**Pols559 Text as Data**

**HOW TO CREATE AND ACCESS AN AMAZON INSTANCE - IN FOUR EASY PAGES**

This assumes that you have already created an Amazon account and have requested EC=2 access.

You will need a bash terminal/shell. Macs already have one. Windows users may want to download and install git bash. Then search for it on your laptop and open it.

You will also need a free github account

**Part I Setting up and accessing an AWS instance**

1. Sign into your AWS account and launch an EC-2 ubuntu instance (the 5th one down)

The different instance options represent different ‘Amazon Machine Images’ (AMIs) or software suites. For example there are deep learning AMIs that include python and pytorch.

As part of this process (after you click launch) you will be asked to create a key (xxxx.pem) This secure id allows you to access your instances remotely. Store the .pem file on your laptop (client) in a working directory you’ll access shortly.

Now go to EC2 on AWS and make sure your instance is ‘running.’ To start or stop it, right click on the box to the far left.

[Probably safer to stop your instance when you aren’t using it (though it should be free for a year)]

1. Connect to your new instance

Open the bash terminal on your laptop

Change directories (e.g. cd desktop) to the location of your key

Type this line, except use your key.pem file name and the ip address of your AWS instance

ssh -i xxxx.pem ubuntu@xx.xxx.xxx.xxx

1. OK! You are now connected to your instance.

Your instance already contains limited software tools. For example, type ‘python’ to see your version of python and then type some basic python commands such as ‘1+2+3’

But you can’t just can point and click to start programs or navigate folders. Usually people use AWS to run scripts written elsewhere (rather than writing scripts on AWS). They upload the scripts and data files and execute them on the instance using bash.

Git bash cheat sheet: <https://education.github.com/git-cheat-sheet-education.pdf>

Bash commands: <https://www.unr.edu/it/research-resources/the-grid/using-the-grid/bash-commands>

1. Moving files to your instance

The bash ‘scp’ command can be used to transfer files from your (client) laptop to your instance (e.g. )

scp -i xxxx.pem myfile.txt ubuntu@ec2-xxx-xxx-xxxx.com/home/ubuntu/myfile.txt

More commonly, users push their files to a GitHub repository, and then ‘clone’ the repository (repo) to the instance. Here’s how:

[Still connected to our instance via the bash shell]

1. Install git on the instance

sudo apt-get install git

1. Configure your git username and display name

 git config - -global user.email “enteryourgithubusername” (there are two dashes)

 git config - -global user.name “yourarbitraryname”

1. Make a directory to put the cloned repo. For example:

 cd (make a directory called desktop)

 cd desktop

1. Copy the repo link from the github account and clone!

 git clone <https://github.com/pjreddie/darknet.git>

(Darknet is an open source neural network framework written in C and CUDA)

1. Now if you type

 dir #directory

You should see ‘darknet.’ This means that your instance now contains the darknet repository.

1. cd #exit the desktop

---------------------------------------------------------------------------------

**Part II Installing a jupyter notebook to access from a browser**

Remember last week where we were able to access and run commands in a jupyter notebook via a web browser? This is how to create that option.

1. First we have to install jupyter notebook and to do that we have to install pip

sudo apt-get install python-pip #if it doesn’t work google ‘ubuntu pip install’ for solution

pip install jupyter notebook

1. Next we have to open an 8888 access port on the instance (just need to do this one time)

**[1]** Go to the AWS-ec2 panel and click on the security groups of your instance (far right).

**[2]** Select the **Inbound** tab in the lower panel and click to **Edit**

**[3]** Click on **Add Rule** and add a new **Custom TCP** connection

- **Type**: Custom TCP

- **Protocol**: TCP

- **Port Range**: 8888

- **Source**: Custom [0.0.0.0/0](http://0.0.0.0/0)

- **Description**: SSH for Admin Desktop

Getting there but still not finished! The steps below can be found at this link if you need clarification

<https://docs.aws.amazon.com/mxnet/latest/dg/setup-jupyter-configure-server.html>

$ cd

$ mkdir ssl

$ cd ssl

$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:1024 -keyout "cert.key" -out "cert.pem" -batch

$ ipython #If you don’t have ipython, pip install it using the command above

 iPythonPrompt> from IPython.lib import passwd

 iPythonPrompt> passwd()

Important! When you create your password, a hash (long string of characters) will be displayed. Copy it and save it to be copied and pasted later

Also, write down your password! (you’ll need it later)

$ exit

**Create a jupyter notebook configuration file**

$ jupyter notebook --generate-config

Open the jupyter notebook configuration file

$ vi ~/.jupyter/jupyter\_notebook\_config.py (could also use nano instead of vi)

This is the tricky part. You are inside of the config file. Scroll to the very bottom, and make sure that your cursor is at the very bottom. Then copy and paste the lines of code below. Carefully replace the hash in the last line with your own hash.

c = get\_config() # Get the config object.

c.NotebookApp.certfile = u'/home/ubuntu/ssl/cert.pem' # path to the certificate we generated

c.NotebookApp.keyfile = u'/home/ubuntu/ssl/cert.key' # path to the certificate key we generated

c.IPKernelApp.pylab = 'inline' # in-line figure when using Matplotlib

c.NotebookApp.ip = '\*' # Serve notebooks locally.

c.NotebookApp.open\_browser = False # Do not open a browser window by default when using notebooks.

c.NotebookApp.password = '*sha1:fc216:3a35a98ed980b9...*'

To exit the config file, press escape, then : then type ‘exit’

**Part III Getting really close now! Accessing the notebook from a browser**

Connect to your instance:

ssh -i vpg\_key.pem ubuntu@54.82.242.87 #only works if you are in the directory of your key

jupyter notebook #to start it

Connect to the appropriate port:

Open another new bash terminal (do not close the existing one).

Change to the directory of your key

Enter the following using your key and the ip address of your instance:

**ssh -i vpg\_key.pem -L 8157:**[**127.0.0.1:8888**](http://127.0.0.1:8888)**ubuntu@ec2-54-82-242-87.compute-1.amazonaws.com**

Make sure you are connected! If not, look for clues in the bash message

Connect via your web browser and login using the password you created earlier!

- Mac: <https://127.0.0.1:8157>

- Windows/PC: <https://ec2->54-82-242-87.[compute-1.amazonaws.com:8888/](http://compute-1.amazonaws.com:8888/)

[If you actually wanted to use your instance you would upload data files and scripts via bash or git, and access them from the browser-based notebook]

STOP YOUR INSTANCE ON THE AWS MANAGEMENT CONSOLE (closing the bash terminal won’t do it)

