



Introducción a TensorFlow

Roberto Muñoz

Doctor en Astrofísica

Director Centro I+D MetricArts

 @RobertoKPax

METRICARTS

Temario

- ¿Qué es TensorFlow?
- ¿Porqué debería usarlo?
- ¿Cómo funciona?



TensorFlow

TensorFlow

Google TensorFlow

- Originalmente desarrollado por el Brain team de Google dentro la unidad de investigación Machine Intelligence de Google
- TensorFlow entrega primitivas para crear funciones sobre tensores y automáticamente calcular sus derivadas
- Librería de software abierto para hacer cálculos numéricos usando grafos de datos



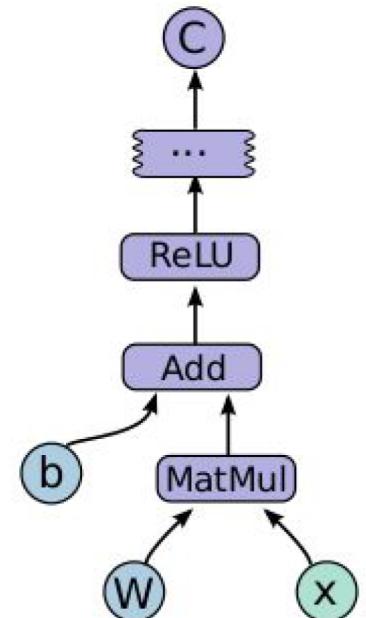
Tensor

- Los tensores pueden entenderse como un arreglo n-dimensional de números
 - Un escalar es un tensor
 - Un vector es un tensor
 - Una matriz es un tensor

$$T = \begin{matrix} X_{11N} & X_{12N} & X_{13N} & \cdots & X_{1NN} \\ X_{112} & X_{122} & X_{132} & \cdots & X_{1N2} \\ X_{111} & X_{121} & X_{131} & \cdots & X_{1N1} \\ X_{211} & X_{221} & X_{231} & \cdots & X_{2N1} \\ \vdots & \vdots & \vdots & & \vdots \\ X_{N11} & X_{N21} & X_{N31} & \cdots & X_{NN1} \end{matrix}$$

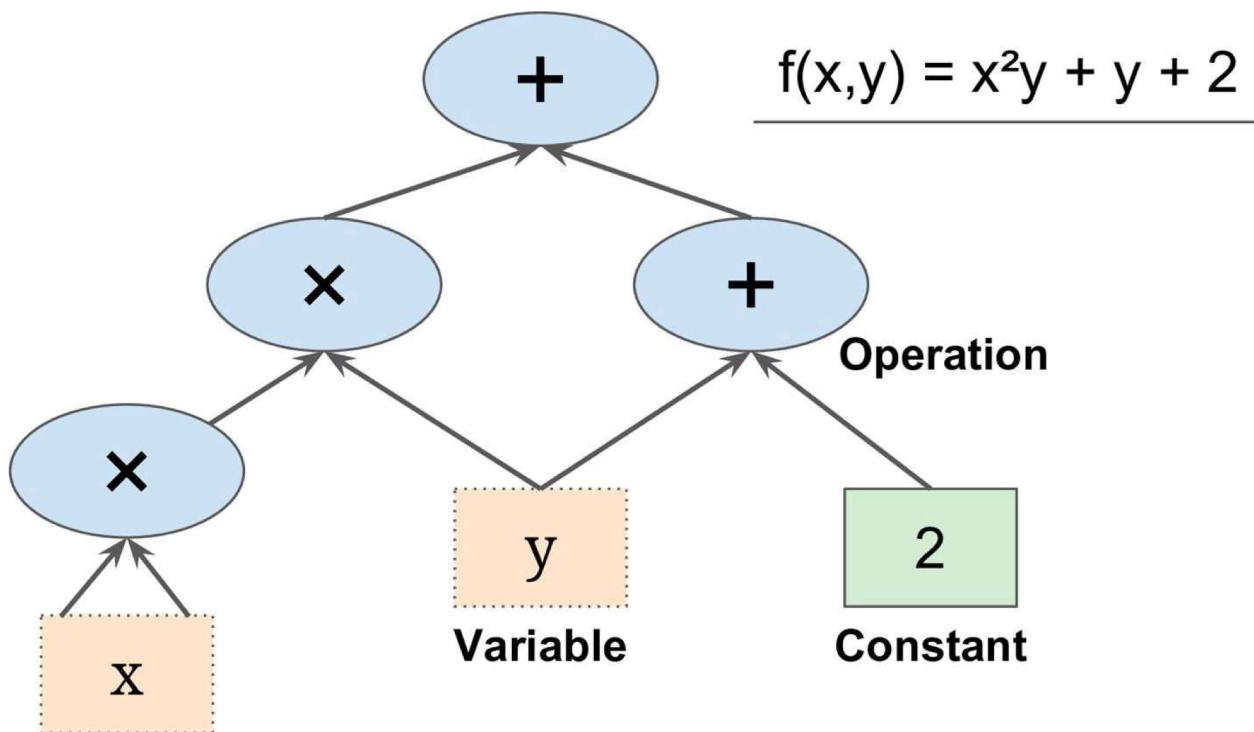
Grafo de flujo de datos

- Cálculos son representados como grafos
 - Los nodos son las operaciones
 - Los extremos son los Tensores
- Un programa típico consiste de 2 fases
 - Fase de construcción: Ensamblar un grafo (modelo)
 - Fase de ejecución: Empujar datos a través del grafo



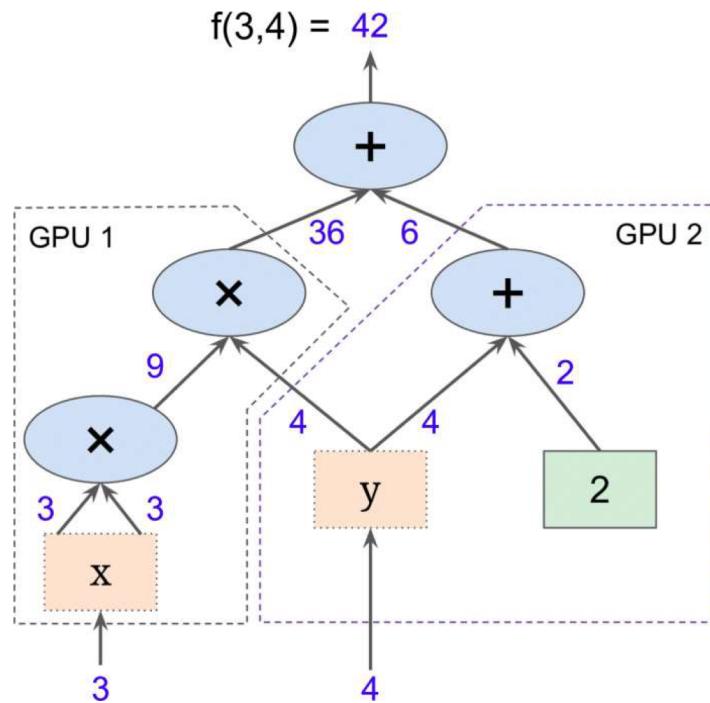
Construcción de grafos

- El grafo se construye en Python y luego se ejecuta de manera eficiente en C++



Construcción de grafos

- El grafo puede ser dividido en múltiples partes
- Cada parte puede correr de manera paralela en múltiples CPUs y GPUs

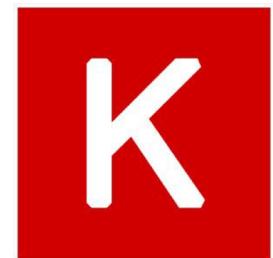


Cross platform

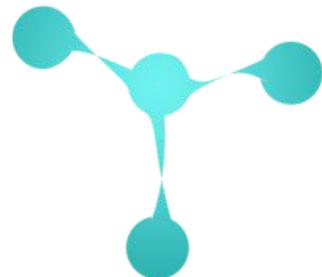
- TensorFlow puede correr en múltiples sistemas operativos
 - Windows
 - Linux
 - macOS
 - iOS
 - Android

Frameworks de DL

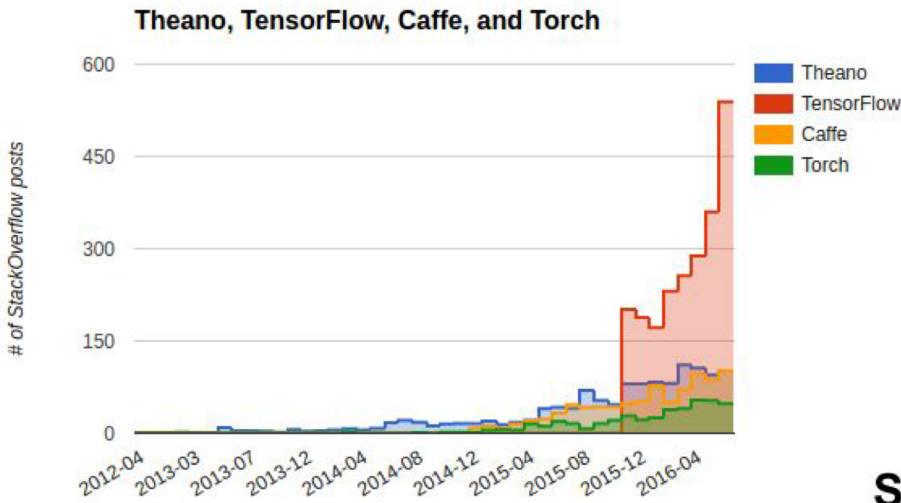
- Hay varias alternativas
 - PyTorch
 - Caffe
 - Theano
 - CuDNN
 - Mxnet
 - Darknet
 - CNTK
 - DL4J



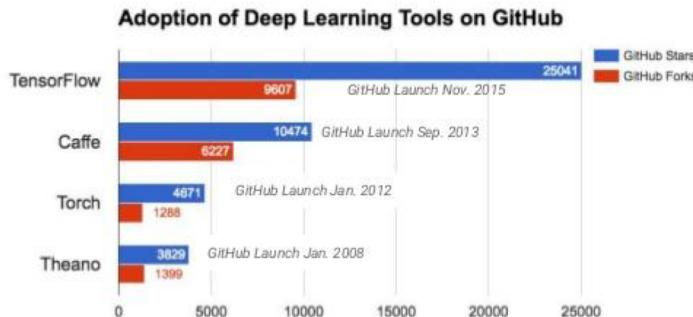
theano



Comunidad de TensorFlow



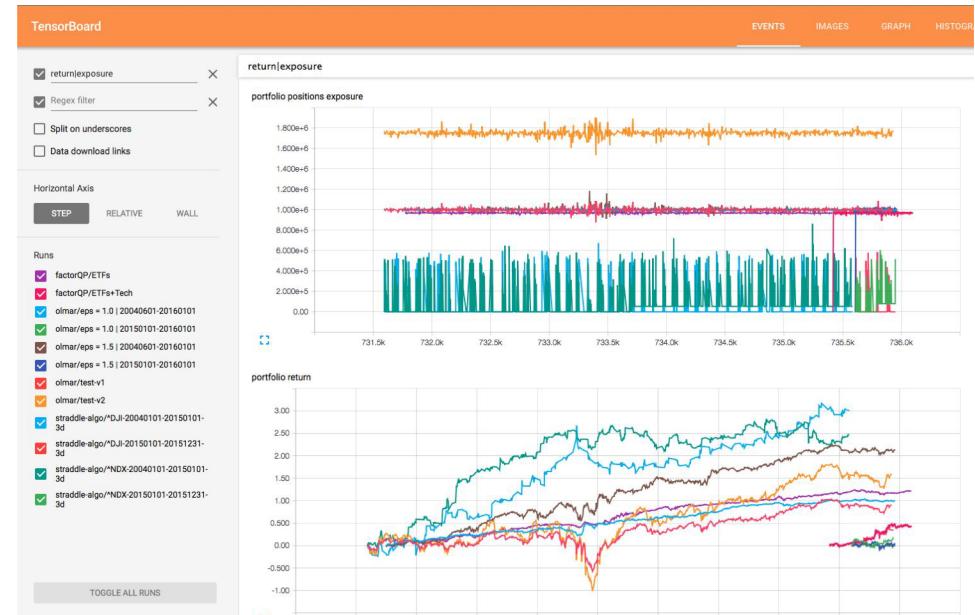
Strong External Adoption



50,000+ binary installs in 72 hours, 500,000+ since Nov, 2015
Most forks of any GitHub repo in 2015, despite only being available starting in Nov, 2015
(source: <http://donnemartin.com/viz/pages/2015>)

TensorFlow y más

- Funcionalidades específicas para hacer deployment (TF serving, CloudML)
- Documentación extensiva (Python)
- Herramienta de visualización (Tensorboard)



Ejemplo de grafo

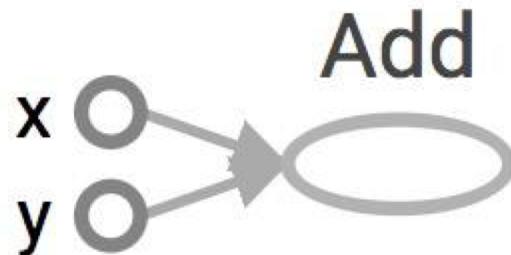
```
② import tensorflow as tf  
② a = tf.add(3, 5)
```

¿Porqué x, y?

TF automáticamente nombra los nodos cuando no se definen explícitamente

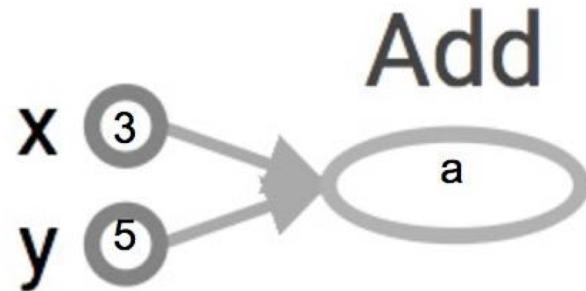
x=4

y=5



Ejemplo de grafo

```
② import tensorflow as tf  
② a = tf.add(3, 5)
```



Nodos: Operadores, Variables y Constantes

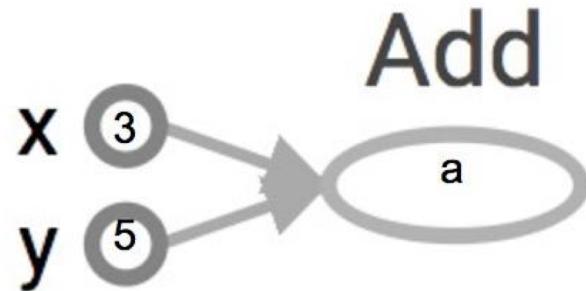
Extremos: Tensores

Tensores son datos

Data flow -> Tensor flow

Antes de eager

```
[?] import tensorflow as tf  
[?] a = tf.add(3, 5)  
[?] print(a)
```

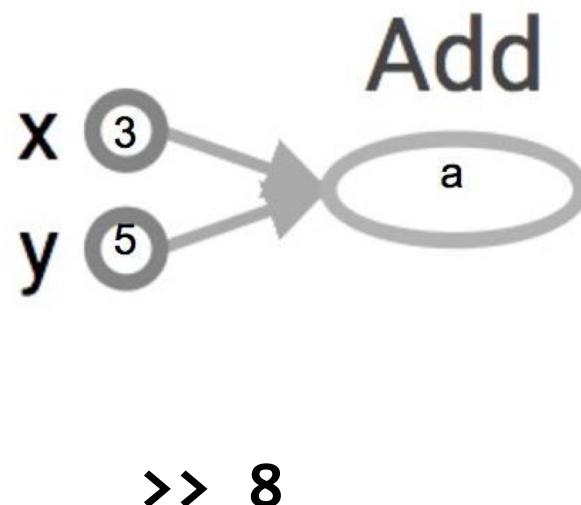


```
>> Tensor("Add:0", shape=(), dtype=int32)  
(Esperábamos 8)
```

Cómo obtengo el valor de a

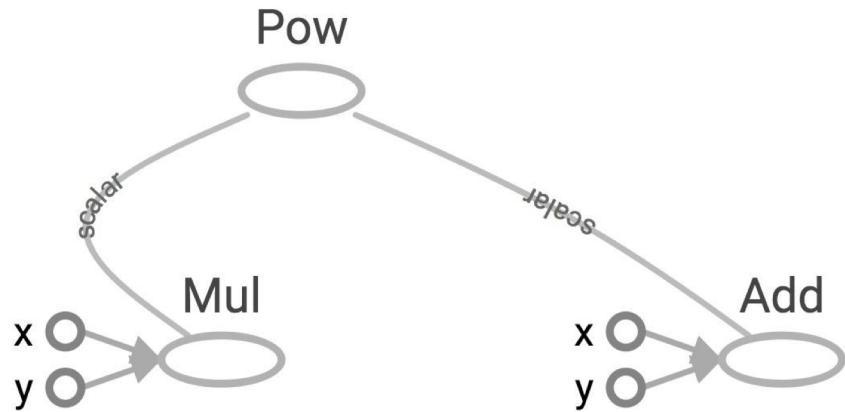
- Crear una sesión, asignarla a la variable sess de manera que podemos llamarla más tarde
- Dentro de la sesión, evaluar el grafo para así determinar el valor de a (resultado)

```
[?] import tensorflow as tf  
[?] a = tf.add(3, 5)  
[?] sess = tf.Session()  
[?] print sess.run(a)  
[?] sess.close()
```



Más grafos

- ☒ `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)`



Construir más de un grafo

- Se puede construir más de un grafo
- Pero realmente solo se necesita un grafo y múltiples subgrafos
- La sesión corre el grafo por defecto
- Múltiples grafos requieren múltiples sesiones, y cada una tratará de usar todos los recursos disponibles por defecto

tf.Graph()

Crear grafo

- ❑ `g = tf.Graph()`

Definirlo como grafo por defecto

- ❑ `with g.as_default():
 x = tf.add(3, 5)`

- ❑ `sess = tf.Session(graph=g)`

- ❑ `with tf.Session() as sess:
 sess.run(x)`

tf.Graph()

- ② g1 = tf.get_default_graph()
- ② g2 = tf.Graph()

```
# add ops to the default graph
```

- ② with g1.as_default():
 a = tf.Constant(3)

Funciona, pero no es
suficientemente bueno ¡pues hay
más de un grafo!

```
# add ops to the user created graph
```

- ② with g2.as_default():
 b = tf.Constant(5)