We will start in 5min

Getting Started with TensorFlow 2.0

TensorFlow 2.0

TensorFlow

An open source Deep Learning library

- **>1,800 contributors worldwide**
- Apache 2.0 license
- Released by Google in 2015

TensorFlow 2.0

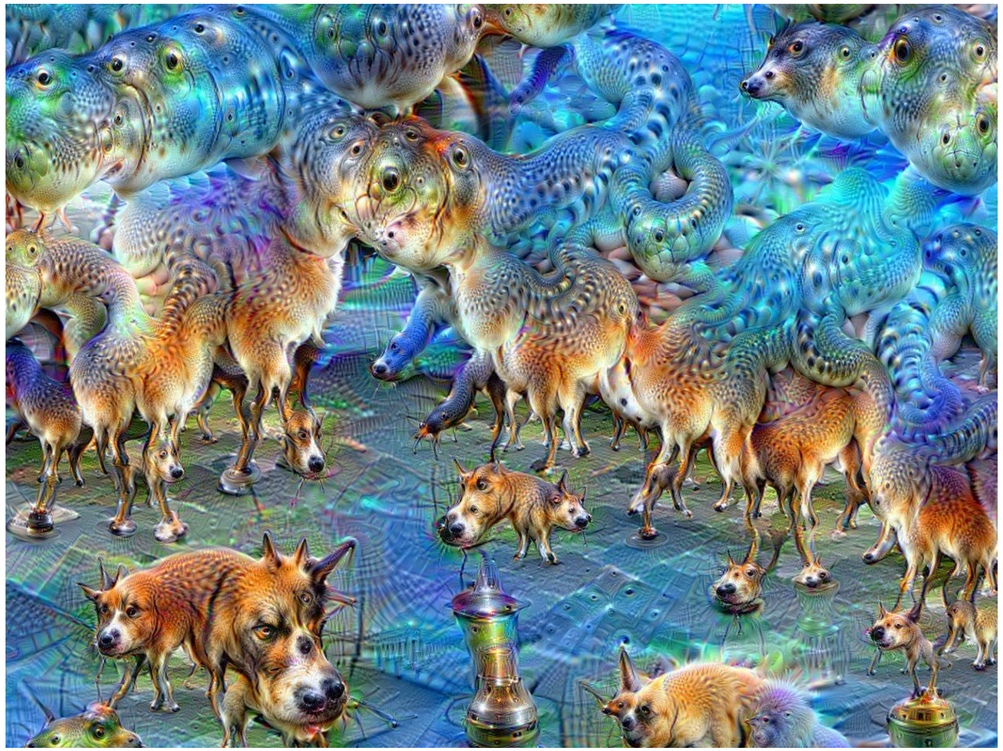
- Easier to learn and use
- For beginners and experts
- Available for everyone

You can install it today

```
!pip install tensorflow
```

Amazing things (1 of 3)

Art and science in one. Deep learning as representation learning.

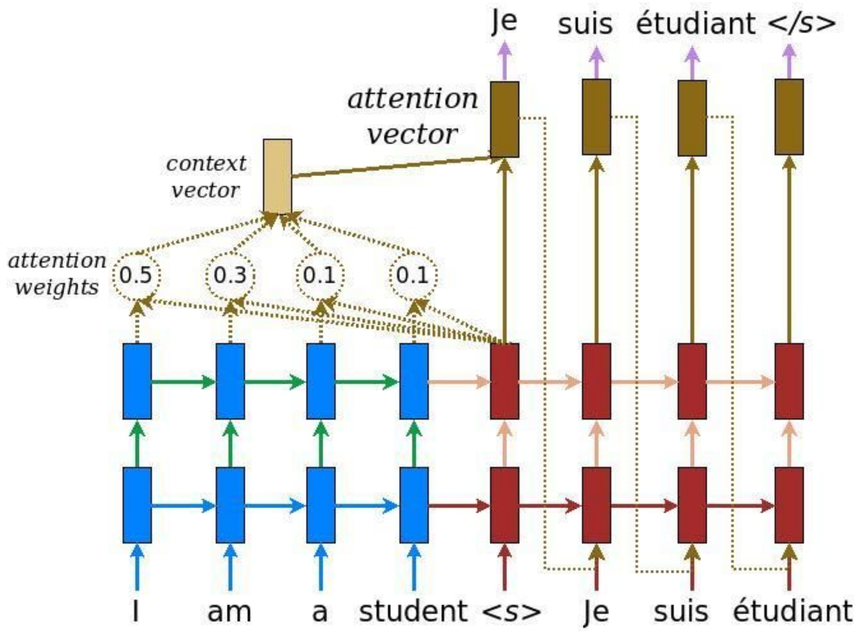


bit.ly/mini-dream
tensorflow.org/alpha/tutorials/generative/style_transfer

Amazing things (2 of 3)

Encoder / decoders. Deep learning as compression.

- Sentence -> RNN -> vector




```
sentences = [

  ("Do you want a cup of coffee?",
   "¿Quieres una taza de café?"),
    ...




]
```

bit.ly/mini-nmt
tensorflow.org/alpha/tutorials/text/nmt_with_attention

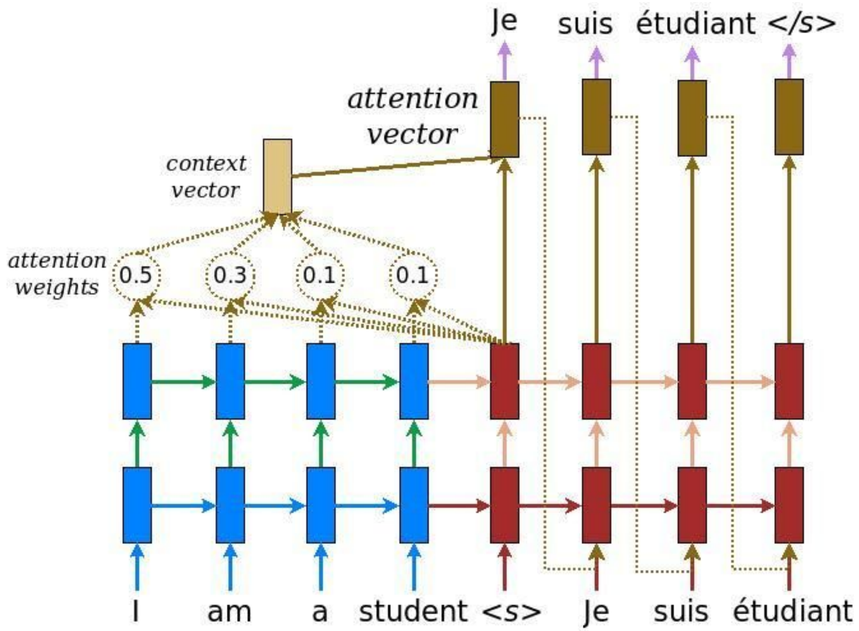
Is anyone trilingual?

Do you translate directly from ***source -> target***, or from ***source -> meaning -> target***?

tensorflow.org/alpha/tutorials/text/nmt_with_attention

Amazing things (2 of 3)

Opportunities at the intersection of linguistics and Deep Learning.

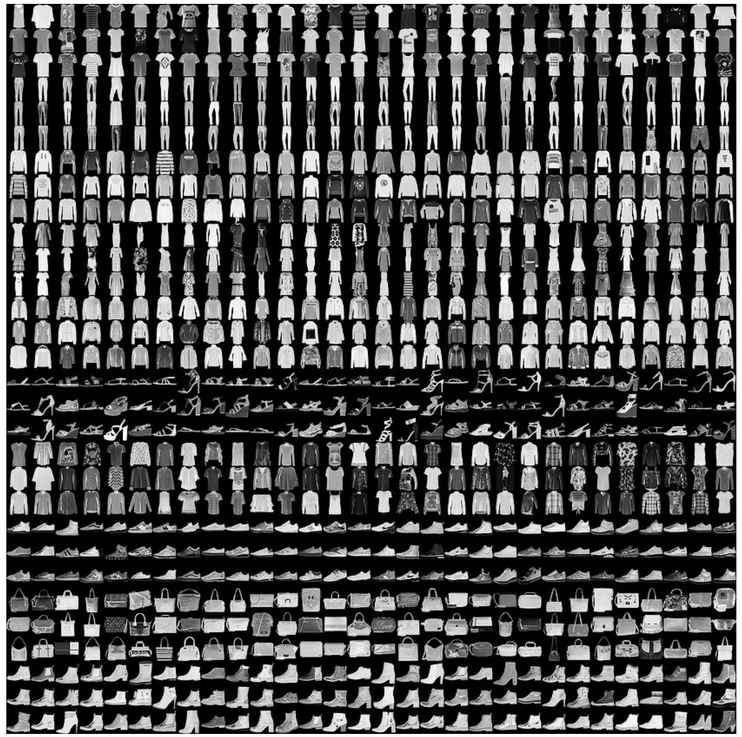


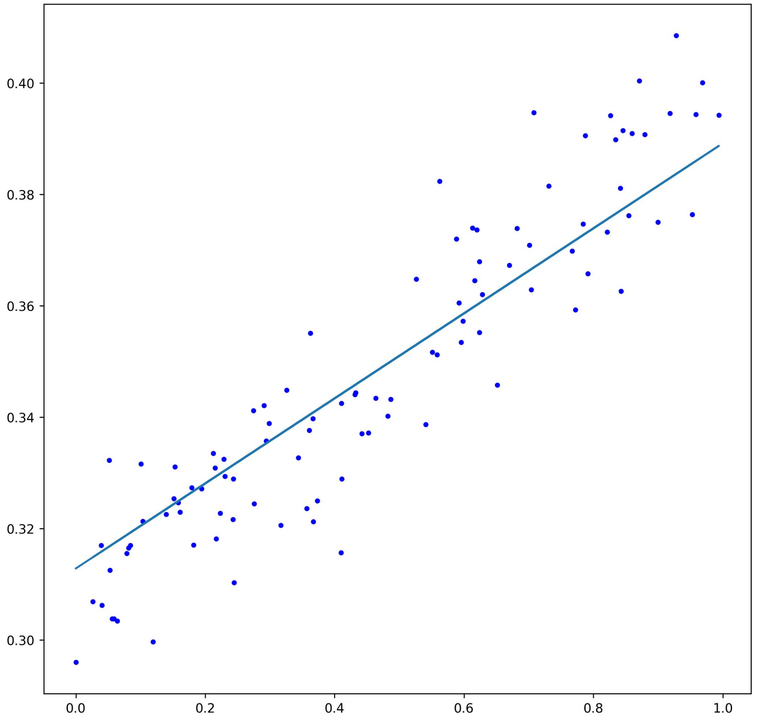

```
sentences = [

  ("Do you want a cup of coffee?",
   "¿Quieres una taza de café?"),
    ...

  (“Voulez-vous une tasse de café?”,
   “¿Quieres una taza de café?”)
    ...

]
```

bit.ly/mini-nmt

tensorflow.org/alpha/tutorials/text/nmt_with_attention

TensorFlow is great for learning (with 2.0, it's a perfect time to start)





bit.ly/tf-linear
tensorflow.org/alpha/tutorials/quickstart/beginner

```
x = tf.placeholder(tf.float32, [None, 200])

W = tf.Variable(tf.zeros([200, 10]))
b = tf.Variable(tf.zeros([10]))
y = tf.nn.softmax(tf.matmul(x, W) + b)

...

with tf.Session() as sess:
  sess.run(tf.initialize_all_variables())
  tf.train.start_queue_runners(sess)

  example_batch = tf.train.batch([x], batch_size=10, num_threads=4, capacity=10)

  max_steps = 1000
  for step in range(max_steps):
    x_in = sess.run(example_batch)
    sess.run(train_step, feed_dict={x: train_data, y_: train_labels})
    if (step % 100) == 0:
      print(step, sess.run(accuracy, feed_dict={x: test_data, y_: test_labels}))
```

We’ve learned a lot since 1.0

TensorFlow 2.0

Usability

- `tf.keras` as the recommended high-level API.
- Eager execution by default.

```
>>> tf.add(2, 3)
<tf.Tensor: id=2, shape=(), dtype=int32, numpy=5>
```

TensorFlow 2.0

Clarity

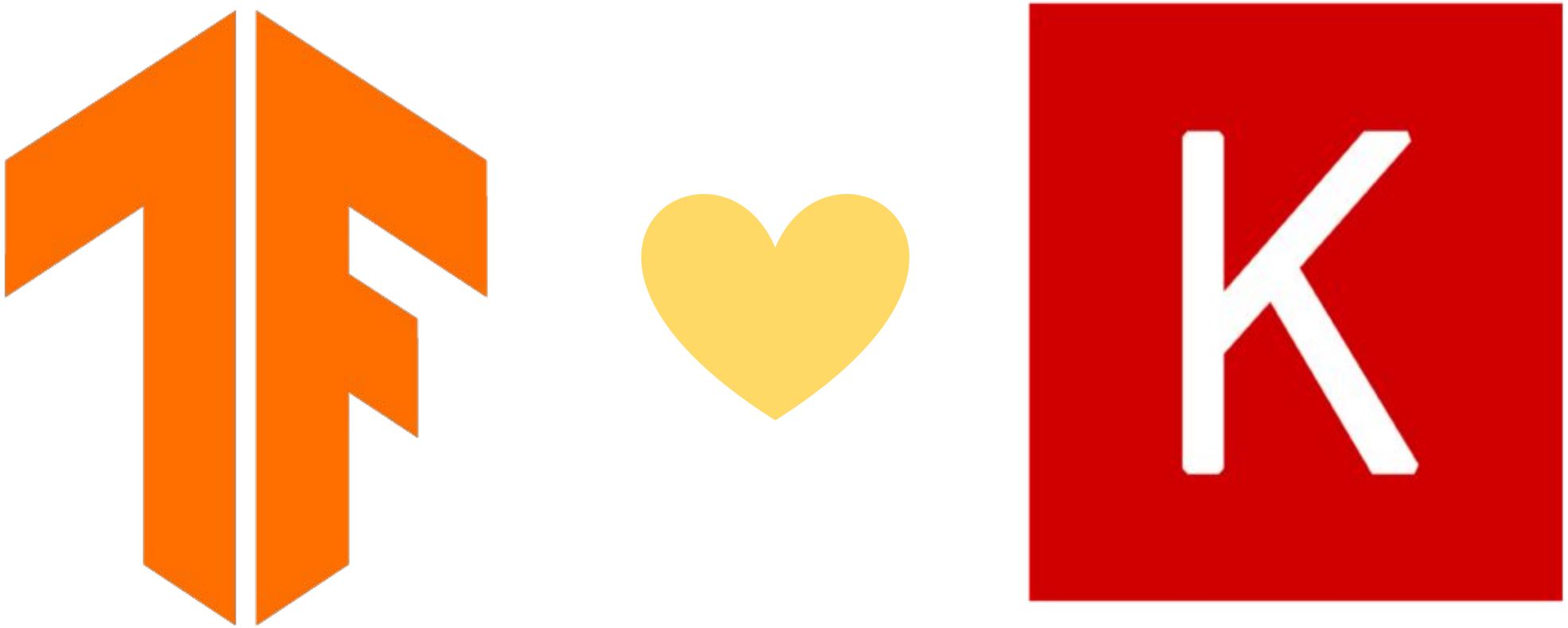
- Remove duplicate functionality
- Consistent, intuitive syntax across APIs
- Compatibility throughout the TensorFlow ecosystem

TensorFlow 2.0

Flexibility

- Full lower-level API.
- Internal ops accessible in `tf.raw_ops`
- Inheritable interfaces for variables, checkpoints, layers.

All with one API



With styles for
beginners and experts

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)
```

TF 1.x

```
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)
```

TF 2.0

```
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)
```

Technical differences

Symbolic (Sequential)

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

bit.ly/symbolic-imperative

Technical differences

Symbolic (Sequential)

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

Imperative (Subclassing)

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

bit.ly/symbolic-imperative

Training loops

Use a built-in training loop...

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

Or define your own

```
model = MyModel()

with tf.GradientTape() as tape:
  logits = model(images)
  loss_value = loss(logits, labels)

grads = tape.gradient(loss_value, model.trainable_variables)
optimizer.apply_gradients(zip(grads, model.trainable_variables))
```

Keras vs tf.keras?

keras.io

- Reference implementation
- *import keras*

tf.keras

- TensorFlow's implementation (a superset, built-in to TF)
- *from tensorflow import keras*

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'])
```

The docs are runnable

Pix2Pix

Run in Google Colab

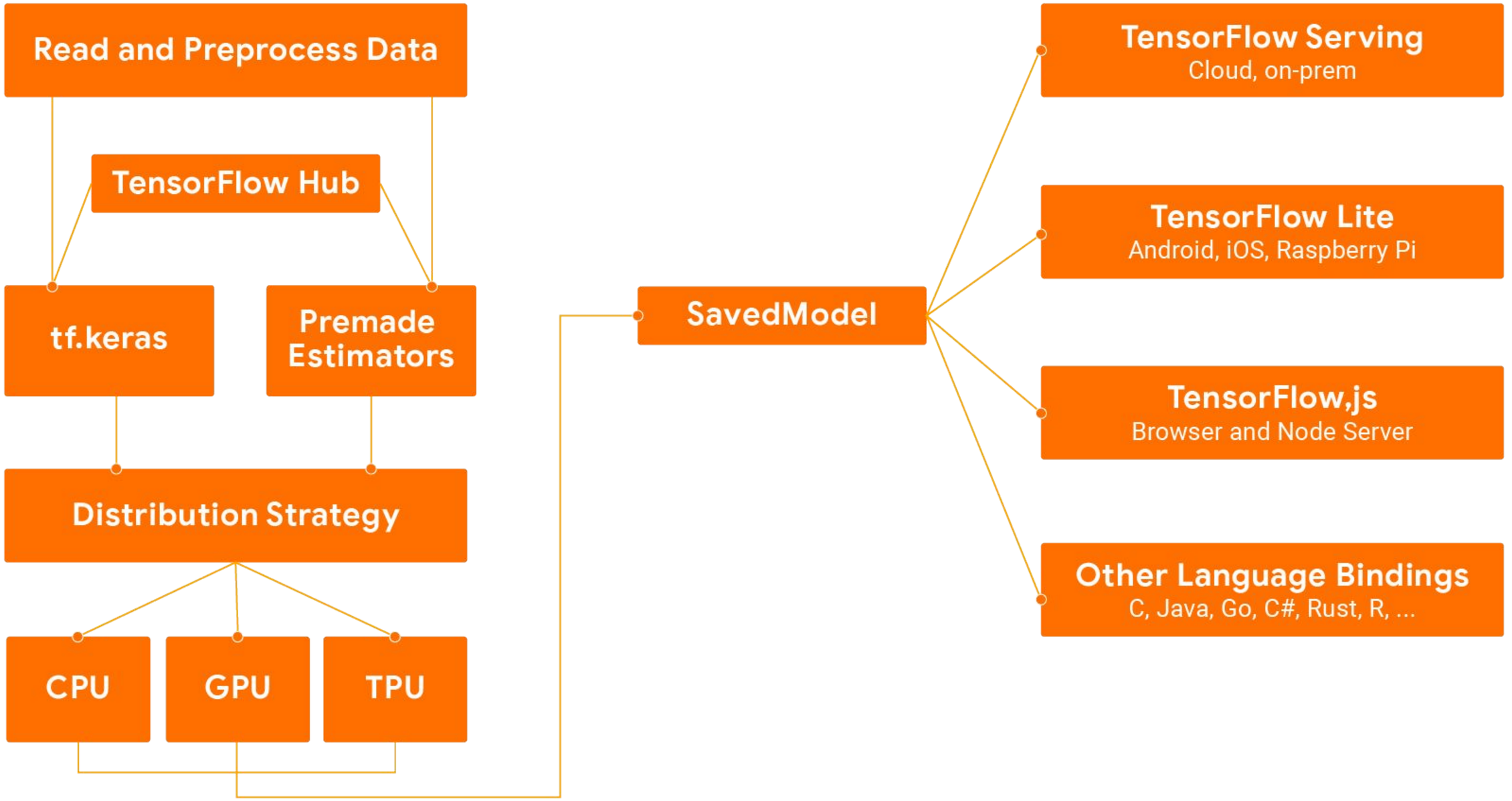
View source on GitHub

Speaking of amazing things

This notebook demonstrates image to image translation using conditional GAN's, as described in Image-to-Image Translation with Conditional Adversarial Networks. Using this technique we can colorize black and white photos, convert google maps to google earth, etc. Here, we convert building facades to real buildings.

tensorflow.org/alpha
tensorflow.org/alpha/tutorials/generative/pix2pix

Compatibility with the TensorFlow Ecosystem


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, ...

Deploy anywhere

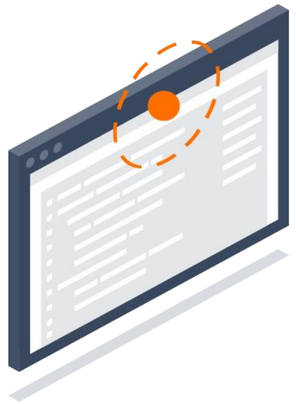
Servers

TensorFlow
Extended

Edge devices

TensorFlow
Lite

JavaScript

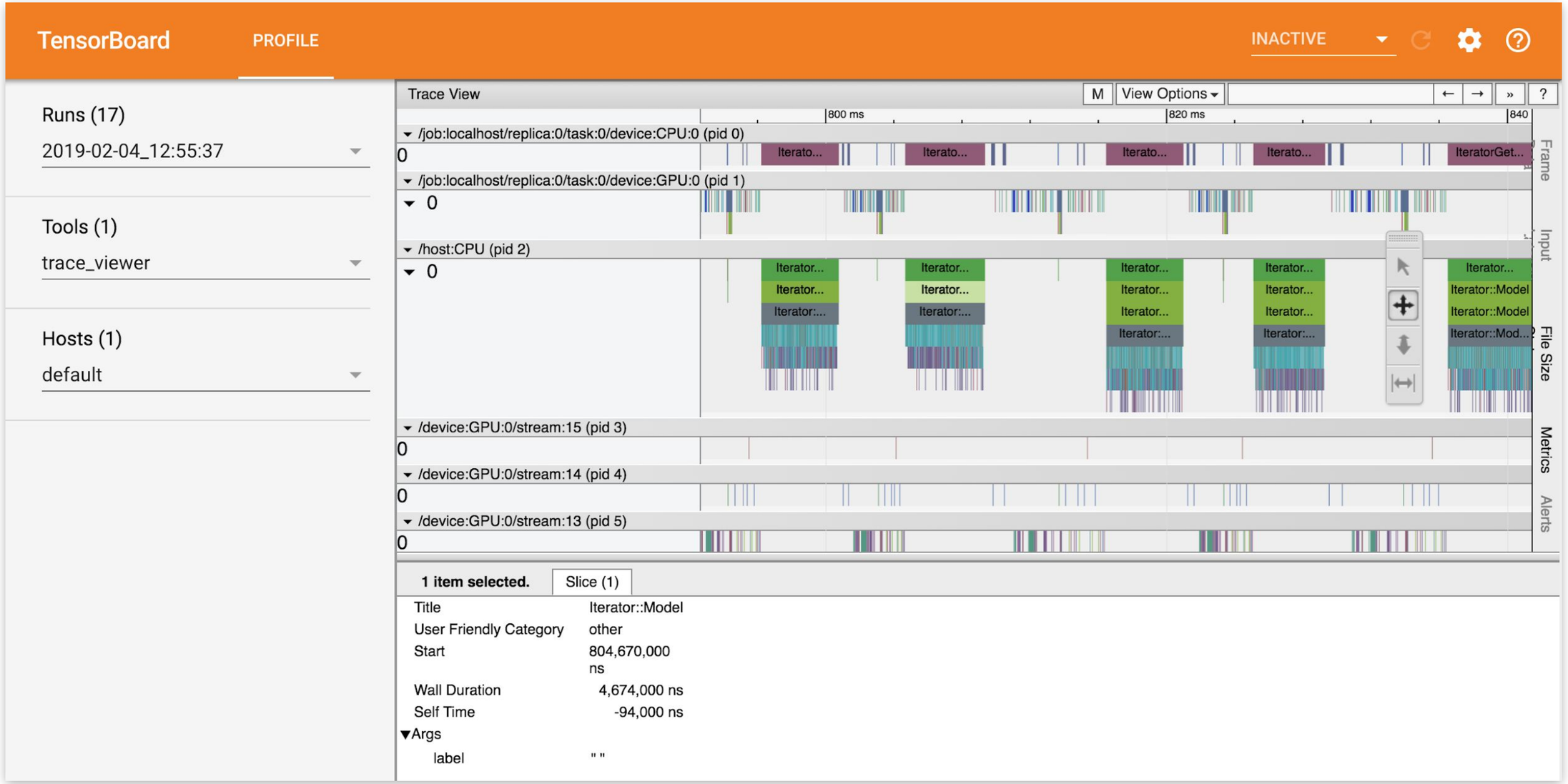


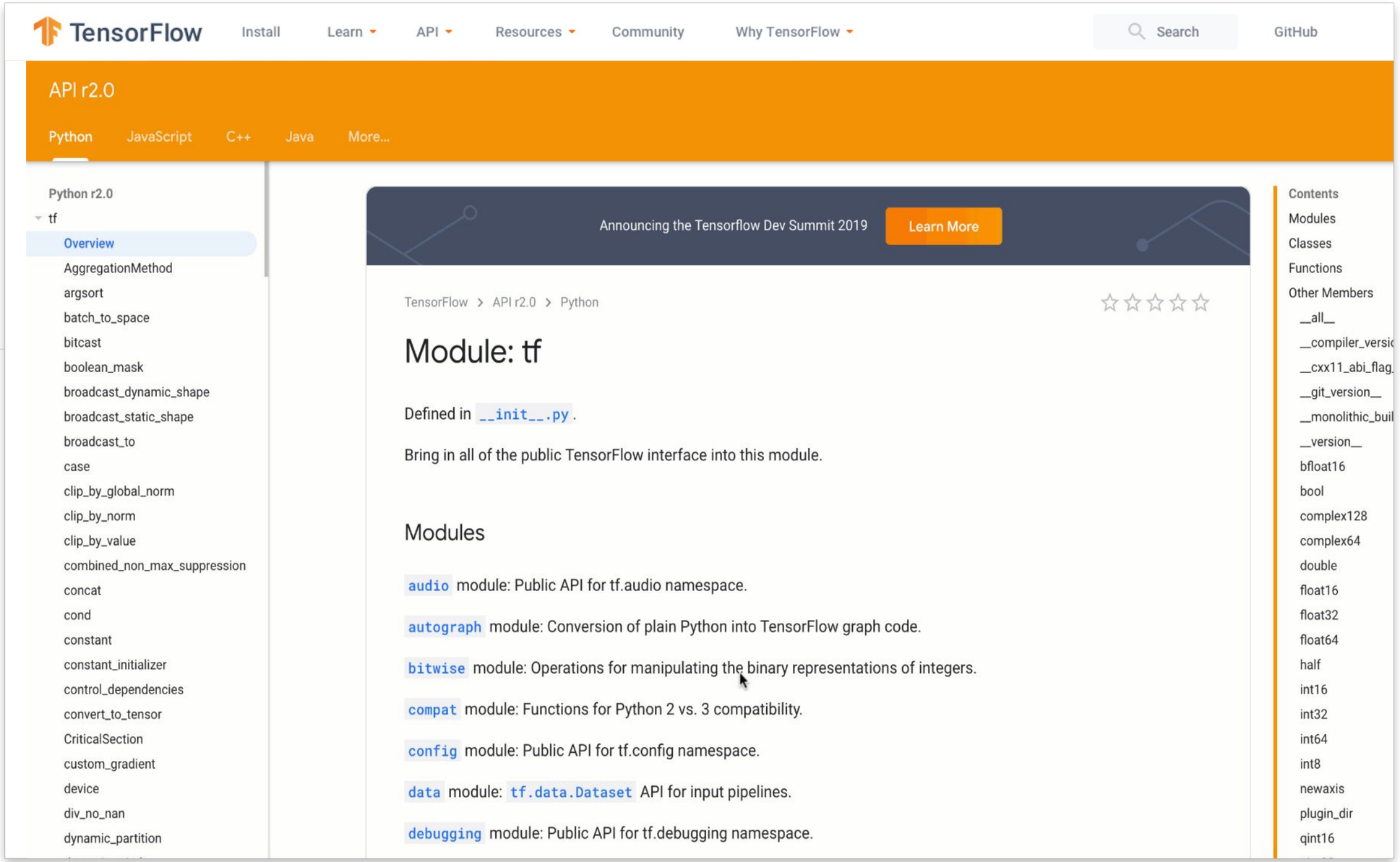
TensorFlow
.JS

Training Workflow

Data Ingestion and Transformation	Model Building	Training	Saving
tf.data Feature Columns	Keras Premade Estimators Custom	Eager Execution Autograph Distribution Strategy Tensorboard	SavedModel

Built-in performance profiling




TensorFlow
Install
Learn
API
Resources
Community
Why TensorFlow
Search
GitHub
API r2.0
Python
JavaScript
C++
Java
More...
Python r2.0
tf
Overview
AggregationMethod
argsort
batch_to_space
bitcast
boolean_mask
broadcast_dynamic_shape
broadcast_static_shape
broadcast_to
case
clip_by_global_norm
clip_by_norm
clip_by_value
combined_non_max_suppression
concat
cond
constant
constant_initializer
control_dependencies
convert_to_tensor
CriticalSection
custom_gradient
device
div_no_nan
dynamic_partition
Announcing the Tensorflow Dev Summit 2019
Learn More
TensorFlow > API r2.0 > Python
Module: tf
Defined in __init__.py.
Bring in all of the public TensorFlow interface into this module.
Modules
audio module: Public API for tf.audio namespace.
autograph module: Conversion of plain Python into TensorFlow graph code.
bitwise module: Operations for manipulating the binary representations of integers.
compat module: Functions for Python 2 vs. 3 compatibility.
config module: Public API for tf.config namespace.
data module: tf.data.Dataset API for input pipelines.
debugging module: Public API for tf.debugging namespace.
Contents
Modules
Classes
Functions
Other Members
__all__
__compiler_versi
__cxx11_abi_flag
__git_version__
__monolithic_buil
__version__
bfloat16
bool
complex128
complex64
double
float16
float32
float64
half
int16
int32
int64
int8
newaxis
plugin_dir
qint16

Developers need great performance

Since last year...

1.8x training speedup on NVIDIA Tesla V100

1.6x training speedup on Google Cloud TPU v2

3.3x inference speedup on Intel Skylake

Focused on speed

Incredible inference performance

CPU	CPU 1.9x	GPU 7.7x	Edge TPU 62x
124 ms	64ms	16 ms	2 ms
CPU on MobileNet V1	CPU w/ Quantization	Flow OpenGL 16	Quantized Fixed-point

Pixel 2 - Single Threaded CPU

TF Probability

TF Agents

Tensor2Tensor

TF Ranking

TF Text

TF Federated

TF Privacy

...

How do I upgrade?

Making upgrading easy:

- Escape to backwards compatibility module: `tf.compat.v1` (does not include `tf.contrib`)

- Migration guides and best practices

- Conversion script: `tf_upgrade_v2`

Text generation using a RNN with eager execution

View on TensorFlow.org Run in Google Colab View source on GitHub

```
[ ]  !pip install tf-nightly
```

This tutorial demonstrates how to generate text using a character-based RNN. We will work with a dataset of Shakespeare's writing from Andrej Karpathy's The Unreasonable Effectiveness of Recurrent Neural Networks. Given a sequence of characters from this data ("Shakespear"), train a model to predict the next character in the sequence ("e"). Longer sequences of text can be generated by calling the model repeatedly.

Note: Enable GPU acceleration to execute this notebook faster. In Colab: *Runtime > Change runtime type > Hardware acclerator > GPU*. If running locally make sure TensorFlow version >= 1.11.

This tutorial includes runnable code implemented using tf.keras and eager execution. The following is sample output when the model in this tutorial trained for 30 epochs, and started with the string "Q":

```
QUEENE:
I had thought thou hadst a Roman; for the oracle,
Thus by All bids the man against the word,
Which are so weak of care, by old care done;
Your children were in your holy love,
And the precipitation through the bleeding throne.

BISHOP OF ELY:
Marry, and will, my lord, to weep in such a one were prettiest;
Yet now I was adopted heir
Of the world's lamentable day,
```

TF 2.0 Upgrade Script

File Edit View Insert Runtime Tools Help

COMMENT

CODE TEXT CELL CELL

RAM Disk

Table of contents Code snippets Files

UPLOAD REFRESH

..

sample_data

text_generation.py

```
[2]  !pip install tf-nightly-2.0-preview
```

```
[3]  import tensorflow as tf
     tf.__version__
```

```
'2.0.0-dev20190218'
```

```
!tf_upgrade_v2 --infile text_generation.py --outfile text_generation_upgraded.py
```

```
!cat report.txt
```

```
[ ]
```

```
[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'
```

```
[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'
```

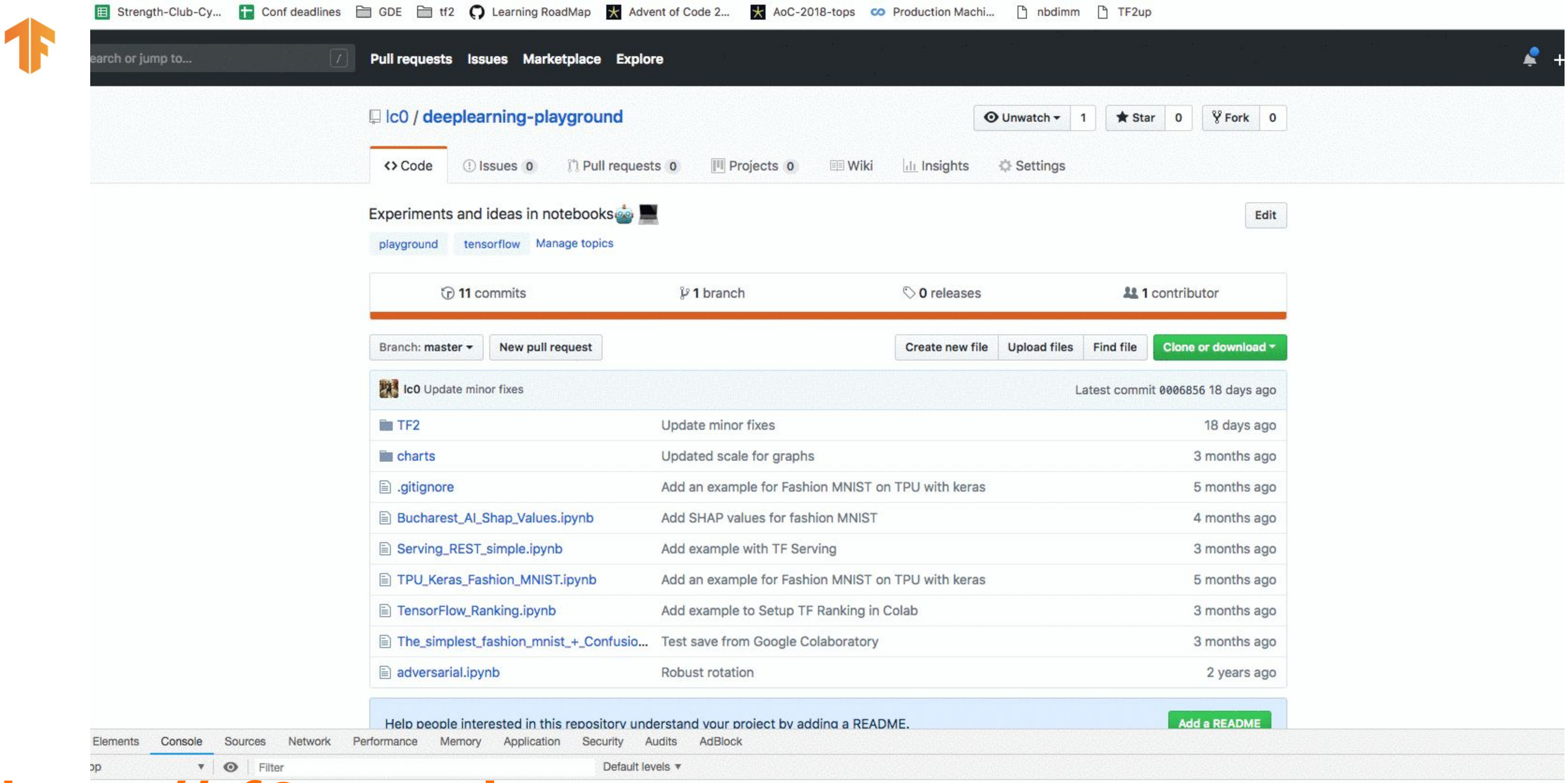
```
[4]  !tf_upgrade_v2 --infile text_generation.py --outfile text_generation_upgraded.py
```

```
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'
```

Table of contents Code snippets Files
UPLOAD REFRESH
..
sample_data
report.txt
text_generation.py
text_generation_upgraded.py

[4]

```
================================================================================
--------------------------------------------------------------------------------
Processing file 'text_generation.py'
 outputting to 'text_generation_upgraded.py'
--------------------------------------------------------------------------------

4:0: INFO: Renamed 'tf.enable_eager_execution' to 'tf.compat.v1.enable_eager_execution'
240:16: INFO: Renamed 'tf.train.AdamOptimizer' to 'tf.compat.v1.train.AdamOptimizer'
332:21: INFO: Added keywords to args of function 'tf.multinomial'
332:21: INFO: Renamed 'tf.multinomial' to 'tf.random.categorical'
375:12: INFO: Renamed 'tf.train.AdamOptimizer' to 'tf.compat.v1.train.AdamOptimizer'
392:21: INFO: tf.losses.sparse_softmax_cross_entropy requires manual check. tf.losses have been replaced with objec
392:21: INFO: Renamed 'tf.losses.sparse_softmax_cross_entropy' to 'tf.compat.v1.losses.sparse_softmax_cross_entropy
--------------------------------------------------------------------------------
```



http://tf2up.ml

Timeline

Now

Spring

2.0 alpha
cut from head, updated frequently

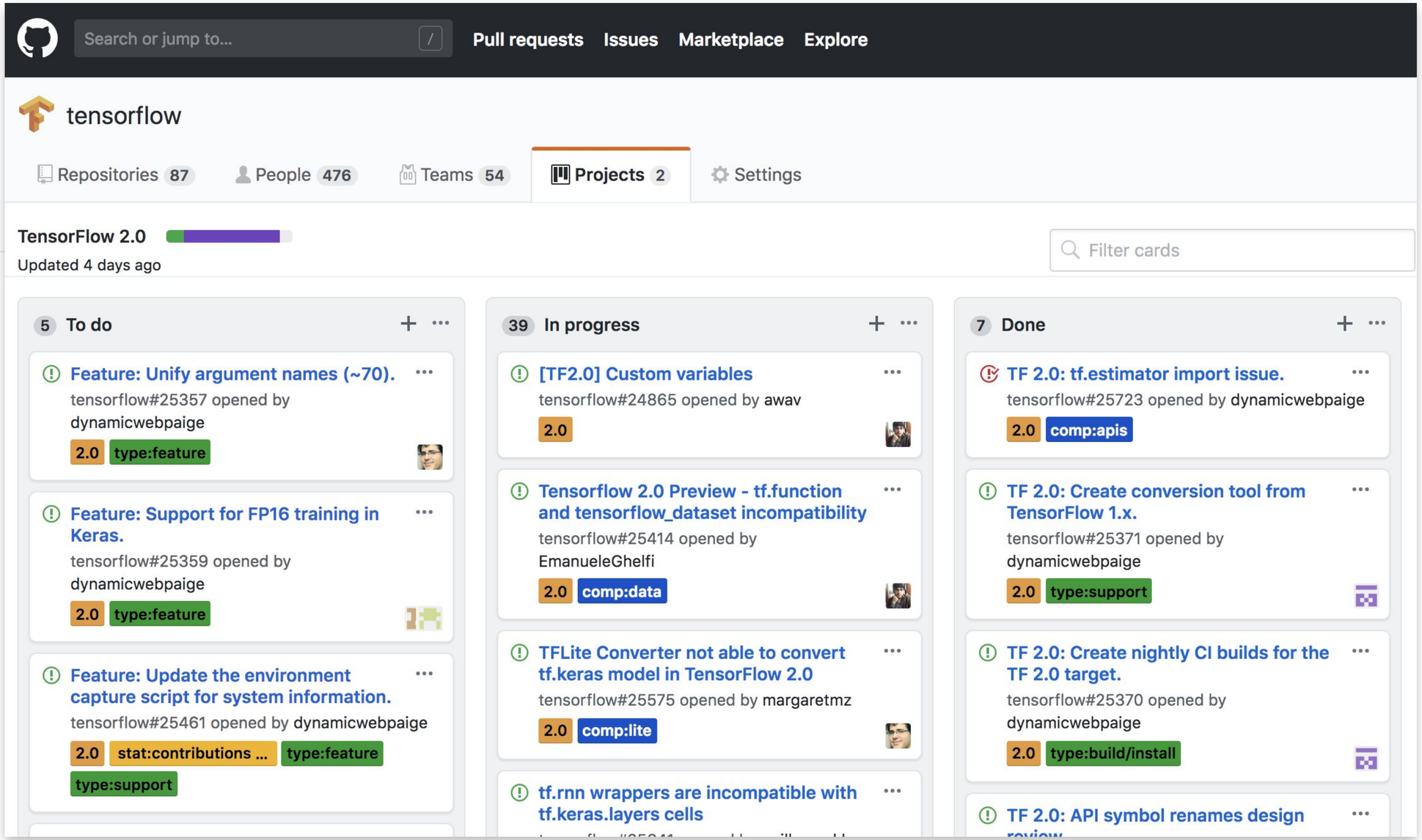
2.0 RC
(branch created)

2.0
final

- implement remaining features
- convert libraries, Google
- lots of testing, optimization

- release & integration testing

Progress


Search or jump to...
Pull requests
Issues
Marketplace
Explore
tensorflow
Repositories 87
People 476
Teams 54
Projects 2
Settings
TensorFlow 2.0
Updated 4 days ago
Filter cards
5 To do
Feature: Unify argument names (~70).
tensorflow#25357 opened by dynamicwebpaige
2.0
type:feature
Feature: Support for FP16 training in Keras.
tensorflow#25359 opened by dynamicwebpaige
2.0
type:feature
Feature: Update the environment capture script for system information.
tensorflow#25461 opened by dynamicwebpaige
2.0
stat:contributions ...
type:feature
type:support
39 In progress
[TF2.0] Custom variables
tensorflow#24865 opened by awav
2.0
Tensorflow 2.0 Preview - tf.function and tensorflow_dataset incompatibility
tensorflow#25414 opened by EmanueleGhelfi
2.0
comp:data
TFLite Converter not able to convert tf.keras model in TensorFlow 2.0
tensorflow#25575 opened by margaretmz
2.0
comp:lite
tf.rnn wrappers are incompatible with tf.keras.layers cells
7 Done
TF 2.0: tf.estimator import issue.
tensorflow#25723 opened by dynamicwebpaige
2.0
comp:apis
TF 2.0: Create conversion tool from TensorFlow 1.x.
tensorflow#25371 opened by dynamicwebpaige
2.0
type:support
TF 2.0: Create nightly CI builds for the TF 2.0 target.
tensorflow#25370 opened by dynamicwebpaige
2.0
type:build/install
TF 2.0: API symbol renames design

Under the hood

What exactly *is* TensorFlow?
What problems does it solve?

About how much slower is Python than C?

- Multiplying matrices: +/- 100X
- 6 seconds vs. 10 minutes
- Running vs. flying (6 MPH and 600 MPH)

Python is *incredibly popular* for scientific computing

- Why? NumPy!
- C performance, Python ease of use

TensorFlow is a similar idea

- A C++ backend with a Python frontend

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'>
```

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
```

Learning more

Tutorials

tensorflow.org/beta

Tip: Ways to know you're using a 1.x tutorial

- `tf.enable_eager_execution()`
- `session.run`
- `tf.placeholder`
- `feed_dict`

Books

Two of our favorites

- Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow 2.0, by Aurélien Géron

- Deep Learning with Python, by François Chollet

Thank you!

Abderrahman JAIZE
twitter.com/jaizemsd