Nghia Ho

TECHNICAL

USING TENSORFLOW/KERAS WITH CSV FILES

JULY 25, 2016 | NGHIAHO12 | 6 COMMENTS

I've recently started learning TensorFlow in the hope of speeding up my existing machine learning tasks by taking advantage of the GPU. At first glance the documentation looks decent but the more I read the more I found myself scratching my head on how to do even the most basic task. All I wanted to do was load in a CSV file and run it through a simple neural network. When I was downloading the necessary CUDA libraries from NVIDIA I noticed they listed a handful of machine learning framework that were supported. One of them was Keras, which happens to build on top of TensorFlow. After some hard battles with installing CUDA, TensorFlow and Keras on my Ubuntu 16.04 box and a few hours of Stackoverflow reading I finally got it working with the following python code.

```
from keras.models import Sequential
from keras.layers import Dense, Activation

import numpy as np
import os.path

if not os.path.isfile("data/pos.npy"):
    pos = np.loadtxt('data/pos.csv', delimiter=',', dtype=np.float32)
    np.save('data/pos.npy', pos);
else:
    pos = np.load('data/pos.npy')

if not os.path.isfile("data/neg.npy"):
    neg = np.loadtxt('data/neg.csv', delimiter=',', dtype=np.float32)
    np.save('data/neg.npy', neg);
else:
    neg = np.load('data/neg.npy')

pos_labels = np.ones((pos.shape[0], 1), dtype=int);
neg_labels = np.zeros((neg.shape[0], 1), dtype=int);

print "positive samples: ", pos.shape[0]
```

```
print "negative samples: ", neg.shape[0]

HIDDEN_LAYERS = 4

model = Sequential()

model.add(Dense(output_dim=HIDDEN_LAYERS, input_dim=pos.shape[1]))
model.add(Activation("relu"))
model.add(Dense(output_dim=1))
model.add(Activation("sigmoid"))

model.compile(optimizer='rmsprop', loss='binary_crossentropy', metrics=
['accuracy'])
model.fit(np.vstack((pos, neg)), np.vstack((pos_labels, neg_labels)),
nb_epoch=10, batch_size=128)

# true positive rate
tp = np.sum(model.predict_classes(pos))
tp_rate = float(tp)/pos.shape[0]

# false positive rate
fp = np.sum(model.predict_classes(neg))
fp_rate = float(fp)/neg.shape[0]

print ""
print ""

print "tp rate: ", tp_rate
print "fp rate: ", fp_rate
```

I happened to have my positive and negative samples in separate CSV files. The CSV files are converted to native Numpy binary for subsequent loading because it is much faster than parsing CSV. There's probably some memory wastage going on with the np.vstack that could be improved on.

6 THOUGHTS ON "USING TENSORFLOW/KERAS WITH CSV FILES"

Pingback: d243: Using Tensorflow/Keras with CSV files [study material] | AI:Mechanic

---

Fthi

FEBRUARY 14, 2017 AT 8:01 AM

Can you attach the dataset as well.

★ nghiaho12

FEBRUARY 18, 2017 AT 1:29 PM

No, because the dataset was for work.

Mike

APRIL 12, 2017 AT 11:22 PM

Thanks for the great post! These deep-learning frameworks are pretty impressive, but most seem to omit the obvious when it comes to getting started with real-world data of your own.

Rusty

MAY 20, 2017 AT 2:11 AM

Seems like a dataset for sentiment analysis. What was the format of your dataset?

★ nghiaho12

MAY 20, 2017 AT 2:17 AM

CSV with an image per row in ascii