

Canthink Script Generation for HTTP Request

Copyright (c) 2017, Yida Wang, Canthink, All rights reserved.

Use of this source code is governed by a BSD-style license that can be found in the LICENSE file.

Server Setting

Ubuntu

Install screen and ssh for remote control and background monitoring.

```
sudo apt-get install htop screen ssh
```

More modern versions of Ubuntu you could just install `pip` via, it works on my Ubuntu 16.04:

```
sudo apt-get install python-dev python-virtualenv python-pip  
python3-pip
```

CentOS

Coming soon.

MacOS

Coming soon.

TensorFlow Setting

Information

You can find installing information for Linux system on official website of [TensorFlow](#).

Nvidia Support(optional)

NVIDIA requirements to run TensorFlow with GPU support

```
sudo apt-get install libcupti-dev
```

Install the nvidia drivers for Ubuntu (for 14.04 and newer), firstly add the graphics-drivers ppa:

```
sudo add-apt-repository ppa:graphics-drivers/ppa sudo apt-get update
```

Then install the recommended driver

```
sudo ubuntu-drivers autoinstall
```

Then restart your system.

Install CUDA 8, you can also find information on [Nvidia-CUDA](#) website.

Downloading script:

```
wget
```

```
http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64/cuda-repo-ubuntu1604_8.0.61-1_amd64.deb
```

Installation Instructions:

```
sudo dpkg -i cuda-repo-ubuntu1604_8.0.61-1_amd64.deb
```

```
sudo apt-get update
```

```
sudo apt-get install cuda
```

Install TensorFlow

```
pip install --upgrade tensorflow # for Python 2.7
```

```
pip3 install --upgrade tensorflow # for Python 3.n
```

```
pip install --upgrade tensorflow-gpu # for Python 2.7 and GPU
```

```
pip3 install --upgrade tensorflow-gpu # for Python 3.n and GPU
```

Install Server and Client Monitor

Installing Jupyter Notebook

As an existing Python user, you may wish to install Jupyter using Python's package manager, pip, instead of Anaconda.

First, ensure that you have the latest pip; older versions may have trouble with some dependencies:

```
pip3 install --upgrade pip
```

Then install the Jupyter Notebook using:

```
pip3 install jupyter
```

Remote Access to IPython Notebooks via SSH

Here I took a reference on blog in [Coderwall](#)

Scenario: On your local computer, you want to open and manipulate an IPython notebook running on a remote computer. We will do this by opening an SSH tunnel. This tunnel will forward the port used by the remotely running IPython instance to a port on the local machine, where it can be accessed in a browser just like a locally running IPython instance.

On the remote machine, start the IPython notebooks server:

```
remote_user@remote_host$ ipython notebook --no-browser --port=8889
```

Usually IPython opens a browser to display the available notebooks, but we do not need that so we use the option `--no-browser`. We also change the port to 8889, for no other reason than to show how this is done.

On the local machine, start an SSH tunnel:

```
local_user@local_host$ ssh -N -f -L localhost:8888:localhost:8889 remote_user@remote_host
```

The first option `-N` tells SSH that no remote commands will be executed, and is useful for port forwarding. The second option `-f` has the effect that SSH will go to background, so the local tunnel-enabling terminal remains usable. The last option `-L` lists the port forwarding configuration (remote port 8889 to local port 8888).

Now open your browser on the local machine and type in the address bar

```
localhost:8888
```

which displays your remotely running IPython notebook server. To close the SSH tunnel on the local machine, look for the process and kill it manually:

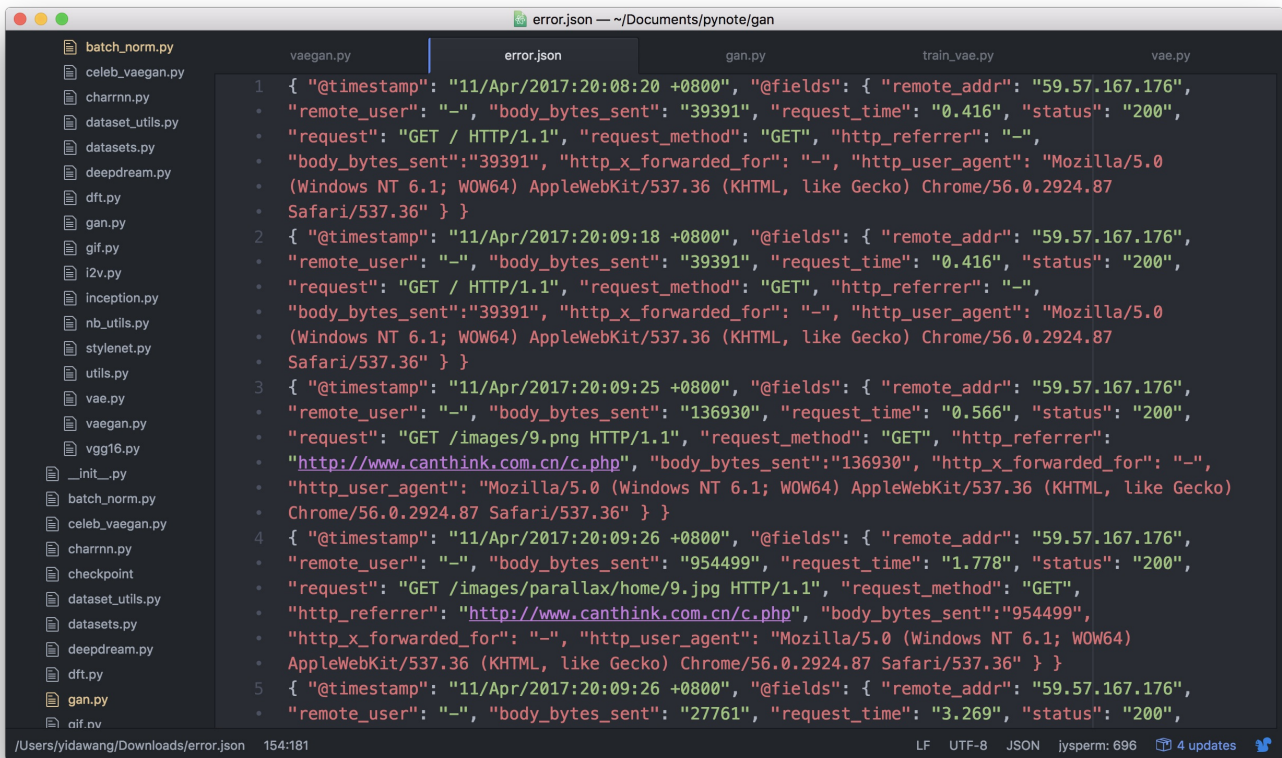
```
local_user@local_host$ ps aux | grep localhost:8889
```

Alternatively, you can start the tunnel without the `-f` option. The process will then remain in the foreground and can be killed with `ctrl-c`.

On the remote machine, kill the IPython server with `ctrl-c ctrl-c`.

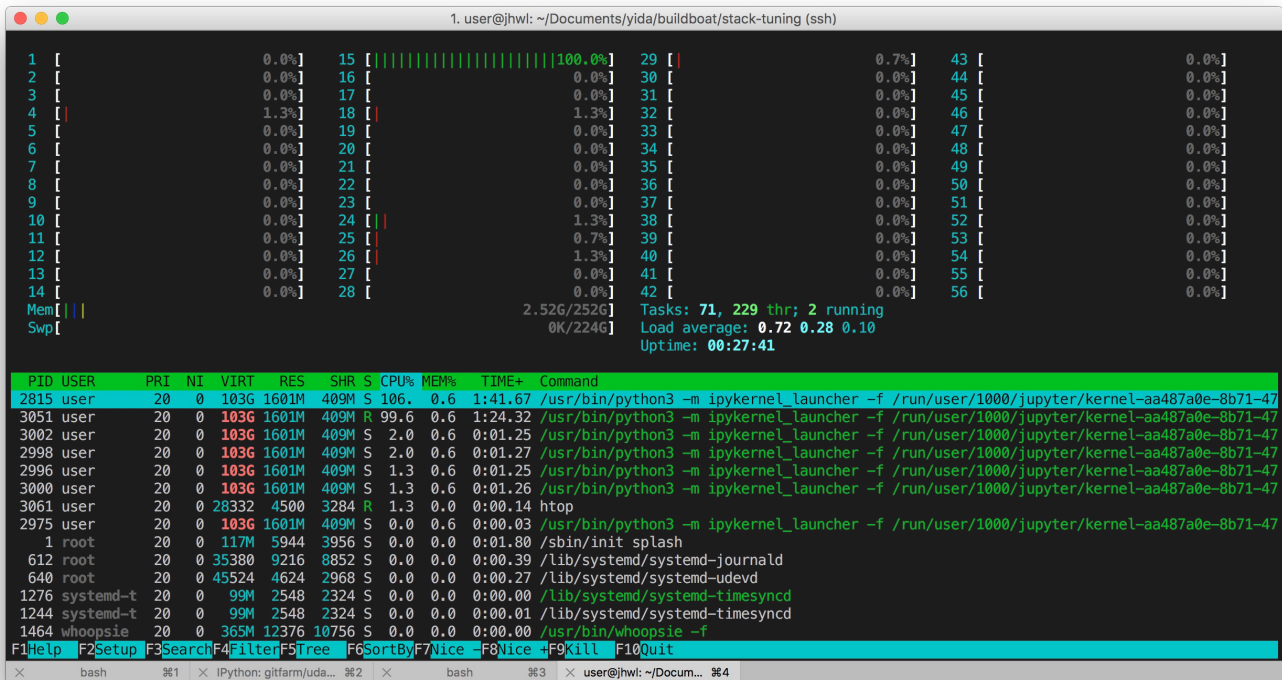
Platform Information

Training Data



Server Status

Server Status of CPU



Server Status of GPU

```
1. user@jhlw: ~/Documents/yida/buildboat/stack-tuning (ssh)
user@jhlw:~/Documents/yida/buildboat/stack-tuning$ ls script_generation/error.txt
script_generation/error.txt
user@jhlw:~/Documents/yida/buildboat/stack-tuning$ htop
user@jhlw:~/Documents/yida/buildboat/stack-tuning$ nvidia-smi
Fri Apr 14 00:32:52 2017

+-----+
| NVIDIA-SMI 375.51                  Driver Version: 375.51           |
+-----+-----+
| GPU  Name      Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|  Memory-Usage | GPU-Util  Compute M. |
+-----+-----+
| 0  GeForce GTX 108...    Off | 0000:03:00.0    On  |          35%      N/A |
| 31%   57C   P2    99W / 250W | 10690MiB / 11170MiB |              Default |
+-----+-----+
| 1  GeForce GTX 108...    Off | 0000:04:00.0   Off  |          0%      N/A |
| 23%   37C   P8    9W / 250W | 10622MiB / 11172MiB |              Default |
+-----+-----+
| 2  GeForce GTX 108...    Off | 0000:81:00.0   Off  |          0%      N/A |
| 24%   42C   P8    9W / 250W | 10622MiB / 11172MiB |              Default |
+-----+-----+
| 3  GeForce GTX 108...    Off | 0000:82:00.0   Off  |          0%      N/A |
| 23%   34C   P8    9W / 250W | 10622MiB / 11172MiB |              Default |
+-----+-----+

+-----+
| Processes:                         GPU Memory |
| GPU       PID  Type  Process name                               Usage |
+-----+-----+
| 0          1674  G   /usr/lib/xorg/Xorg                               60MiB |
| 0          2815  C   /usr/bin/python3                               10625MiB |
| 1          2815  C   /usr/bin/python3                               10619MiB |
| 2          2815  C   /usr/bin/python3                               10619MiB |
| 3          2815  C   /usr/bin/python3                               10619MiB |
+-----+-----+

user@jhlw:~/Documents/yida/buildboat/stack-tuning$
```

Client

Input Definition

localhost:8888/notebooks/script_generation.ipynb

Jupyter script_generation Last Checkpoint: 11 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Help Trusted Python 3

Script Generation

This project is developed by ideas of utilizing [Simpsons](#) TV scripts using RNNs. You'll be using part of the [Simpsons dataset](#) of scripts from 27 seasons.

Get the Data

In testing phase, you'll be using a subset of the original dataset. It consists of only the scenes in Moe's Tavern. This doesn't include other versions of the tavern, like "Moe's Cavern", "Flaming Moe's", "Uncle Moe's Family Feed-Bag", etc..

```
In [2]: """
DON'T MODIFY ANYTHING IN THIS CELL
"""
import helper

data_dir = './error.txt'
text = helper.load_data(data_dir)
# Ignore notice, since we don't use it for analysing the data
# text = text[81:]
```

Explore the Data

Play around with `view_sentence_range` to view different parts of the data.

5d6fbd03-c638-4c99-ad...

8c22f04b-dcfc-447b-bbd...

6e88c021-62c2-4084-9...

train.csv

Show All

GPU Information of Server

localhost:8888/notebooks/script_generation.ipynb

Jupyter script_generation Last Checkpoint: 11 hours ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Help Trusted Python 3

Check the Version of TensorFlow and Access to GPU

```
In [8]: """
DON'T MODIFY ANYTHING IN THIS CELL
"""
from distutils.version import LooseVersion
import warnings
import tensorflow as tf

# Check TensorFlow Version
assert LooseVersion(tf.__version__) >= LooseVersion('1.0'), 'Please use TensorFlow version 1.0 or newer'
print('TensorFlow Version: {}'.format(tf.__version__))

# Check for a GPU
if not tf.test.gpu_device_name():
    warnings.warn('No GPU found. Please use a GPU to train your neural network.')
else:
    print('Default GPU Device: {}'.format(tf.test.gpu_device_name()))

TensorFlow Version: 1.0.1
Default GPU Device: /gpu:0
```

Input

Implement the `get_inputs()` function to create TF Placeholders for the Neural Network. It should create the following placeholders:

- Input text placeholder named "input" using the [TF Placeholder](#) name parameter.

5d6fbd03-c638-4c99-ad... 8c22f04b-dcfc-447b-bbd... 6e88c021-62c2-4084-9... train.csv Show All

Training Progress

localhost:8888/notebooks/script_generation.ipynb

Jupyter script_generation Last Checkpoint: 11 hours ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Help Trusted Python 3

```
for epoch_i in range(num_epochs):
    state = sess.run(initial_state, {input_text: batches[0][0]})

    for batch_i, (x, y) in enumerate(batches):
        feed = {
            input_text: x,
            targets: y,
            initial_state: state,
            lr: learning_rate
        }
        train_loss, state, _ = sess.run([cost, final_state, train_op], feed)

        # Show every <show_every_n_batches> batches
        if (epoch_i * len(batches) + batch_i) % show_every_n_batches == 0:
            print('Epoch {:>3} Batch {:>4}/{} train_loss = {:.3f}'.format(
                epoch_i,
                batch_i,
                len(batches),
                train_loss))

    # Save Model
    saver = tf.train.Saver()
    saver.save(sess, save_dir)
    print('Model Trained and Saved')
```

Epoch	0	Batch	0/167	train_loss	= 7.913
Epoch	1	Batch	0/167	train_loss	= 3.236
Epoch	2	Batch	0/167	train_loss	= 3.242
Epoch	3	Batch	0/167	train_loss	= 2.527
Epoch	4	Batch	0/167	train_loss	= 2.225

5d6fbd03-c638-4c99-ad... 8c22f04b-dcfc-447b-bbd... 6e88c021-62c2-4084-9... train.csv Show All

Result