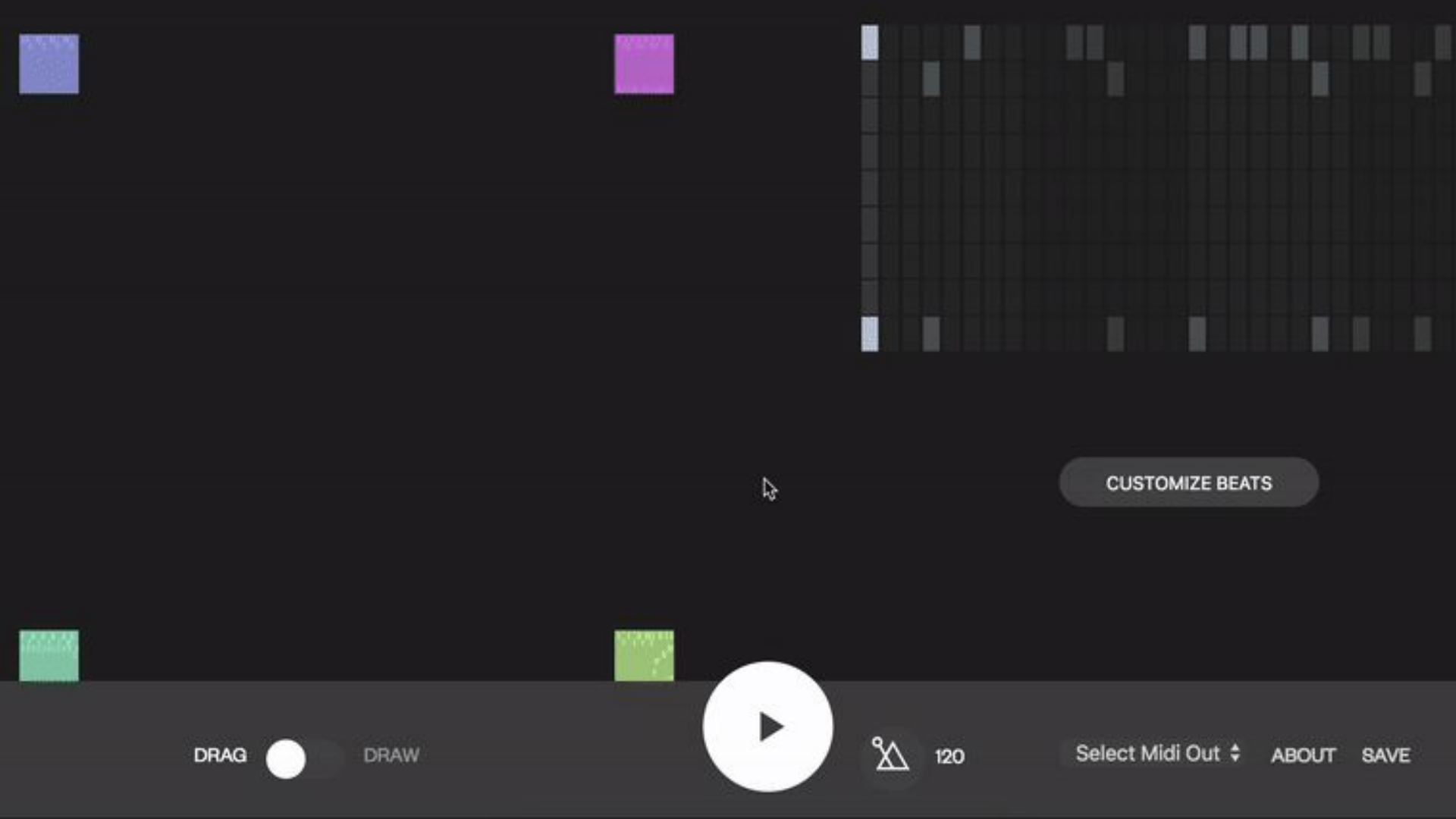
TensorFlow

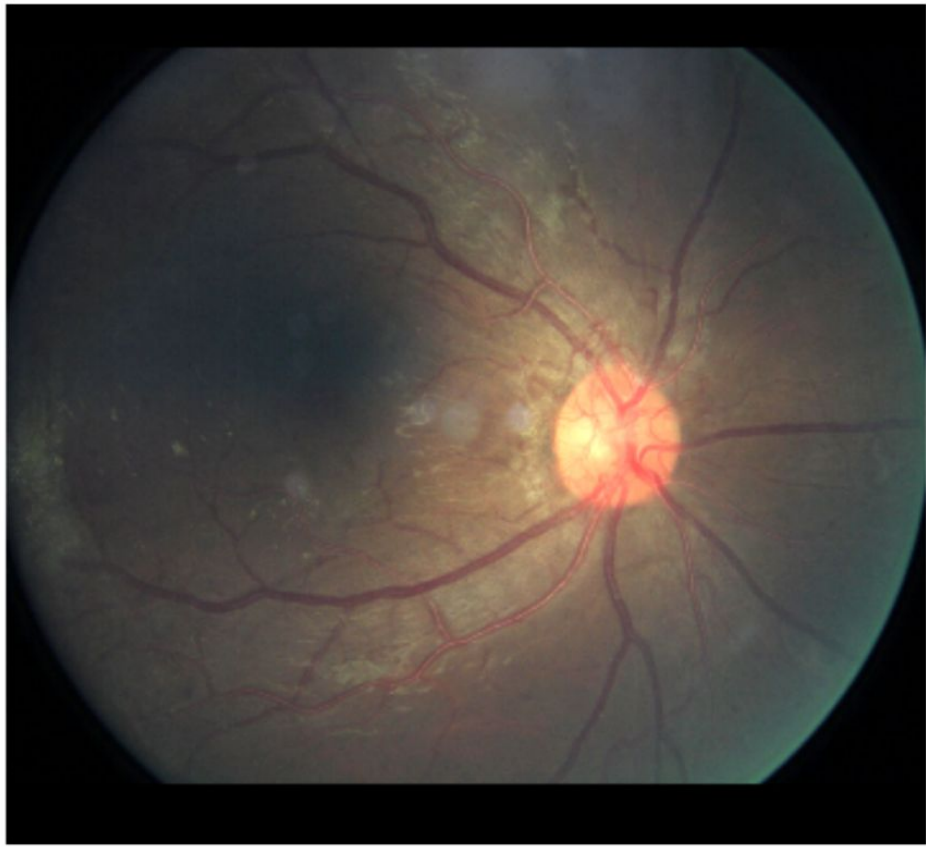
TensorFlow @ Google

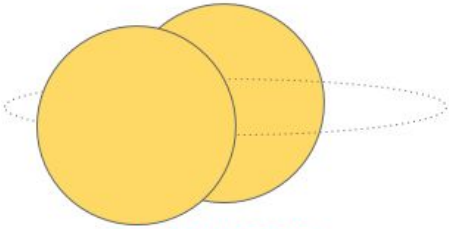
AI-powered data center efficiency

Global localization in Google Maps

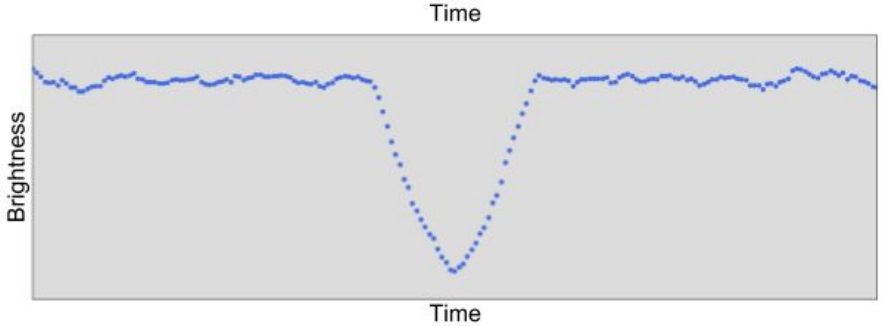
Portrait Mode on Google Pixel

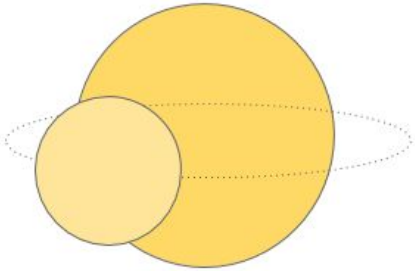

CUSTOMIZE BEATS
DRAG
DRAW
120
Select Midi Out
ABOUT
SAVE



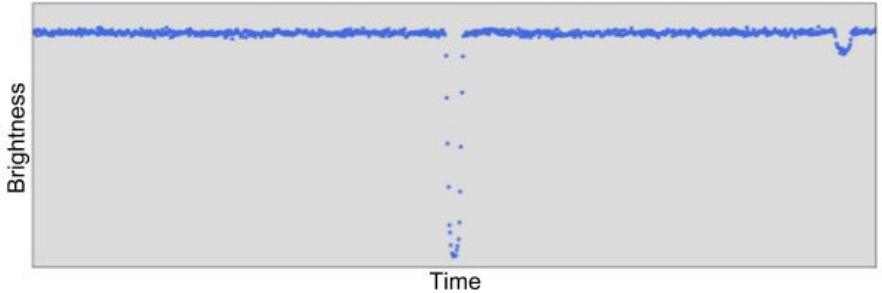


BINARY STAR



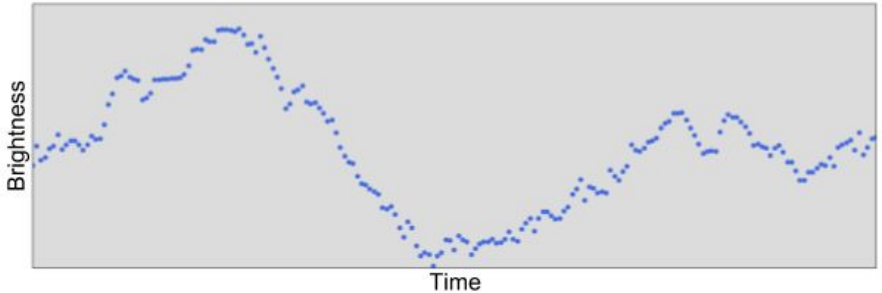


BINARY STAR



?

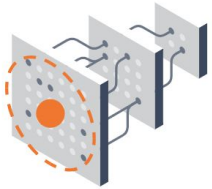
OTHER PHENOMENA



TensorFlow

2.0

TensorFlow 2.0



Easy

Simplified APIs.
Focused on Keras and
eager execution

Powerful

Flexibility and performance.
Power to do cutting edge research
and scale to > 1 exaflops

Scalable

Tested at Google-scale.
Deploy everywhere

Deploy anywhere

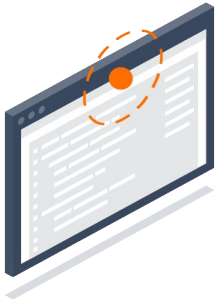
Servers

TensorFlow
Extended

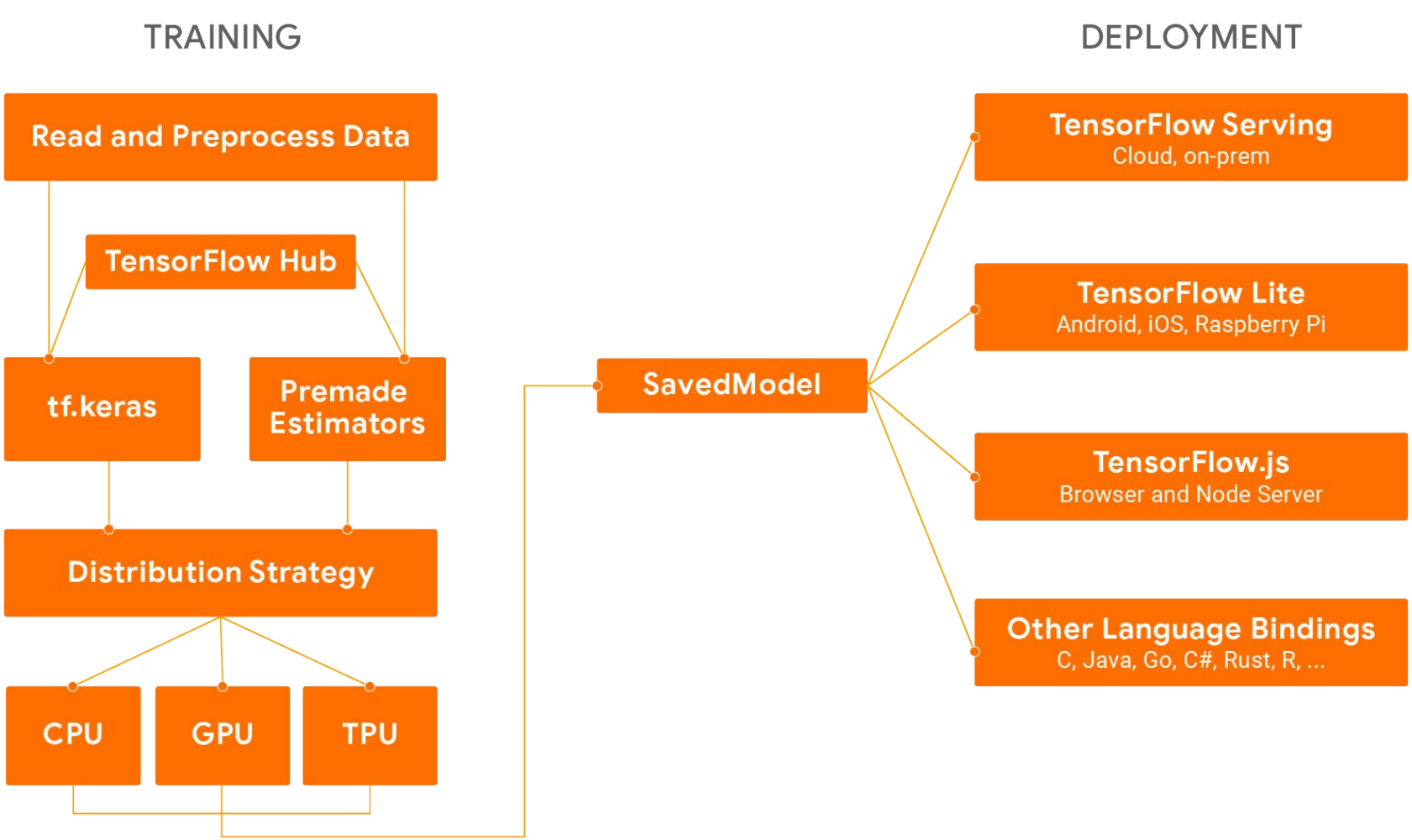
Edge devices

TensorFlow
Lite

JavaScript



TensorFlow
.JS


TRAINING
DEPLOYMENT
Read and Preprocess Data
TensorFlow Hub
tf.keras
Premade Estimators
Distribution Strategy
CPU
GPU
TPU
SavedModel
TensorFlow Serving
Cloud, on-prem
TensorFlow Lite
Android, iOS, Raspberry Pi
TensorFlow.js
Browser and Node Server
Other Language Bindings
C, Java, Go, C#, Rust, R, ...

TF Probability

TF Agents

Tensor2Tensor

TF Ranking

TF Text

TF Federated

TF Privacy

...

You can use TF 2.0 like NumPy

```
import tensorflow as tf # Assuming TF 2.0 is installed

a = tf.constant([[1, 2],[3, 4]])
b = tf.matmul(a, a)

print(b)
# tf.Tensor( [[ 7 10] [15 22]], shape=(2, 2), dtype=int32)

print(type(b.numpy()))
# <class 'numpy.ndarray'>
```

Specifics

What's Gone

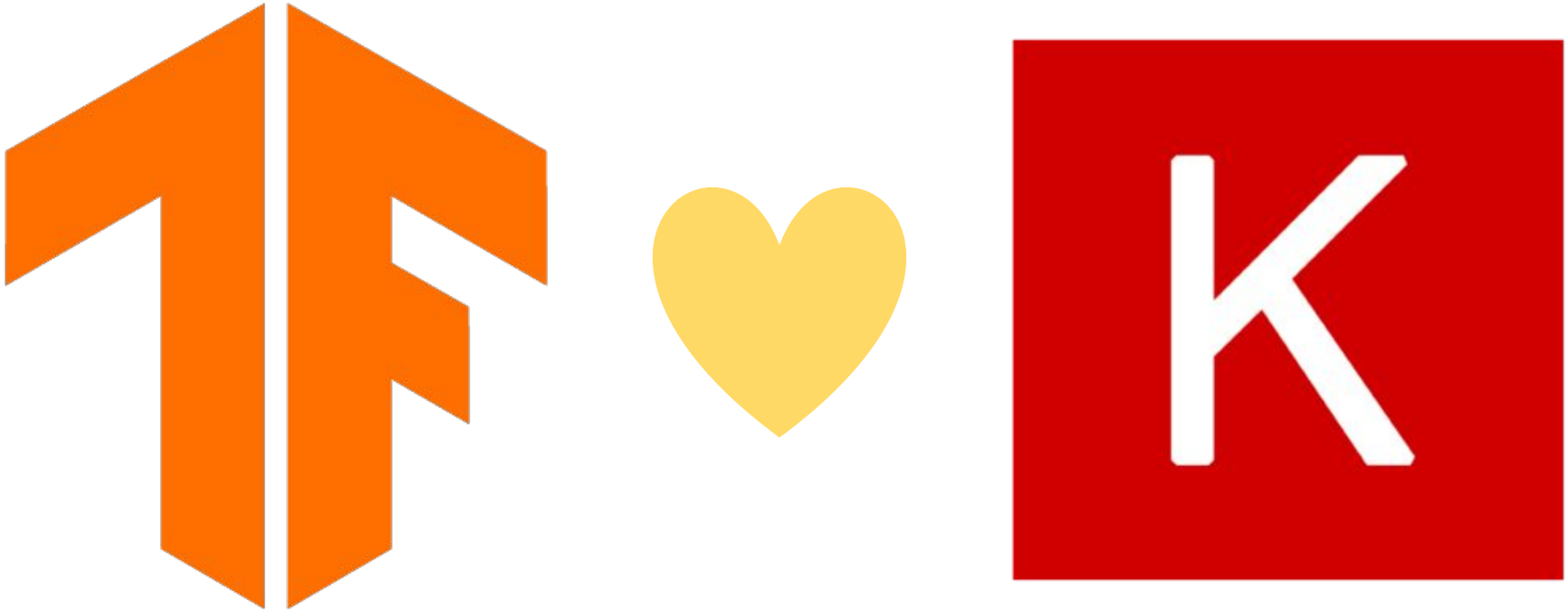
Session.run
tf.control_dependencies
tf.global_variables_initializer
tf.cond, tf.while_loop
tf.contrib

What's New

Eager execution by default
tf.function
Keras as main high-level api

TensorFlow

tf.keras



Keras and tf.keras

Fast prototyping, advanced research, and production

keras.io = reference implementation

```
import keras
```

tf.keras = **TensorFlow's implementation** (a superset, built-in to TF, no need to install Keras separately)

```
from tensorflow import keras
```

For Beginners

```
model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(),
  tf.keras.layers.Dense(512, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(10, activation='softmax')
])
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)
model.evaluate(x_test, y_test)
```

For Experts

```
class MyModel(tf.keras.Model):
  def __init__(self, num_classes=10):
    super(MyModel, self).__init__(name='my_model')
    self.dense_1 = layers.Dense(32, activation='relu')
    self.dense_2 = layers.Dense(num_classes, activation='sigmoid')

  def call(self, inputs):
    # Define your forward pass here,
    x = self.dense_1(inputs)
    return self.dense_2(x)
```

TensorFlow

What’s the difference?

Symbolic vs Imperative APIs

Symbolic (For Beginners)

- Your model is a graph of layers
- Any graph you compile will run
- **TensorFlow helps you debug** by catching errors at **compile time**

Imperative (For Experts)

- Your model is Python bytecode
- Complete flexibility and control
- Harder to debug / **harder to maintain**

TensorFlow

tf.function

Let’s make this faster

```
lstm_cell = tf.keras.layers.LSTMCell(10)



def fn(input, state):
  return lstm_cell(input, state)

input = tf.zeros([10, 10]); state = [tf.zeros([10, 10])] * 2
lstm_cell(input, state); fn(input, state) # warm up

# benchmark
timeit.timeit(lambda: lstm_cell(input, state), number=10) # 0.03
```

Let’s make this faster

```
lstm_cell = tf.keras.layers.LSTMCell(10)

@tf.function
def fn(input, state):
  return lstm_cell(input, state)

input = tf.zeros([10, 10]); state = [tf.zeros([10, 10])] * 2
lstm_cell(input, state); fn(input, state) # warm up

# benchmark
timeit.timeit(lambda: lstm_cell(input, state), number=10) # 0.03
timeit.timeit(lambda: fn(input, state), number=10) # 0.004
```

AutoGraph makes this possible

```
@tf.function
def f(x):
  while tf.reduce_sum(x) > 1:
    x = tf.tanh(x)
  return x

# you never need to run this (unless curious)
print(tf.autograph.to_code(f))
```

Generated code

```
def tf__f(x):
  def loop_test(x_1):
    with ag__.function_scope('loop_test'):
      return ag__.gt(tf.reduce_sum(x_1), 1)
  def loop_body(x_1):
    with ag__.function_scope('loop_body'):
      with ag__.utils.control_dependency_on_returns(tf.print(x_1)):
        tf_1, x = ag__.utils.alias_tensors(tf, x_1)
        x = tf_1.tanh(x)
        return x,
  x = ag__.while_stmt(loop_test, loop_body, (x,), (tf,))
  return x
```

TensorFlow

tf.distribution.Strategy

Going big: tf.distribute.Strategy

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(64, input_shape=[10]),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(10, activation='softmax')])

model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

Going big: Multi-GPU

```
strategy = tf.distribute.MirroredStrategy()

with strategy.scope():
  model = tf.keras.models.Sequential([
      tf.keras.layers.Dense(64, input_shape=[10]),
      tf.keras.layers.Dense(64, activation='relu'),
      tf.keras.layers.Dense(10, activation='softmax')])

  model.compile(optimizer='adam',
                loss='categorical_crossentropy',
                metrics=['accuracy'])
```

TensorFlow

tensorflow_datasets

```python
# Load data

import tensorflow_datasets as tfds

dataset = tfds.load('cats_vs_dogs', as_supervised=True)
mnist_train, mnist_test = dataset['train'], dataset['test']

def scale(image, label):
  image = tf.cast(image, tf.float32)
  image /= 255
  return image, label

mnist_train = mnist_train.map(scale).batch(64)
mnist_test = mnist_test.map(scale).batch(64)
```

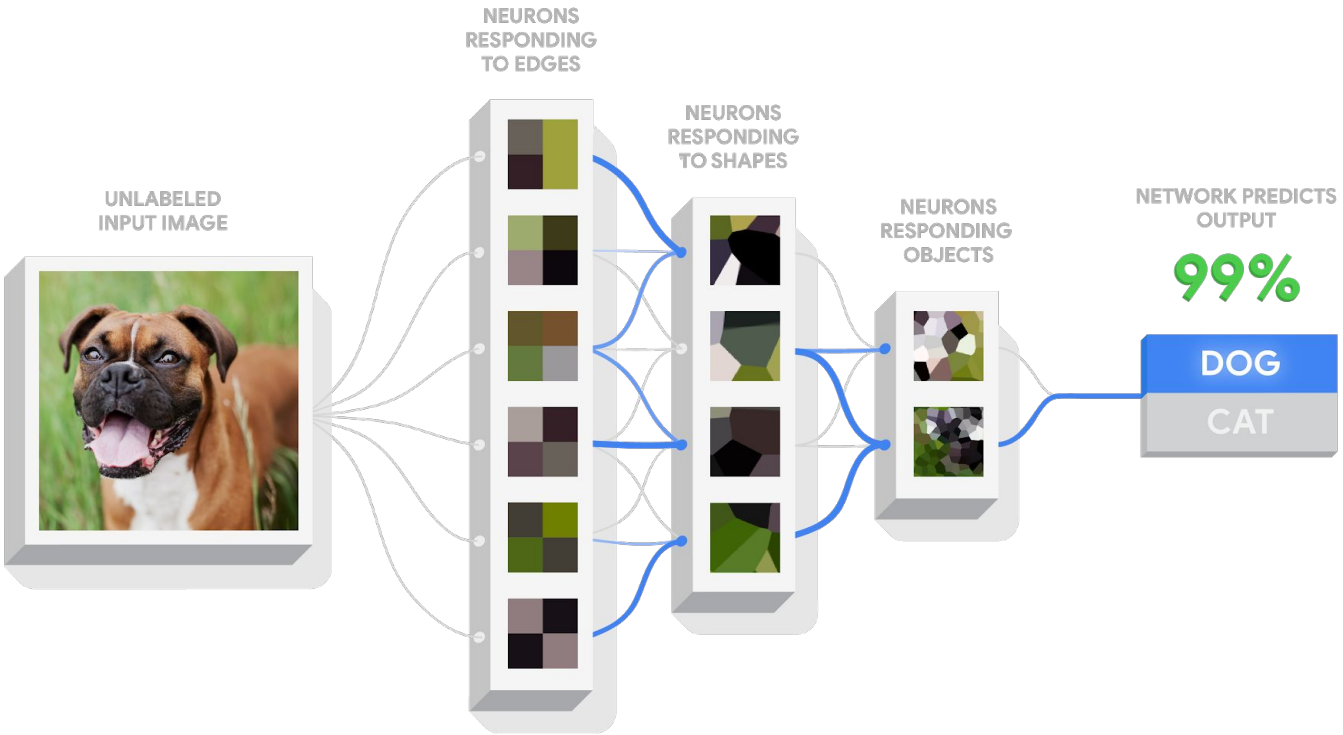
TensorFlow Datasets

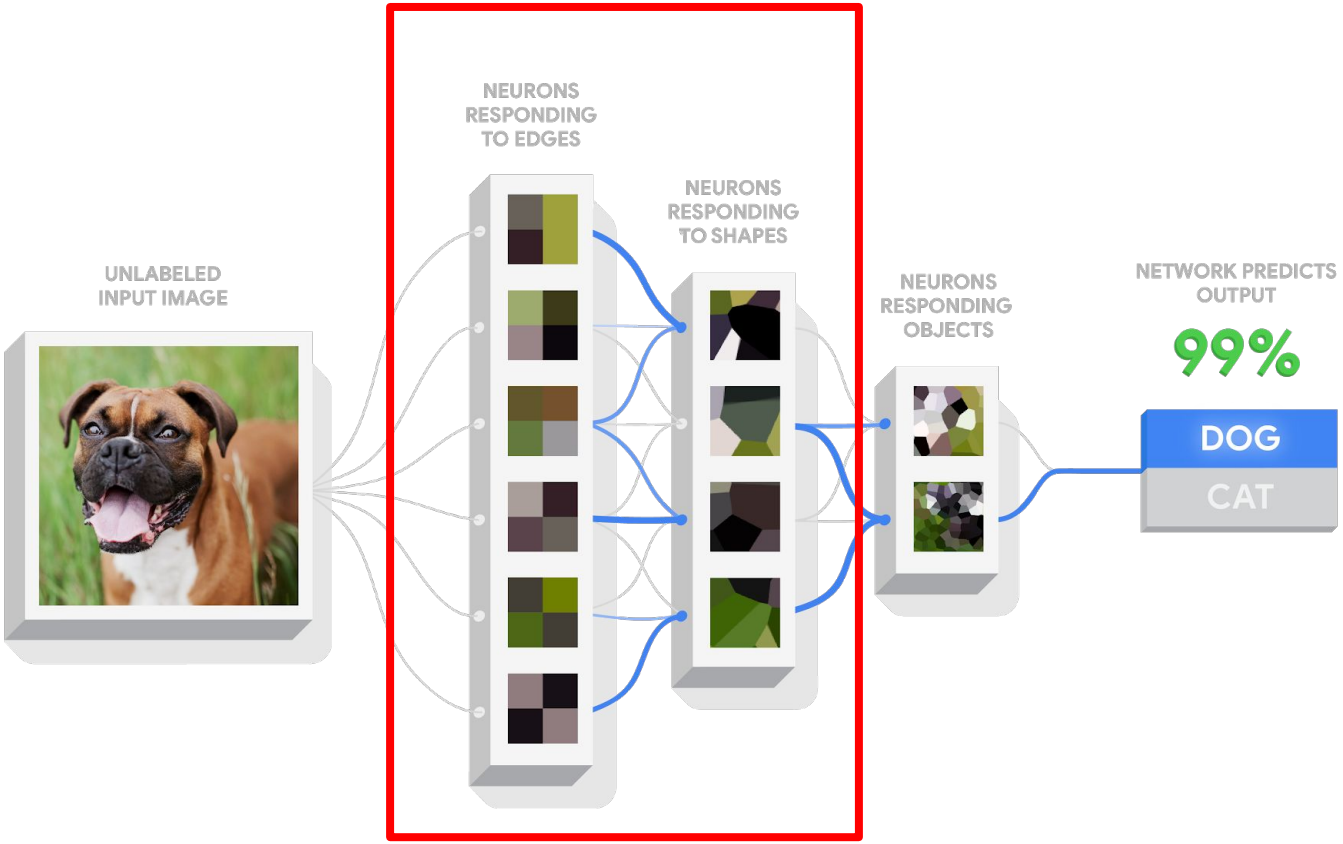
- audio
  - "nsynth"
- image
  - "cifar10"
  - "diabetic_retinopathy_detection"
  - "imagenet2012"
  - "mnist"
- structured
  - "titanic"
- text
  - "imdb_reviews"
  - "lm1b"
  - "squad"
- translate
  - "wmt_translate_ende"
  - "wmt_translate_enfr"
- video
  - "bair_robot_pushing_small"
  - "moving_mnist"
  - "starcraft_video"

More at tensorflow.org/datasets

TensorFlow

Transfer Learning


UNLABELED
INPUT IMAGE
NEURONS
RESPONDING
TO EDGES
NEURONS
RESPONDING
TO SHAPES
NEURONS
RESPONDING
OBJECTS
NETWORK PREDICTS
OUTPUT
99%
DOG
CAT


UNLABELED
INPUT IMAGE
NEURONS
RESPONDING
TO EDGES
NEURONS
RESPONDING
TO SHAPES
NEURONS
RESPONDING
OBJECTS
NETWORK PREDICTS
OUTPUT
99%
DOG
CAT

Transfer Learning

```
import tensorflow as tf

base_model = tf.keras.applications.SequentialMobileNetV2(
                 input_shape=(160, 160, 3),
                 include_top=False,
                 weights='imagenet')

base_model.trainable = False

model = tf.keras.models.Sequential([
  base_model,
  tf.keras.layers.GlobalAveragePooling2D(),
  tf.keras.layers.Dense(1)
])
# Compile and fit
```

TensorFlow Hub

USER GUIDE

Text

Embedding

Image

Classification

Feature Vector

Generator

Video

Classification

Publishers

Google

DeepMind

Text embedding

universal-sentence-encoder By Google

text-embedding DAN en

Encoder of greater-than-word length text trained on a variety of data.

nnlm-en-dim128 By Google

text-embedding Google News NNLM en

Token based text embedding trained on English Google News 200B corpus.

elmo By Google

text-embedding 1 Billion Word Benchmark ELMo en

Embeddings from a language model trained on the 1 Billion Word Benchmark.

View more text embeddings

Image feature vectors

imagenet/inception_v3/feature_vector By Google

image-feature-vector ImageNet (ILSVRC-2012-CLS) Inception V3

Feature vectors of images with Inception V3 trained on ImageNet (ILSVRC-2012-CLS).

TensorFlow

Upgrading

Upgrading

Migration guides

`tf.compat.v1` for backwards compatibility

`tf_upgrade_v2` script

```
[4]  !tf_upgrade_v2 --infile text_generation.py --outfile text_generation_upgraded.py
```

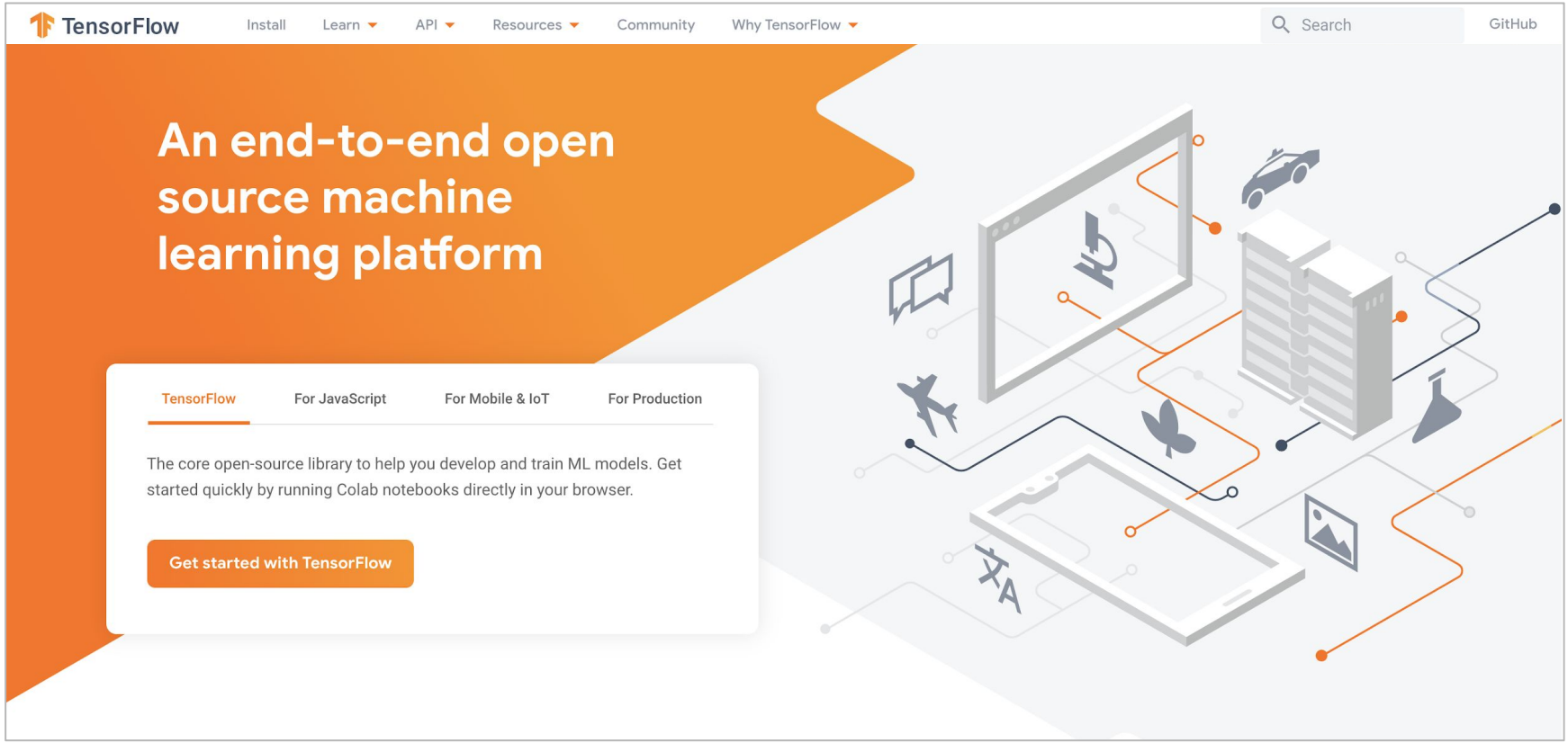
```
INFO line 4:0: Renamed 'tf.enable_eager_execution' to 'tf.compat.v1.enable_eager_executi
INFO line 240:16: Renamed 'tf.train.AdamOptimizer' to 'tf.compat.v1.train.AdamOptimizer'
INFO line 332:21: Added keywords to args of function 'tf.multinomial'
INFO line 332:21: Renamed 'tf.multinomial' to 'tf.random.categorical'
INFO line 375:12: Renamed 'tf.train.AdamOptimizer' to 'tf.compat.v1.train.AdamOptimizer'
INFO line 392:21: tf.losses.sparse_softmax_cross_entropy requires manual check. tf.losse
INFO line 392:21: Renamed 'tf.losses.sparse_softmax_cross_entropy' to 'tf.compat.v1.loss
TensorFlow 2.0 Upgrade Script
-----------------------------
Converted 1 files
Detected 0 issues that require attention
--------------------------------------------------------------------------------


Make sure to read the detailed log 'report.txt'
```

TensorFlow

Getting Started

TensorFlow 2.0

```
pip install tensorflow
```


TensorFlow
Install
Learn
API
Resources
Community
Why TensorFlow
Search
GitHub
An end-to-end open source machine learning platform
TensorFlow
For JavaScript
For Mobile & IoT
For Production
The core open-source library to help you develop and train ML models. Get started quickly by running Colab notebooks directly in your browser.
Get started with TensorFlow

New Courses

deeplearning.ai

coursera

Introduction to TensorFlow
for AI, ML and DL

coursera.org/learn/introduction-tensorflow

UDACITY

Intro to TensorFlow
for Deep Learning

udacity.com/tensorflow

JS

3 To do

- Feature: Unify argument names (~70).
  tensorflow#25357 opened by dynamicwebpaige
  2.0 type:feature
- Keras model evaluate() progress bar randomly stops before 100%
  tensorflow#24593 opened by ageron
  2.0 comp:keras
- Side effects induced by attribute and slice operators must be limited in Python control flow.
  tensorflow#26189 opened by dynamicwebpaige
  2.0

28 In progress

- 2.0 Reference Models: Keras Application Set (1 GPU)
  tensorflow#25341 opened by dynamicwebpaige
  2.0 type:feature
- 2.0 Reference Models: Transformer (1 GPU, 8 GPU with dist strat and Keras)
  tensorflow#25342 opened by dynamicwebpaige
  2.0 type:feature
- 2.0 Reference Models: NMT Model (1 GPU, 8 GPU with dist strat and Keras)
  tensorflow#25343 opened by dynamicwebpaige
  2.0 type:feature

18 Done

- Feature: modify the TF 2.0 upgrade script to convert .ipynb files.
  tensorflow#25448 opened by dynamicwebpaige
  2.0 type:feature
- Feature: TensorFlow.js compatibility with TF 2.0.
  tensorflow#25360 opened by dynamicwebpaige
  2.0 type:feature
- Feature: TensorFlow Lite compatibility with TF 2.0.
  tensorflow#25361 opened by dynamicwebpaige
  2.0 comp:lite type:feature
- Feature: TensorBoard compatibility

github.com/orgs/tensorflow/projects/4

Go build.

```
pip install tensorflow
```

tensorflow.org

TensorFlow

tf.thanks!

Brad Miro - @bradmiro

tensorflow.org

October 2019