Recitation 0D: Setting Up AWS Deep Learning AMI

Introduction to Deep Learning

Table of Contents

Video

Here is the link to the video that steps through this whole document and gives a gentle general workflow for the Part 2s of the Homework for Introduction to Deep Learning. Get your snacks!

EC2 Instances

Amazon EC2 provides a wide selection of ***instance*** types optimized to fit different use cases. Instance types comprise varying combinations of CPU, memory, storage, and networking capacity and give you the flexibility to choose the appropriate mix of resources for your applications.

Instances are classified based on machine size as: nano, micro, medium, large, xlarge, 2xlarge, …, 16xlarge.

What do we recommend?

1. Do all the testing of your model and verify the quality of your model on **t2.micro**.
2. When you are done, download your model, and open a **g4dn.xlarge (or p instance)**, and train your model there.
3. Other recommendations.

Why use different instances?

Save credits! You will have a limited amount of AWS credits throughout the semester. Your credits will have to last through HW1P2, HW2P2, HW3P2, HW4P2, and your semester project (for 11-785 students).

What do we put on EC2 Instances?

- Virtual images of existing machines
    - You can create an image of your machine
        - Transfer it to a different machine
        - Save it as a backup
- Use cases
    - Software packages that are incredibly difficult to install
    - Need to create multiple different machines with the exact same data for parameters servers
    - Load balancing - create a new machine with the same AMI to be used in a different region depending on load

Why do we need to know about EC2?

In this course we will be using large amounts of data (tens of GBs) to train neural networks and therefore:

- Anyone who doesn't have an NVIDIA GPU, will need to use EC2 instances to efficiently complete the homework.
- Anyone who doesn't have their GPU configured to be correctly used by Pytorch/Tensorflow/etc, will need to use EC2 instances to efficiently complete the homework.
- GPUs are good for training not processing data, if the code is configured to leverage a GPU.

Setup an AWS Account

Make an Account

- Sign up for your AWS account here.
- Login to your AWS account and follow the steps below.

Launching an EC2 Instance (General Workflow)

Choose Region and then click on Services

Choose a region in the top right of the page. If you are on the Pittsburgh campus we recommend `US East (N. Virginia) us-east-1` or `US East (Ohio) us-east-2`. The instances you make are linked to specific regions. After you have selected your region, click on services in the top left. Then select `EC2` under `Compute`.

Launching an Instance

1. Within the `Resources` module, select `Running instances`, then press `Launch Instance`.
2. Type `Deep Learning` into the search bar.
3. Select `Deep Learning AMI (Ubuntu 16.04) Version 26.0`.
4. Select the instance type you would like to use. For this class we recommend `t2.micro` for testing and `g4dn.xlarge` for training.
5. After selecting your instance, In the bottom right, press `Next: Configure Instance Details`.
6. Press `Next: Add Storage`.
7. You may need to add storage onto your instance depending on the size of the data we are working with for a given assignment. Adjust the initial value as necessary.
8. Continue pressing next until you get to the review page.
9. Launch your instance and select a key pair that you have access to.
10. Huzzah!
11. Click `View Instances` in the bottom right to see all of your instances listed.

Connecting to the Instance

Go to your terminal and ssh into your instance with the following command:

```
ssh -i deep_learning.pem -L 8000:localhost:8888 ubuntu@instance
```

The highlighted information will be dependent upon the name of your instance and the name of the pem file you created. There is more information about connecting to the jupyter notebook and getting files onto your instance in the Jupyter Notebook section.

Key Pair

- You can either create a key pair when you are creating an instance or create one from the EC2 dashboard under the `Resources` module. Just click on `Key Pairs`, click on `Create Key Pair`, then give a valid name like `deep_learning`.
- When you ssh into an instance, you have to use the pem file you specified in step 9 of Launching an Instance above.
- You technically only need one key pair for our course, but you can make more if you want.

If you will use an SSH client on a Mac or Linux computer to connect to your instance, use the following command to set the permissions of your private key file so that only you can read it.

```
chmod 400 deep_learning.pem
```

Using windows? (Options below, sorry about windows, also check Piazza...)

- Try using an Ubuntu VM.
- Install cygwin. It emulates linux commands nicely and you can chmod.
- (Try putty) https://www.clickittech.com/aws/connect-ec2-instance-using-ssh/

Getting GPU Permissions

You Will Need Permission to get an Instance with a GPU.

1. Go to this link.
2. Click `Create Case`.
3. Ensure `Service Limit Increase` is checked.
4. Ensure `EC2 Instances` is selected under the `Case Classification` module.
5. In the `Requests` module, select the region, instance type and the new limit value that you'd like. Ask for at least 4 in the **New limit value** section.
6. Under the case description, explain that you are taking a class in deep learning at a university and that you are requesting GPUs such that you can complete the homework assignments. You will usually be granted access, but if for some reason you are denied, please let us know, and you will need to open a new case.

Stopping the Instance

When you are done using your instance:

- IF you need to continue using the instance but would like to stop working for the night (or an extended period of time), then go to your running instances, select an instance, and select actions, instance state, stop.
    - When stopped, you will only be charged for storage (which is fairly cheap, but could add up.)
    - To start the instance back up, follow the same steps but select start.
- IF you are done with an instance, follow the same steps, but terminate instead of stopping the instance.

Spot Instances

- A cheaper alternative to regular instances are Spot Instances. You **cannot** stop these instances, which means that they will either be running or terminated.
    - When you terminate an instance, you cannot get any of the information back.
    - Spot instances are generally preferred (because they are cheaper) but they have a few extra steps involved if you want to save your information.
- To start a spot instance, after Step 5 of **Launching an Instance**, within **Configure Instance**, click **Request Spot instances**.
- You have to be careful though, because sometimes you can get kicked off of spot instances.
- Step by step guide for Spot Instance and EFS snapshot save and reattach.

Other options

- Get a Google Colab account and get a free 300$ to start out. Beware of other Colab bugs that have been extensively discussed on Piazza.

Billing

In billing you can check how much money you have spent as well as add credits to your account. Go to the billing page, and then go to the credit subcategory and put in the code we supply to you. You can check the Bills and the Cost Explorer too, to see how much money you are spending on a given day as well.

- To stop excessive spending,
    - Stop instances when they aren't in use.
    - Don't use GPUs to write your code, only to train.
    - Keep track of how much you are spending, you only get 150$ of AWS credits from us, that we give out to everyone at the same time.

Jupyter Notebook

We recommended that you use a Jupyter Notebook on an EC2 instance for homework. These are the commands and instructions that you will have to follow to get yourself setup.

SSH and install Kaggle

```
ssh -i deep_learning.pem -L 8000:localhost:8888 ubuntu@instance
sudo pip3 install kaggle
source activate pytorch_p36
```

Move Kaggle key into EC2 instance

Type kaggle into the ec2 instance, then scp the kaggle key into the instance from a local terminal. You will have to download the kaggle key from the kaggle website.

```
kaggle
scp -i deep_learning.pem ./kaggle.json ubuntu@instance:~/.kaggle/
```

Download data file

```
kaggle competitions download -c 11-785-s20-someHW#
```

Untar file

```
tar -zxvf NameOfTar.tar.gz
```

(If zip, install unzip, and then unzip it)

Open jupyter notebook

```
jupyter notebook --no-browser --port=8888
```

Upload files either via Jupyter or scp (below)

On local machine, to move file/folder to remote server

```
scp -i deep_learning.pem ./fileName ubuntu@instance:~/
scp -i deep_learning.pem -r ./folderPath/ ubuntu@instance:~/
```

On local machine, to move file/folder from remote server to current directory

```
scp -i deep_learning.pem ubuntu@instance:~/filePathRemote ./
scp -i deep_learning.pem -r ubuntu@instance:~/folderPathRemote/ ./
```

TMUX

When you step out for a cup of tea, or want to go to sleep, or want to go to Giant Eagle and pick up some potatoes, AND you would like to leave your model training, then you want to use TMUX. TMUX allows you to leave your code running, while your laptop is closed or whatnot. The following are a series of commands that are useful.

SSH into your instance, then start a window

```
tmux new -s awesomeSess
```

Making Panes

```
Ctrl-b % (Split the window vertically)
Ctrl-b " (Split window horizontally)
Ctrl-b o (use to traverse panes)
Ctrl-b x (kill panes)
```

Activate pytorch in a pane

```
source activate pytorch_p36
```

Open jupyter notebook and go ham coding

```
jupyter notebook --no-browser --port=8888
```

When done coding, covert code to a .py file, then run in terminal

```
jupyter-nbconvert --to script run.ipynb
python3 run.py
```

Detach from window! Your program is running and you can close the terminal.

```
Ctrl-b d
```

SSH back in, and attach again! And everything should be awesome!

```
tmux attach -t awesomeSess
```

When you're done, kill that sess

```
tmux kill-session -t awesomeSess
```

Notes

https://gist.github.com/michaellihs/b6d46fa460fa5e429ea7ee5ff8794b96

Workflow Example

https://towardsdatascience.com/jupyter-and-tensorboard-in-tmux-5e5d202a4fb6

Advanced usage (multi window)

```
# session management
tmux ls (or tmux list-sessions)
Ctrl-b c Create new window
Ctrl-b d Detach current client
Ctrl-b l Move to previously selected window
Ctrl-b n Move to the next window
Ctrl-b p Move to the previous window
Ctrl-b & Kill the current window
Ctrl-b , Rename the current window
Ctrl-b q Show pane numbers (used to switch between panes)
Ctrl-b ? List all keybindings

# moving between windows
Ctrl-b n (Move to the next window)
Ctrl-b p (Move to the previous window)
Ctrl-b l (Move to the previously selected window)
Ctrl-b w (List all windows / window numbers)
Ctrl-b window number (Move to the specified window number, the
default bindings are from 0 -- 9)

# Tiling commands
Ctrl-b q (Show pane numbers, when the numbers show up type the key to go to
that pane)
Ctrl-b { (Move the current pane left)
Ctrl-b } (Move the current pane right)

# Make a pane its own window
Ctrl-b : "break-pane"
```

Suggestions

If

- You have any suggestions on how to improve this document,
- Anything in this document was unclear,
- You have a really good Windows workflow and would like to share,
- You are looking for someone to email,

Please email me at cmgeorge@andrew.cmu.edu and I will try my best to improve the document accordingly.