U Positivo - UTFPR - PUCPr – 11/02/2019 - 12:36:04.8
Prof Dr P Kantek (pkantek@gmail.com)
Tensor Flow VIVO851a, V: 1.05 1
 _____ / _____ / _____

Tensor Flow

É um produto freeware (licença Apache) desenvolvido pela Google para aprendizado de máquina. Substituiu seu antecessor DistBelief que era proprietário. Além de ser um belo projeto de pesquisa ele é usado em produção em muitos produtos google (tradução, classificação de fotos, busca, reconhecimento da fala humana, etc). Sua primeira versão foi liberada em 9/11/15 e seu desenvolvimento começou em 2011.

Está em `www.tensorflow.org`. Embora tenha APIs para uso em C++, Java, Go seu uso mais completo e documentado se dá em Python, já que a ferramenta está desenvolvida neste ambiente. Uma ótima oportunidade para apreender e começar a usar esta linguagem, não é ?

Pode-se visualizar o vídeo em `https://www.youtube.com/watch?v=oZikw5k_2FM`. Se quiser, peça a tradução simultânea para o português. Verá o TS em ação.

Python

Uma das mais recentes liguagens de programação, já que surgiu em 1989. Tem pai: é o programador holandês Guido Van Rossum. Em homenagem ao papel representado por ele, ganhou da comunidade Python o título de BDFL ("Benevolent Dictator for Life"). Por ser nova, a linguagem ultrapassou o carma das suas antecessoras (*precisamos economizar recursos*) e não teve vergonha de herdar ("surrupiar") todas as boas idéias já em curso no final do século XX. Nasceu uma linguagem que permite fazer qualquer coisa de uma maneira muito fácil. Parece um texto em pseudo-código fácil de ler e escrever. Se você olhar o estudo da TIOBE `https://www.tiobe.com/tiobe-index/` verá o bonito papel desempenhado pelo Python.

Linguagem	17	12	07	02	97	92
Java	1	1	1	1	12	-
C	2	2	2	2	1	1
C++	3	3	3	3	2	2
C#	4	4	7	17	-	-
Python	5	7	6	11	27	-
VB.NET	6	19	-	-	-	-
PHP	7	6	4	5	-	-
JavaScript	8	9	8	8	19	-
COBOL	25	28	17	9	3	10
Lisp	32	12	14	12	9	5
Prolog	33	32	26	15	20	12
Pascal	106	15	19	97	8	3

Consulta em 07/2017

Ele é multiplataforma e livre, então pode ser instalado em Windows, Mac, Linux, Android, IOS, etc etc. Para instalar no ambiente windows, vá em `https://www.python.org/downloads/` e baixe a versão 3 mais recente. A versão 2 ainda suportada, não deve ser a escolhida: já está marcada a festa de fim de vida desta versão. A 3 é um pouco diferente e muito melhor.

O mundo Python está organizado em volta do PYPI (Python Package Index) que vem a ser um repositório de pacotes Python. Em meados de 2017 ele tinha mais de 110.000 pacotes, todos livres. Um dos pacotes que são necessário para o Tensor Flow é o pacote NUMPY (Numerical Computation for Python). depois de instalar o Python, se ele ainda não tiver instalado o NUMPY basta ir no diretório onde o Python está instalado e com direitos de adiminstrador fazer `pip install NUMPY` lembrando que para isto funcionar você deve estar conectado à rede.

Vale lembrar a existência de um ambiente Python que pode ser executado avulso (a partir de um pendrive, por exemplo), o que é ótimo quando você não tem os direitos de administrador na máquina que está usando. Trata-se do chamado Python portátil e ele pode ser buscado em `https://winpython.github.io/`. Este sujeito já traz o NUMPY instalado.

Tensor Flow

Este produto só roda em máquinas 64 bits (lembrem-se, ele não economiza recursos) e estando nesse ambiente, a instalação é simples, basta fazer `pip install tensorflow`.

Depois de instalar, para verificar se deu tudo certo faça:

```
>>> import tensorflow as tf
>>> hello = tf.constant('Oi, TensorFlow!')
>>> sess = tf.Session()
>>> print(sess.run(hello))
```

Deve aparecer a mensagem `Oi, TensorFlow!` indicando que a instalação foi bem sucedida.

Tensor

O ponto central em TF é o tensor. Trata-se de um array de valores primitivos configurados em uma matriz (Python). O ranking de um tensor é o número de dimensões da matriz que o contém. Eis alguns exemplos:

```
3 # tensor de rank 0; escalar com forma []
[1. ,2., 3.] # tensor de rank 1;vetor forma [3]
[[1., 2., 3.], [4., 5., 6.]] # tensor rank 2;
#                       matriz com forma [2, 3]
[[[1., 2., 3.]], [[7., 8., 9.]]]
# tensor rank 3; tensor com forma [2, 1, 3]
```

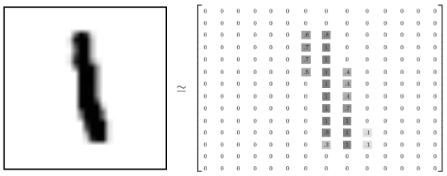
Rede

Uma rede neural é uma série de operações TensorFlow dispostas em um grafo de nodos. Cada nodo tem zero ou mais tensores de entrada e produz um tensor de saída.

Um exemplo completo está a seguir. Ele é um interpretador de dígitos escritos à mão, a partir dos dados do NIST.

É um conjunto de 70.000 imagens de dígitos manuscritos capturadas como imagens de 28×28 pixels ($28 \times 28 = 784$) associados a um vetor [0..9] que indica qual o número certo. Os dados estão divididos em 55.000 (treinamento) 10.000 (teste) e 5.000 validação. Lembre-se que quem aprova não pode ser o que treinou.

Veja um exemplo de um número 1



Se você rodar o programa abaixo terá uma acurácia de 92% aproximadamente. Melhorando os operadores da rede neural chega-se facilmente a 97% e os melhores modelos chegam a 99.7 %.

Eis algumas imagens difíceis do conjunto

Uma explicação completa deste exemplo está em

```
\verb+https://www.tensorflow.org/get_started/mnist/beginners+
O módulo abaixo é mnist_softmax.py...

from __future__ import absolute_import
from __future__ import division
from __future__ import print_function

import argparse
import sys

from tensorflow.examples.tutorials.mnist import input_data

import tensorflow as tf #importa o tensorflow

FLAGS = None


def main(_):
  # Import data
  mnist = input_data.read_data_sets(FLAGS.data_dir, one_hot=True)

  # Create the model
  x = tf.placeholder(tf.float32, [None, 784])  # dados de entrada
  W = tf.Variable(tf.zeros([784, 10]))         # a RN
  b = tf.Variable(tf.zeros([10]))              # os bias
  y = tf.matmul(x, W) + b                      # o resultado da RN

  # Define loss and optimizer
  y_ = tf.placeholder(tf.float32, [None, 10])
        # lugar onde vem a resposta certa

  cross_entropy = tf.reduce_mean(
      tf.nn.softmax_cross_entropy_with_logits(labels=y_, logits=y))
    # quero minimizar a diferença entre y (calculado) e y_ (certo)
  train_step =
  tf.train.GradientDescentOptimizer(0.5).minimize(cross_entropy)

  sess = tf.InteractiveSession()   # cria a sessao
  tf.global_variables_initializer().run()
    # inicializa variáveis
  # Train
  for _ in range(1000):
   # treinamento 1000 sessoes de 100 aleatorios
    batch_xs, batch_ys = mnist.train.next_batch(100)
    # coloca aqui os dados
    sess.run(train_step, feed_dict={x: batch_xs, y_: batch_ys})
    #roda a RN

  # Test trained model
  correct_prediction = tf.equal(tf.argmax(y, 1), tf.argmax(y_, 1))
                  #y é igual a y_?
  accuracy =
   tf.reduce_mean(tf.cast(correct_prediction, tf.float32))
     #acurácia
  print(sess.run(accuracy, feed_dict={x: mnist.test.images,
                                      y_: mnist.test.labels}))
     #imprime acurácia a partir das imagens de teste...


if __name__ == '__main__':
    # descobre se isto está sendo chamado ou importado
  parser = argparse.ArgumentParser()
  parser.add_argument('--data_dir',
    type=str, default='/tmp/tensorflow/mnist/input_data',
              help='Directory for storing input data')
  FLAGS, unparsed = parser.parse_known_args()
  tf.app.run(main=main, argv=[sys.argv[0]] + unparsed)
```

Se você rodar o programa acima terá uma rede neural funcional completa e obterá a acurácia de uma execução (tipicamente 92%).

☞ Para você fazer

Sua tarefa aqui é razoavelmente simples:

1. Instale (ou descubra o uso de) Python 3 em uma máquina 64 bits
2. Instale o pacote tensorflow (`pip install tensorflow`)
3. Estude os tutoriais:
   - Getting started
   - ... with Tensor Flow
   - MNIST for ML beginners
   - Deep MNIST for Experts

   Se quiser, peça para o browser traduzir para o português, o resultado é bastante razoável (feito pelo TF...)
4. Digite ou copie o programa `mnist_softmax.py`
5. Rode-o.

Você terá uma Rede Neural satisfatória e contendo o estado da arte em Machine Learning.