DEEP LEARNING BRASIL
SUMMER SCHOOL

Hands on - Deep Learning com Google TensorFlow and Keras

Sandro Silva Moreira

AI Engineer at DeepCardio Project

UniRV - Universidade de Rio Verde

https://www.tensorflow.org

TensorFlow

- Biblioteca de código aberto para aprendizado de máquina
- Segunda geração do sistema projetado pelo Google Brain. A versão 1.0.0 foi lançada em fev/2015, atualmente a versão mais recente é a 1.5
- Embora a implementação de referência seja executada em dispositivos individuais, pode também ser executado em múltiplas CPUs e GPUs (com extensões opcionais CUDA para GPGPU).

TensorFlow

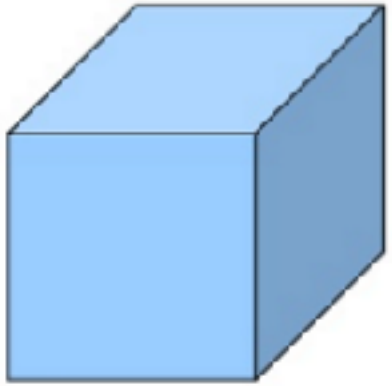
- Fornece uma API em Python, C++, Haskell, Java, Go, e Rust

- Cálculos no TensorFlow são expressos como grafos de fluxo de dados mantendo um estado.

- O nome TensorFlow deriva das operações que tais redes neurais realizam em arranjos de dados multidimensionais.

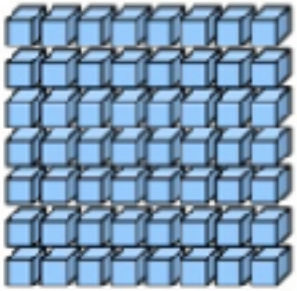
1d-tensor

2d-tensor

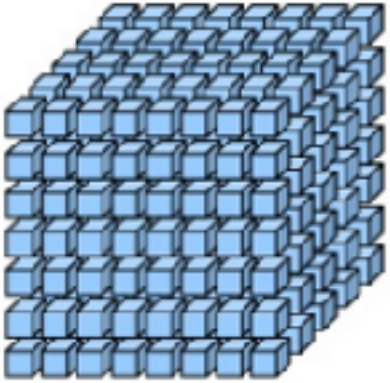


3d-tensor

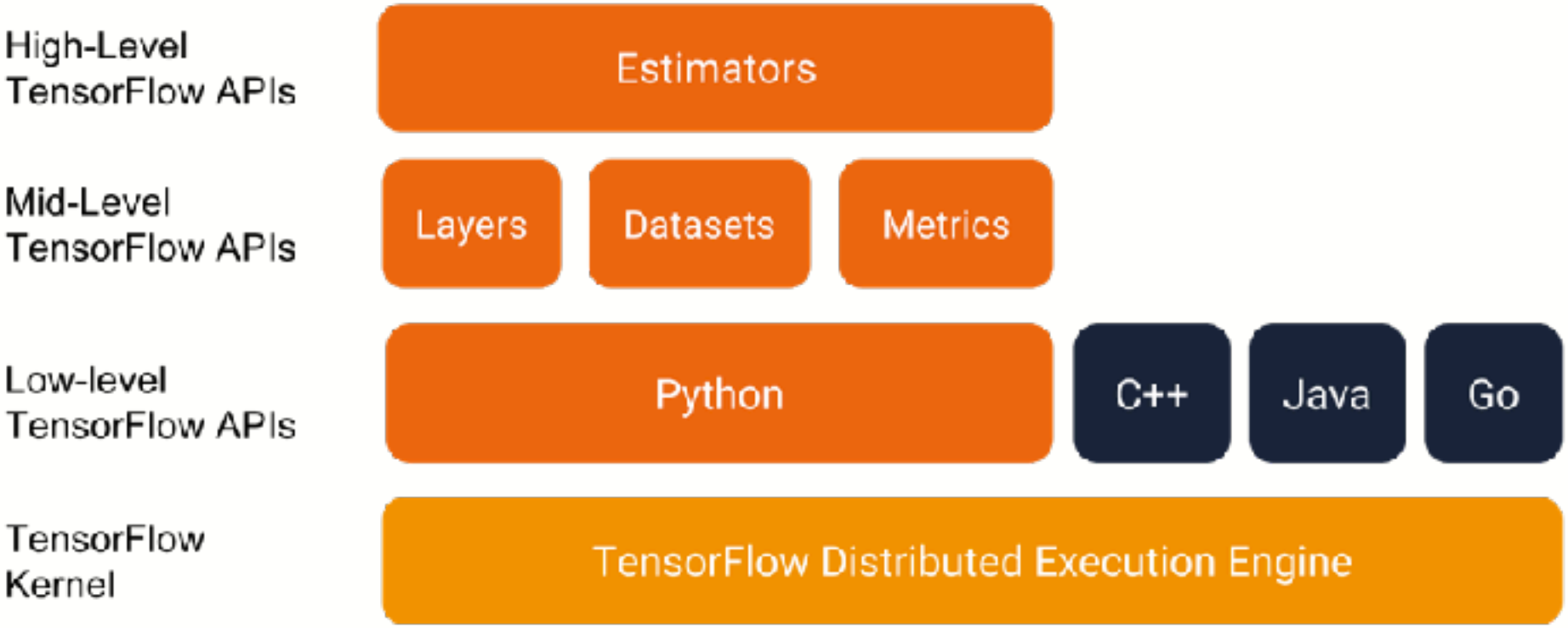
4d-tensor



5d-tensor



6d-tensor



- Estimators, which represent a complete model. The Estimator API provides methods to train the model, to judge the model's accuracy, and to generate predictions.
- Datasets, which build a data input pipeline. The Dataset API has methods to load and manipulate data, and feed it into your model. The Datasets API meshes well with the Estimators API.

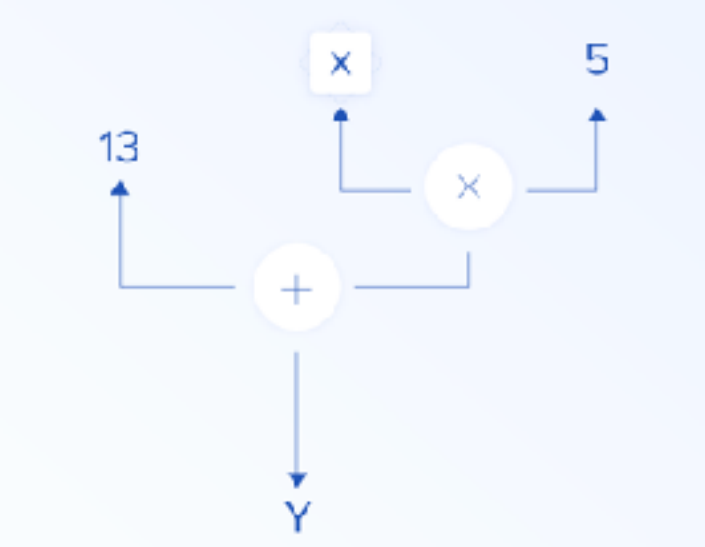
Uso Básico

```
1 import tensorflow as tf
2 #Criamos constantes
3 x_1 = tf.constant(2.0,dtype=tf.float32)
4 #A função tf.constant cria nossas constantes
5 x_2 = tf.constant(5.0,dtype=tf.float32)
6 #Aqui multiplicamos os dois resultados
7 mult = x_1*x_2
8 #Aqui somamos as duas constantes
9 soma = x_1+x_2
10 sess = tf.Session()
11 # Toda vez que quisermos executar usamos o método
12 run,Aqui executamos a multiplicação
13 print(sess.run(mult))
14 #Aqui executamos a soma
15 print(sess.run(soma))
```

Começando com TensorFlow

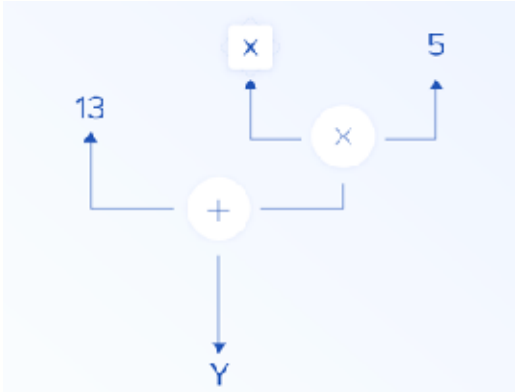
- Em TensorFlow, a computação é descrita usando gráficos de fluxo de dados.
- Cada nó do gráfico representa uma instância de uma operação matemática (como adição, divisão ou multiplicação) e cada borda é um conjunto de dados multidimensionais (tensor) no qual as operações são realizadas.

Função Linear Simples




```
#Python Simples

x = -2.0
y = 5*x + 13
print(y)
```

Função Linear Simples




```
import tensorflow as tf

x = tf.constant(-2.0, name="x", dtype=tf.float32)
a = tf.constant(5.0, name="a", dtype=tf.float32)
b = tf.constant(13.0, name="b", dtype=tf.float32)

y = tf.Variable(tf.add(tf.multiply(a, x), b))

init = tf.global_variables_initializer()

with tf.Session() as session:
    session.run(init)
    print session.run(y)
```

Exemplo 2

```
import tensorflow as tf

x = tf.placeholder(tf.float32, name="x")
y = tf.placeholder(tf.float32, name="y")

z = tf.multiply(x, y, name="z")

with tf.Session() as session:
    print session.run(z, feed_dict={x: 2.1, y: 3.0})
```

Visualizando grafos com TensorBoard

- TensorBoard é uma ferramenta de visualização para análise de gráficos de fluxo de dados. Isso pode ser útil para obter uma melhor compreensão dos modelos de aprendizagem de máquinas.

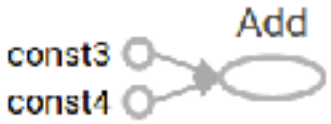
- Com o TensorBoard, você pode obter informações sobre diferentes tipos de estatísticas sobre os parâmetros e detalhes sobre as partes do gráfico computacional em geral. Não é incomum que uma rede neural profunda tenha um grande número de nós. O TensorBoard permite aos desenvolvedores obter informações sobre cada nó e como a computação é executada durante o tempo de execução TensorFlow.

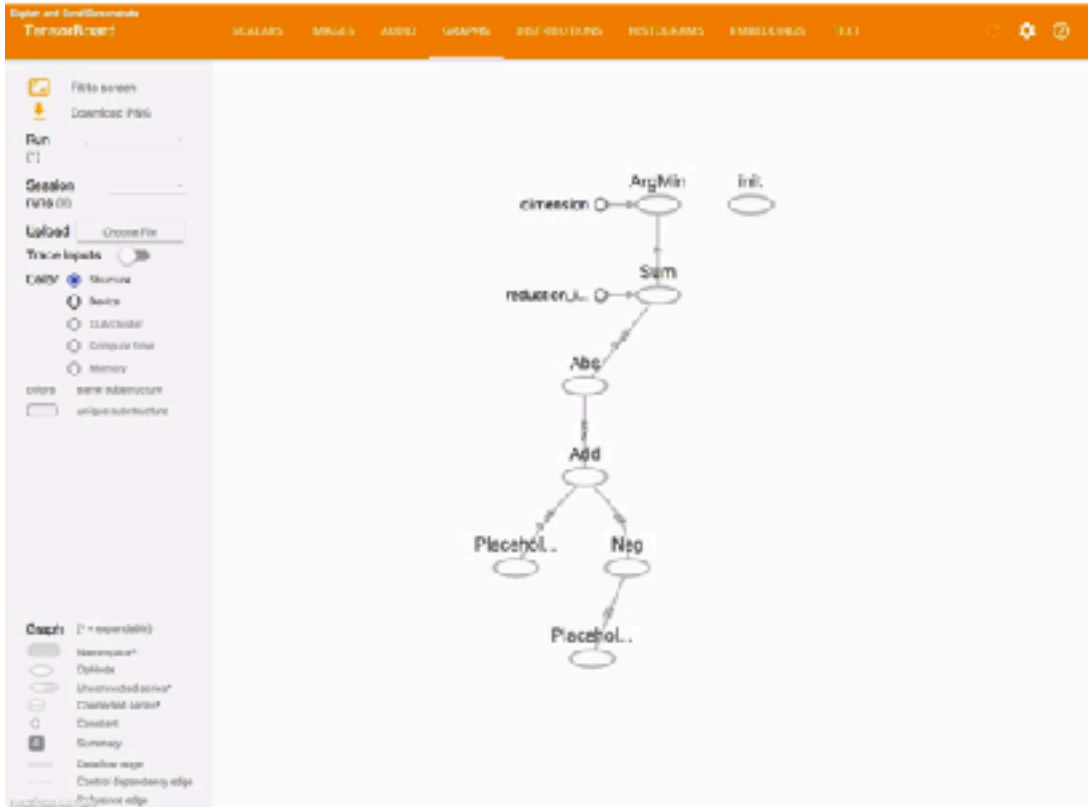
Visualizando grafos com TensorBoard

```
import tensorflow as tf

x = tf.constant(-2.0, name="x", dtype=tf.float32)
a = tf.constant(5.0, name="a", dtype=tf.float32)
b = tf.constant(13.0, name="b", dtype=tf.float32)

y = tf.Variable(tf.add(tf.multiply(a, x), b))

init = tf.global_variables_initializer()

with tf.Session() as session:
    merged = tf.summary.merge_all()
    writer = tf.summary.FileWriter("logs", session.graph)

    session.run(init)
    print session.run(y)
```

TensorBoard

```
tensorboard --logdir logs/
```

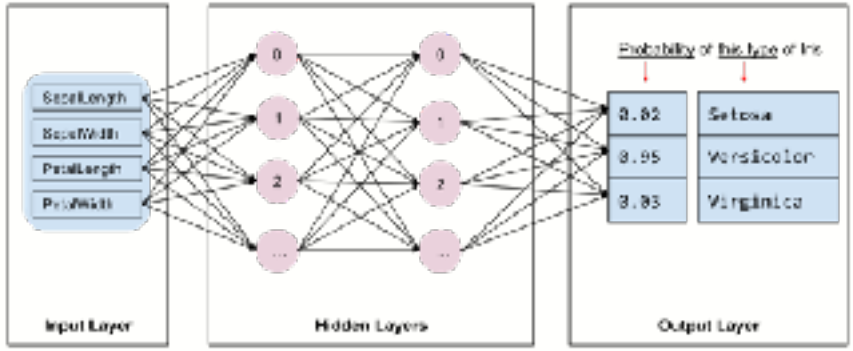




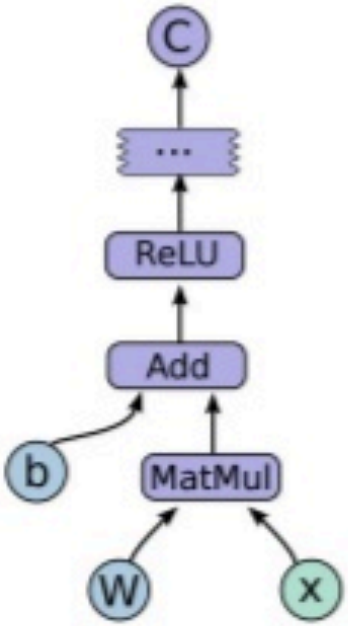
Demo - Notebook (classificador_tensorflow)



sepal length	sepal width	petal length	petal width	species (label)
5.1	3.3	1.7	0.5	0 (Setosa)
5.0	2.3	3.3	1.0	1 (versicolor)
6.4	2.8	5.6	2.2	2 (virginica)



Arquiteturas Deep Learning com TF




```
# define the network
import tensorflow as tf
x = tf.placeholder(tf.float32, [None, 784])
W = tf.Variable(tf.zeros([784, 10]))
b = tf.Variable(tf.zeros([10]))
y = tf.nn.softmax(tf.matmul(x, W) + b)

# define a training step
y_ = tf.placeholder(tf.float32, [None, 10])
xent = -tf.reduce_sum(y_*tf.log(y))
step = tf.train.GradientDescentOptimizer(0.01).minimize
(xent)
```

https://keras.io/layers/core/

Keras

- Biblioteca para rede neural de alto-nível escrita em Python e roda como frontend em TensorFlow ou Theano.

- Foi desenvolvida para facilitar experimentações rápidas, isto é, sem que você tenha que dominar cada um dos backgrounds, de maneira rápida e eficiente.

Keras

- Prototipagem rápida e fácil (total modularidade, minimalismo e extensibilidade)
- Suporte a redes convolucionais e recorrentes, incluindo combinação de ambas
- Suporte a esquemas de conectividade arbitrária (incluindo treino de N para N)
- Roda na CPU ou GPU

Keras - Exemplo

```
import numpy as np
from keras.models import Sequential
from keras.layers import Dense

data = np.random.random((1000,100))
labels = np.random.randint(2,size=(1000,1))
model = Sequential()
model.add(Dense(32,
          activation='relu',
          input_dim=100))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer='rmsprop',
              loss='binary_crossentropy',
              metrics=['accuracy'])
             model.fit(data,labels,epochs=10,batch_size=32)
predictions = model.predict(data)
```

Keras - Exemplo

```
from keras.layers import Dense

model.add(Dense(12,
            input_dim=8,
            kernel_initializer='uniform',
            activation='relu'))

model.add(Dense(8,
            kernel_initializer='uniform',
            activation='relu'))

model.add(Dense(1,kernel_initializer='uniform',activation='sigmoid'))
```

Demonstração - neuralnetworkKeras

- Multilayer Perceptron em Keras
- Dataset: Pima Indians onset of diabetes - This is a standard machine learning dataset from the UCI Machine Learning repository. It describes patient medical record data for Pima Indians and whether they had an onset of diabetes within five years.

1. Load Data.
2. Define Model.
3. Compile Model.
4. Fit Model.
5. Evaluate Model.

Keras - Rede Convolucional

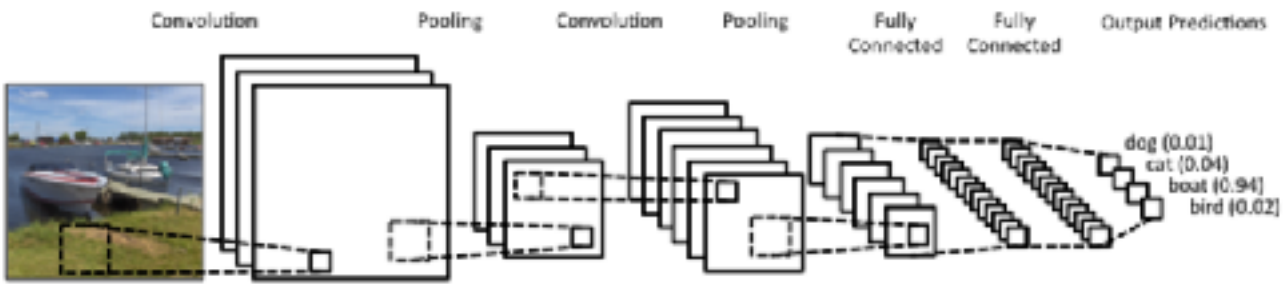
```
from keras.layers import Activation,Conv2D,MaxPooling2D,Flatten
model2.add(Conv2D(32,(3,3),padding='same',input_shape=x_train.shape[1:]))
model2.add(Activation('relu'))
model2.add(Conv2D(32,(3,3)))
model2.add(Activation('relu'))
model2.add(MaxPooling2D(pool_size=(2,2)))
model2.add(Dropout(0.25))
model2.add(Conv2D(64,(3,3), padding='same'))
model2.add(Activation('relu'))
model2.add(Conv2D(64,(3, 3)))
model2.add(Activation('relu'))
model2.add(MaxPooling2D(pool_size=(2,2)))
model2.add(Dropout(0.25))
model2.add(Flatten())
model2.add(Dense(512))
model2.add(Activation('relu'))
model2.add(Dropout(0.5))
model2.add(Dense(num_classes))
model2.add(Activation('softmax'))
```

Keras - Treinando a Rede

```
model.fit(
    x_train,
    y_train,
    batch_size=32,
    epochs=15,
    verbose=1,
    validation_data=(x_test,y_test)
    )
```

Exemplo Prático - TF

Rede Convolucional Inception pré-treinada será retreinada para reconhecer personagens de “Os Simpsons”



Dicas

- TensorFlow Dev Summit 2018 - March 30, 2018 at the Computer History Museum in Mountain View, CA.