Framework para Aprendizaje Profundo

TensorFlow

Francisco Cervantes

Septiembre, 2019

Frameworks de aprendizaje profundo

- TensorFlow
- Keras
- Theano
- Torch
- Caffe
- mxnet
- CNTK
- DL4J
- Lasagne
- PaddlePaddle

Sugerencias para la elección de un un framework:

a. Fácil de programar (desarrollo y despliegue)
b. Velocidad de ejecución
c. Open source
d. Basado en la aplicación y restricciones del proyecto.

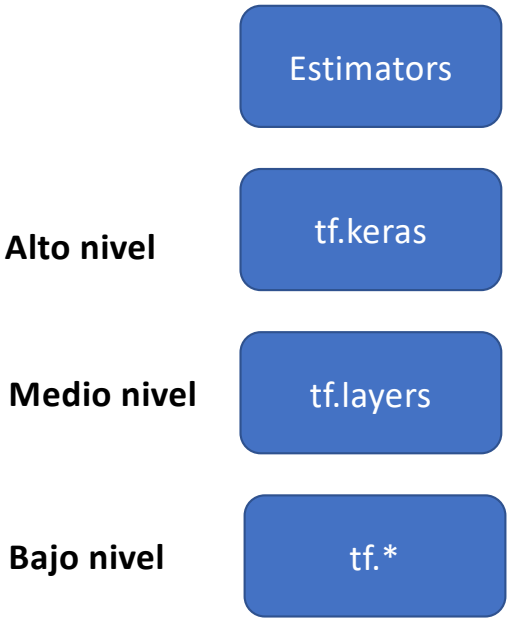
TensorFlow

TensorFlow es un framework Open Source (finales de 2015) y es uno de los más populares.

Flexibilidad de TensorFlow:

- Permite escribir código con APIs de alto nivel y bajo nivel.
- Existe soporte para programar en: Python, JavaScript, Swift, Go, etc.
- Se puede entrenar en CPU, GPU o TPU y desplegar en dispositivos móviles (android, iOS), Web, Raspberry Pi, etc.

APIs para construir modelos en TensorFlow



Estimators encapsula las siguientes acciones:

- Entrenamiento
- Evaluación
- Predicción
- Exportar a servicio

https://www.tensorflow.org/tutorials/estimators/linear

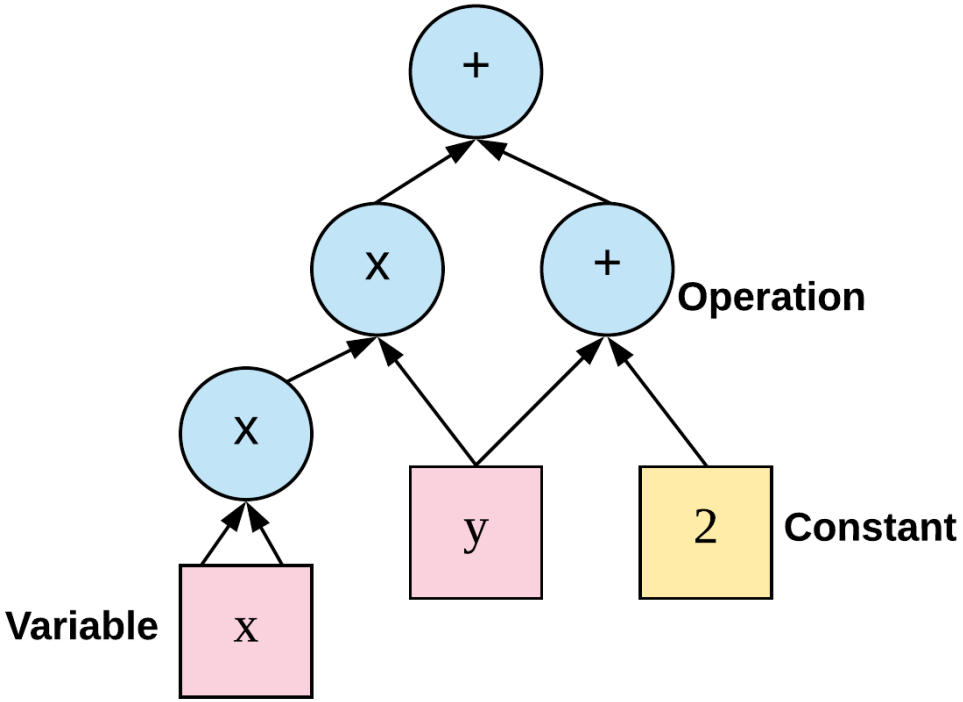
https://www.tensorflow.org/guide/keras

https://www.tensorflow.org/api_docs/python/tf/layers

https://www.tensorflow.org/api_docs/python/tf

Nota: tf.layers quedará fuera con TensorFlow 2.0. Se recomienda utilizar tf.Keras en lugar de tf.layers

Grafos de cómputo



Codificación y ejecución de programas con TensorFlow

1. Crear Tensors (variables)
   (no se ejecutan/evaluan)

2. Definir operaciones entre Tensors

3. Inicializar los Tensors

4. Crear una Session

5. Ejecutar la Session
   (en este momento se ejecutan las operaciones)

Tensores

El elemento central de los datos en TensorFlow es el **tensor.**

Un tensor consiste en un conjunto de valores primitivos conformados en un arreglo de cualquier número de dimensiones.

El **tensor rank** es el número de dimensiones, mientras que sus dimensiones es una tupla de enteros indicando la longitud de cada dimensión.

Veamos algunos ejemplos:

```
3                                  # tensor, rank 0; un escalar de dimensiones [],
[1, 2, 3]                          # tensor, rank 1; un vector con dimensiones [3]
[[1, 2, 3], [4, 5, 6]]             # tensor, rank 2; una matriz con dimensiones  [2, 3]
[[[1, 2, 3]], [[7, 8, 9]]]         # tensor, rank 3; con dimensiones [2, 1, 3]
```


Nota: TensorFlow utiliza **arreglos numpy** para representar valores de tensores.

Tipos de datos

Data type	Python type	Description
`DT_FLOAT`	`tf.float32`	32 bits floating point.
`DT_DOUBLE`	`tf.float64`	64 bits floating point.
`DT_INT8`	`tf.int8`	8 bits signed integer.
`DT_INT16`	`tf.int16`	16 bits signed integer.
`DT_INT32`	`tf.int32`	32 bits signed integer.
`DT_INT64`	`tf.int64`	64 bits signed integer.
`DT_UINT8`	`tf.uint8`	8 bits unsigned integer.
`DT_UINT16`	`tf.uint16`	16 bits unsigned integer.
`DT_STRING`	`tf.string`	Variable length byte arrays. Each element of a Tensor is a byte array.
`DT_BOOL`	`tf.bool`	Boolean.
`DT_COMPLEX64`	`tf.complex64`	Complex number made of two 32 bits floating points: real and imaginary parts.
`DT_COMPLEX128`	`tf.complex128`	Complex number made of two 64 bits floating points: real and imaginary parts.

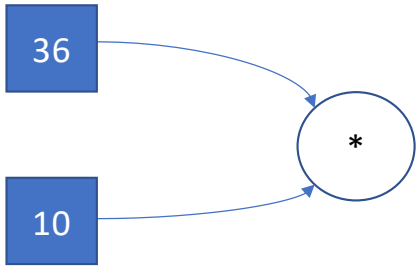
Definición de constantes

```
a = tf.constant(36, name = ‘a')

b = tf.constant(10, name=‘b’)

c = tf.multiply(a, b)


session = tf.Session()
print(f"c = {session.run(c)}")
```



Definición de variables

- El constructor tf.Variable() requiere un valor inicial para la variable. Puede ser un Tensor de cualquier tipo o forma.
- El valor inicial define el tipo y la forma de la variable.
- Después de la construcción el tipo y forma de la variable son fijos.
- Si se quiere modificar la forma de una variable, se debe utilizar una **Op assign** con **validate_shape=False**. (lo veremos en otro momento)
- Al igual que los Tensors, las variables pueden ser utilizadas como entradas para las operaciones del Grafo.

Definición de variables

```
a = tf.constant(36, name = ‘a’)
b = tf.constant(39, name = ‘b’)

r = tf.Variable(a-b, name=‘r')

init = tf.global_variables_initializer()

with tf.Session() as session:
    session.run(init)
    print(f”r = {session.run(r)}")
```

https://www.tensorflow.org/api_docs/python/tf/Variable

https://www.tensorflow.org/api_docs/python/tf/math

Uso de **Session**

```
a = tf.constant(36, name = ‘a’)
b = tf.constant(39, name = ‘b’)

r = tf.Variable(a-b, name=‘r')

init = tf.global_variables_initializer()

with tf.Session() as session:
    session.run(init)
    print(f”r = {session.run(r)}")
```

```
a = tf.constant(36, name = ‘a')

b = tf.constant(10, name=‘b’)

c = tf.multiply(a, b)


session = tf.Session()
print(f"c = {session.run(c)}")
```

https://www.tensorflow.org/api_docs/python/tf/Session

One Hot encodings