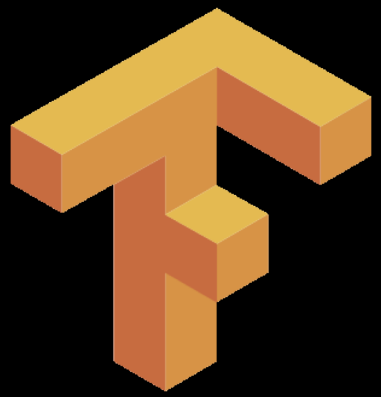
TensorFlow & Keras

An Introduction

(Some of the contents on these slides, along with the template, have been adopted from William Guss (ex TA) and CS 224 and CS20 at Stanford)

Deep Learning Frameworks

- Scale ML code
- Compute Gradients!
- Standardize ML applications for sharing
- Interface with GPUs for parallel processing

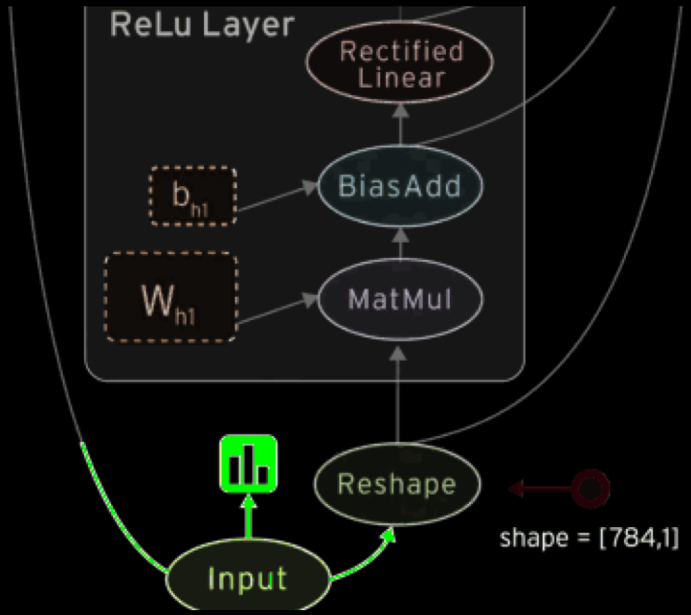


TensorFlow

What is TensorFlow?

TensorFlow is a graph computation framework for deep learning.

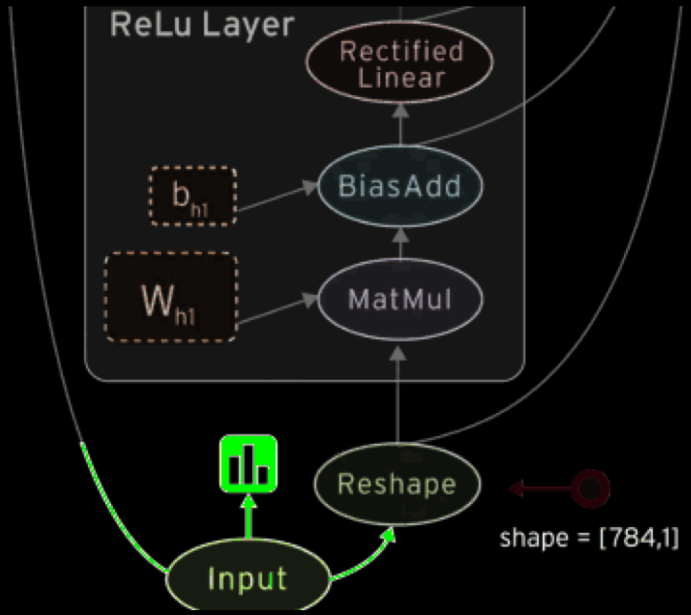
Originally developed by Google Brain Team to conduct ML research



What is TensorFlow?

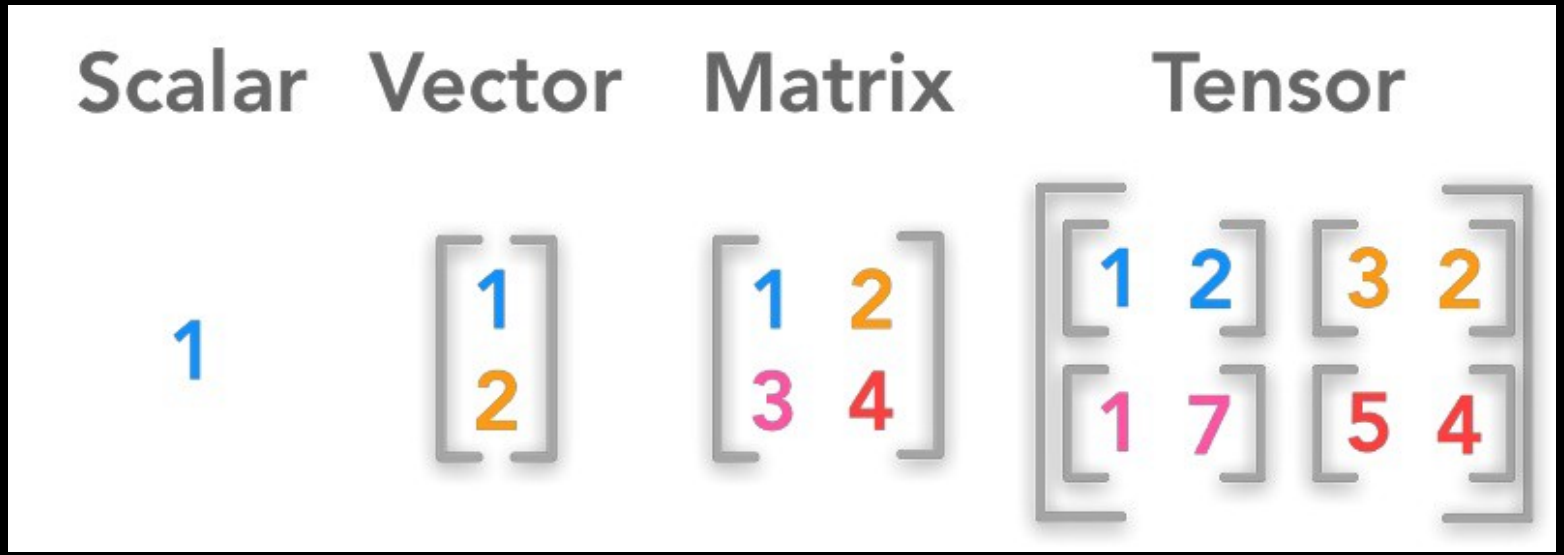
TensorFlow allows for the specification and optimization of complex feed-forward models.

In particular, TensorFlow automatically differentiates a specified model.



What is a tensor?

- An n-dimensional Array



Why TensorFlow?

- Python API
- Portability: deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API
- Visualization: TensorBoard
- Auto-differentiation
- Large community (> 10,000 commits and > 3000 TF-related repos in 1 year)
- Awesome projects already using TensorFlow

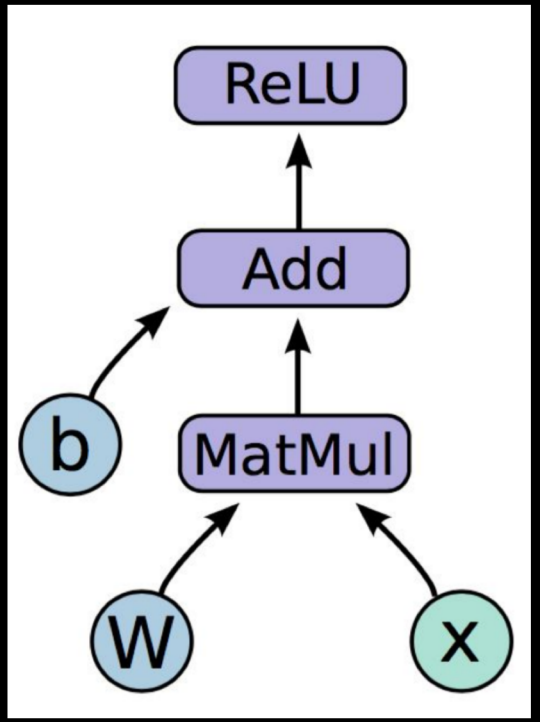
Programming Model

Big idea: express a numeric computation as a graph

- Graph nodes are operations which have any number of inputs and outputs
- Graph edges are tensors which flow between nodes

Programming Model

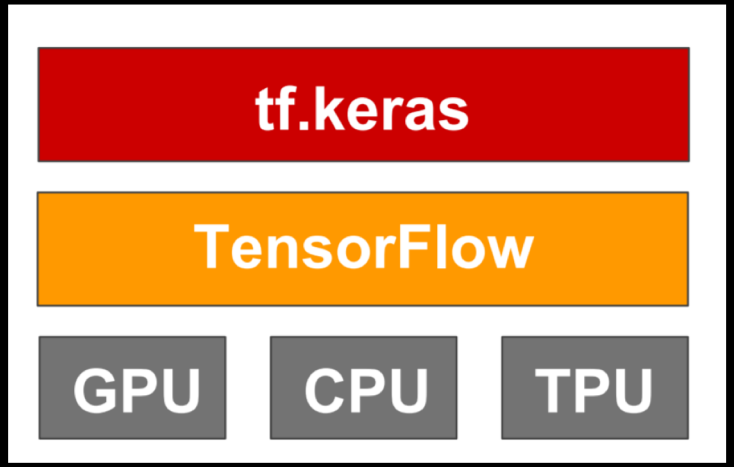
$h = ReLU(Wx + b)$



Keras

Keras is the official high-level API of TensorFlow

- tensorflow.keras (tf.keras) module
- Part of core TensorFlow since v1.4
- Full Keras API
- Better optimized for TF
- Better integration with TF-specific features



What's special about Keras?

- A focus on user experience.
- Large adoption in the industry and research community.
- Multi-backend, multi-platform.
- Easy productization of models.

User Experience

Keras is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear and actionable feedback upon user error.

This makes Keras easy to learn and easy to use. As a Keras user, you are more productive, allowing you to try more ideas than your competition, faster -- which in turn helps you win machine learning competitions.

This ease of use does not come at the cost of reduced flexibility: because Keras integrates with lower-level deep learning languages (in particular TensorFlow), it enables you to implement anything you could have built in the base language. In particular, as tf.keras, the Keras API integrates seamlessly with your TensorFlow workflows.

Using Keras

Three API Styles

- The Sequential Model
  - Dead simple
  - 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 graph topologies
  - Good for 95% of use cases

- Model Subclassing
  - Maximum flexibility
  - Larger potential error surface

The Sequential API

```
import keras
from keras import layers

model = keras.Sequential()
model.add(layers.Dense(20, activation='relu', input_shape=(10,)))
model.add(layers.Dense(20, activation='relu'))
model.add(layers.Dense(10, activation='softmax'))

model.fit(x, y, epochs=10, batch_size=32)
```

The functional API

```
import keras
from keras import layers

inputs = keras.Input(shape=(10,))
x = layers.Dense(20, activation='relu')(x)
x = layers.Dense(20, activation='relu')(x)
outputs = layers.Dense(10, activation='softmax')(x)

model = keras.Model(inputs, outputs)
model.fit(x, y, epochs=10, batch_size=32)
```