What's TensorFlow™?

- Open source software library for numerical computation using data flow graphs
- Originally developed by Google Brain Team to conduct machine learning and deep neural networks research
- General enough to be applicable in a wide variety of other domains as well

TensorFlow provides an extensive suite of functions and classes that allow users to build various models from scratch.

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
- Flexibility: from Raspberry Pi, Android, Windows, iOS, Linux to server farms
- Visualization (TensorBoard is da bomb)
- Checkpoints (for managing experiments)
- Auto-differentiation *autodiff* (no more taking derivatives by hand. Yay)
- Large community (> 10,000 commits and > 3000 TF-related repos in 1 year)
- Awesome projects already using TensorFlow

Companies using Tensorflow

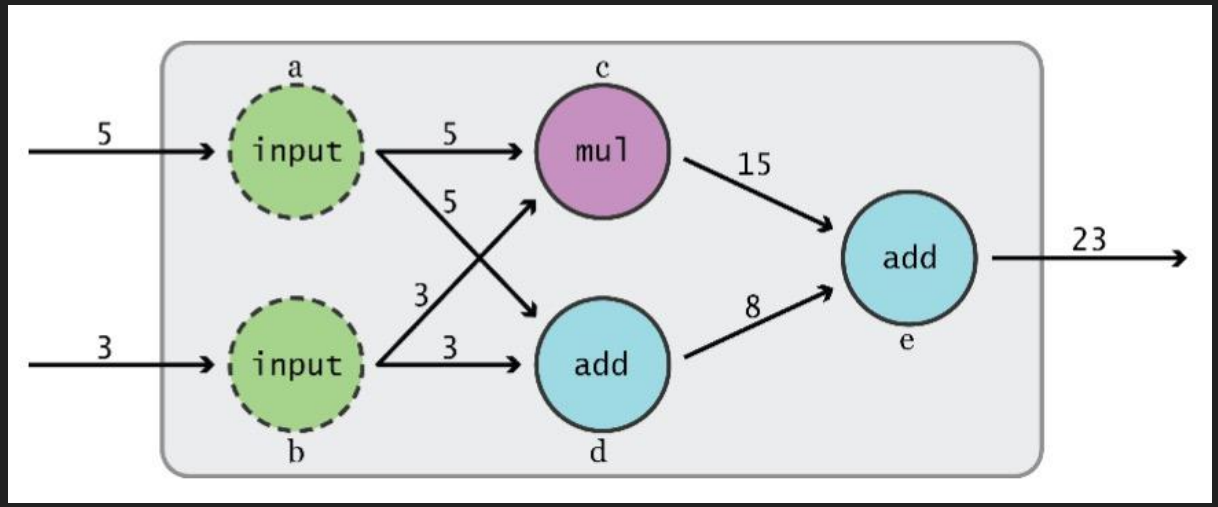
- Google
- OpenAI
- DeepMind
- Snapchat
- Uber
- Airbus
- eBay
- Dropbox
- A bunch of startups

```
import tensorflow as tf
```

Graphs and Sessions

Data Flow Graphs

TensorFlow separates definition of computations from their execution

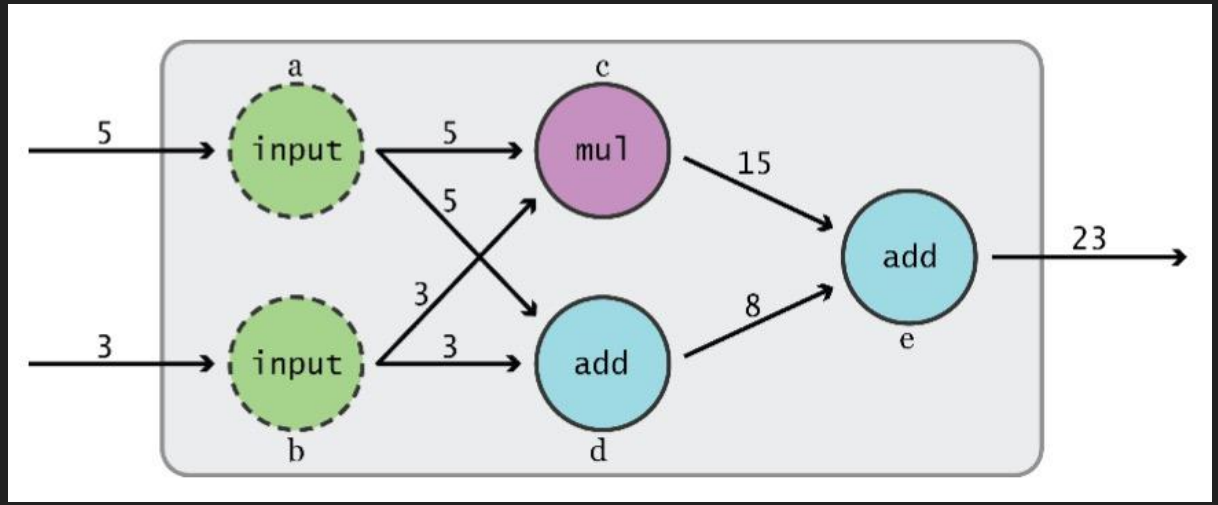


Graph by TFFMI

Data Flow Graphs

Phase 1: assemble a graph

Phase 2: use a session to execute operations in the graph.



Graph by TFFMI

What’s a tensor?

What's a tensor?

An n-dimensional array

0-d tensor: scalar (number)

1-d tensor: vector

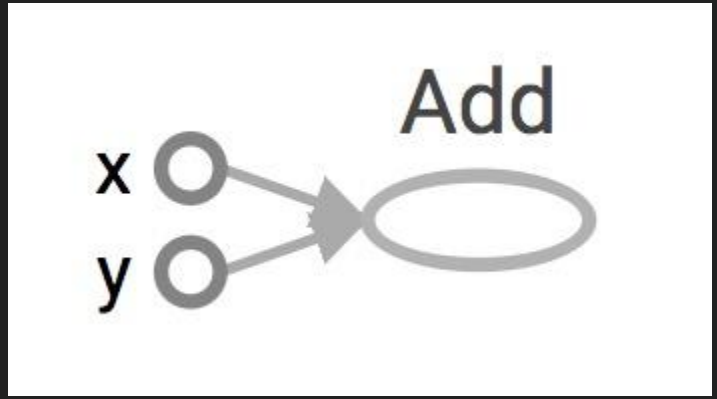
2-d tensor: matrix

and so on

Data Flow Graphs

```
import tensorflow as tf

a = tf.add(3, 5)
```

Visualized by TensorBoard



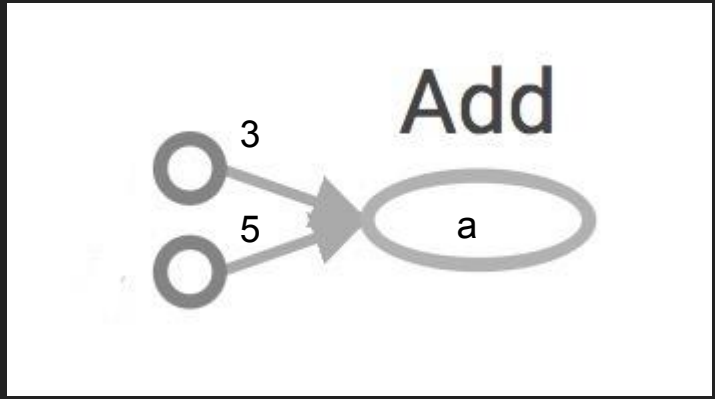
Why x, y?

TF automatically names the nodes when you don't explicitly name them.
x = 3
y = 5

Data Flow Graphs

```
import tensorflow as tf

a = tf.add(3, 5)
```

Interpreted?

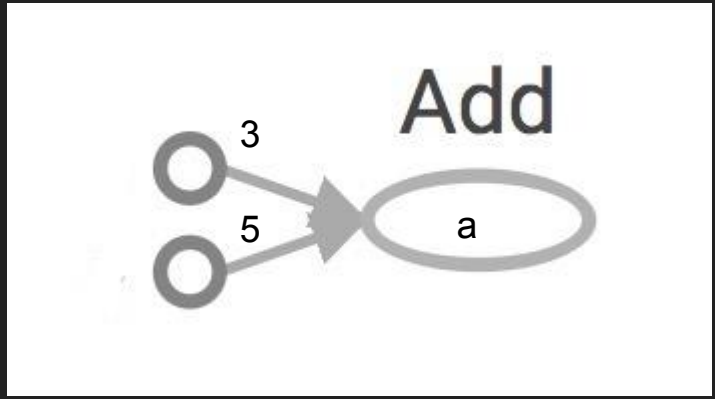


Nodes: operators, variables, and constants
Edges: tensors

Data Flow Graphs

```
import tensorflow as tf

a = tf.add(3, 5)
```

Interpreted?

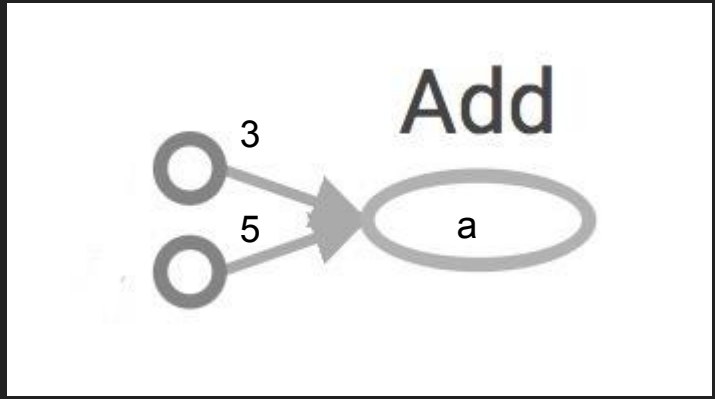


Nodes: operators, variables, and constants
Edges: tensors

Tensors are data.
Data Flow -> Tensor Flow (I know, mind=blown)

Data Flow Graphs

```
import tensorflow as tf

a = tf.add(3, 5)

print  a
```




```
>> Tensor("Add:0", shape=(), dtype=int32)
```

(Not 8)

How to get the value of a?

Create a **session**, assign it to variable sess so we can call it later

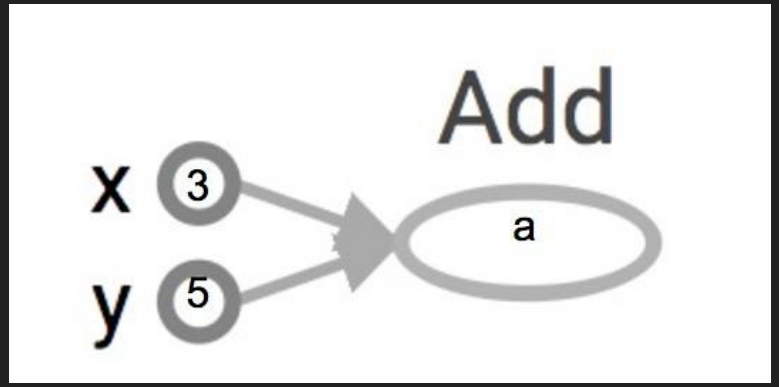
Within the session, evaluate the graph to fetch the value of a

How to get the value of a?

Create a **session**, assign it to variable sess so we can call it later

Within the session, evaluate the graph to fetch the value of a

```
import tensorflow as tf

a = tf.add(3, 5)

sess = tf.Session()

print sess.run(a)  tf.print(a)

sess.close()
```



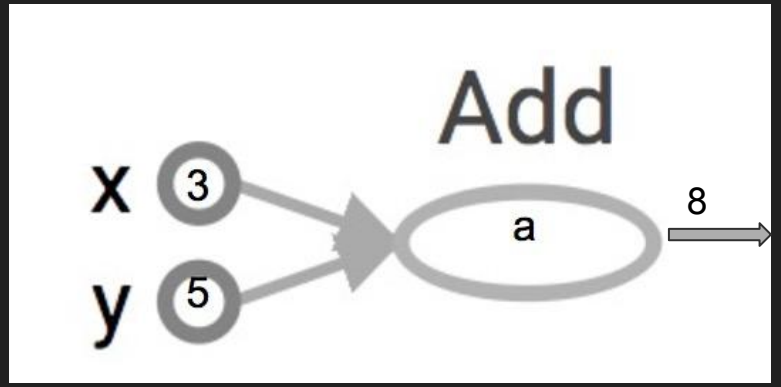
The session will look at the graph, trying to think: hmm, how can I get the value of a, then it computes all the nodes that leads to a.

How to get the value of a?

Create a **session**, assign it to variable sess so we can call it later

Within the session, evaluate the graph to fetch the value of a

```
import tensorflow as tf

a = tf.add(3, 5)

sess = tf.Session()

print sess.run(a)        >> 8

sess.close()
```



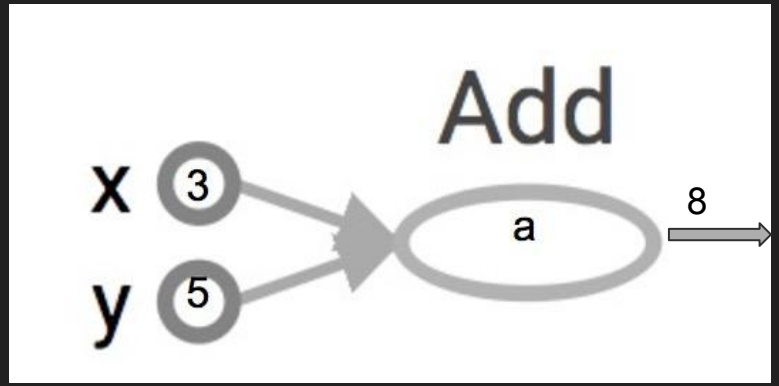
The session will look at the graph, trying to think: hmm, how can I get the value of a, then it computes all the nodes that leads to a.

How to get the value of a?

Create a **session**, assign it to variable sess so we can call it later

Within the session, evaluate the graph to fetch the value of a

```
import tensorflow as tf

a = tf.add(3, 5)

sess = tf.Session()

with tf.Session() as sess:

      print sess.run(a)

sess.close()
```



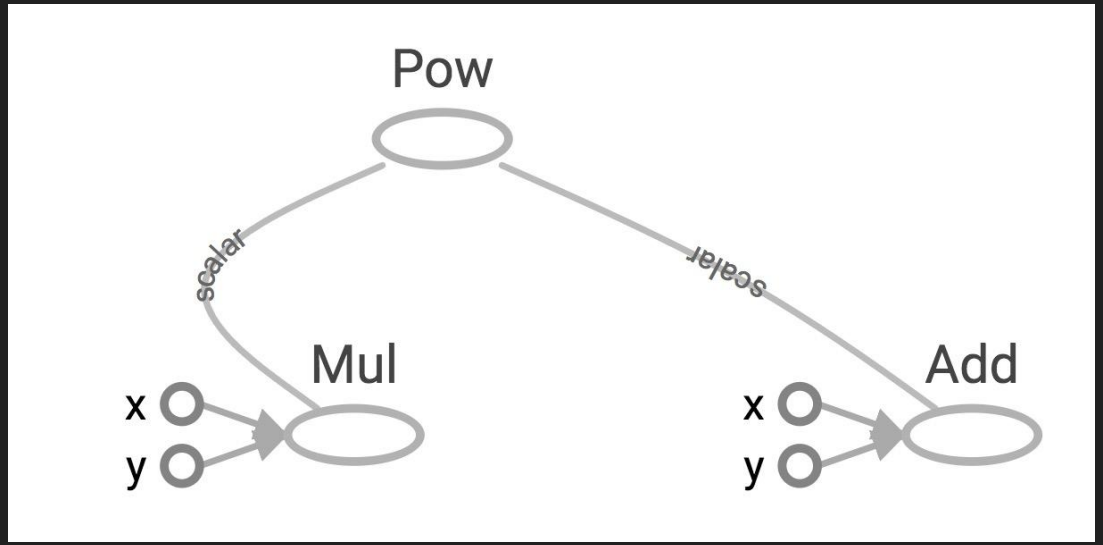
tf.Session()

A Session object encapsulates the environment in which Operation objects are executed, and Tensor objects are evaluated.

More graphs

```
x = 2

y = 3

op1 = tf.add(x, y)

op2 = tf.mul(x, y)

op3 = tf.pow(op2, op1)

with tf.Session() as sess:

      op3 = sess.run(op3)
```

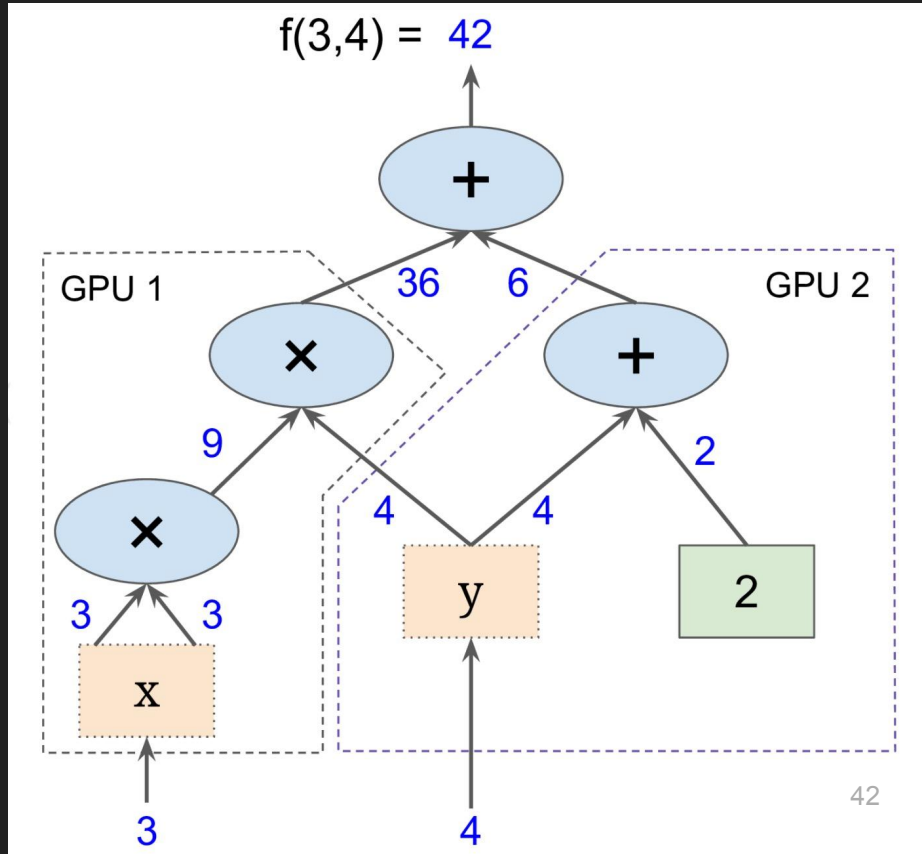
Visualized by TensorBoard



Subgraphs

Possible to break graphs into several chunks and run them parallelly across multiple CPUs, GPUs, or devices

Example: AlexNet



Graph from the book "Hands-On Machine Learning with Scikit-Learn and TensorFlow"

Why graphs

1. Save computation (only run subgraphs that lead to the values you want to fetch)
2. Break computation into small, differential pieces to facilitates auto-differentiation
3. Facilitate distributed computation, spread the work across multiple CPUs, GPUs, or devices
4. Many common machine learning models are commonly taught and visualized as directed graphs already

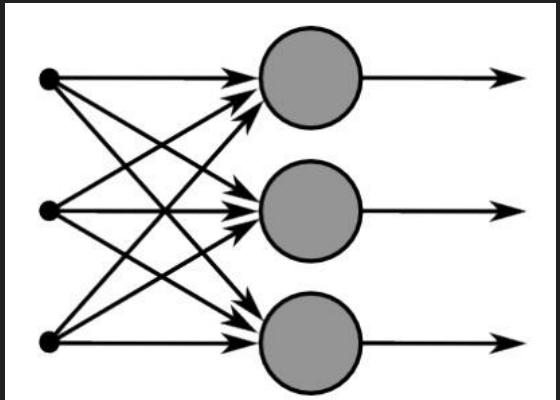


Figure 3: This image captures how multiple sigmoid units are stacked on the right, all of which receive the same input x.

A neural net graph by Richard Socher (CS224D)