Depending on the size of your training set and the speed of your CPU, you might be able to train your neural network on your local CPU. Training could take anywhere from 15 minutes to several hours if you train for many epochs.

A faster alternative is to train on a GPU.

It's possible to purchase your own NVIDIA GPU, or you may have one built into your machine already.

If not, it’s easy ([**although not free**](https://aws.amazon.com/ec2/pricing/on-demand/)) to access a GPU-enabled server (also known as an "instance") through Amazon Web Services.

**1. Create an AWS Account**

Visit [**aws.amazon.com**](https://aws.amazon.com/) and click on the "Create an AWS Account" button.

https://d17h27t6h515a5.cloudfront.net/topher/2016/December/58463173_create-an-aws-account/create-an-aws-account.png

If you have an AWS account already, sign in.

If you do not have an AWS account, sign up.

When you sign up, you will need to provide a credit card. But don’t worry, you won’t be charged for anything yet.

Furthermore, when you sign up, you will also need to choose a support plan. You can choose the free Basic Support Plan.

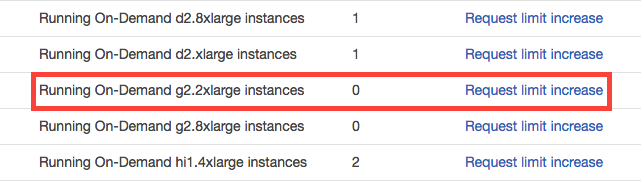
Once you finish signing up, wait a few minutes to receive your AWS account confirmation email. Then return to [**aws.amazon.com**](https://aws.amazon.com/) and sign in.

**2. View Your Limit**

View your EC2 Service Limit report at: [**https://console.aws.amazon.com/ec2/v2/home?#Limits**](https://console.aws.amazon.com/ec2/v2/home?#Limits)

Find your "Current Limit" for the g2.2xlarge instance type.

Note: Not every AWS region supports GPU instances. If the region you've chosen does not support GPU instances, but you would like to use a GPU instance, then change your AWS region.



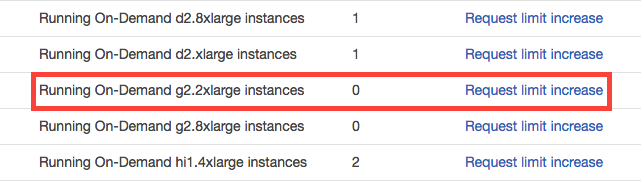
Amazon Web Services has a service called [**Elastic Compute Cloud (EC2)**](https://aws.amazon.com/ec2), which allows you to launch virtual servers (or “instances”), including instances with attached GPUs. The specific type of GPU instance you should launch for this tutorial is called “g2.2xlarge”.

By default, however, AWS sets a limit of 0 on the number of g2.2xlarge instances a user can run, which effectively prevents you from launching this instance.

**3. Submit a Limit Increase Request**

From the EC2 Service Limits page, click on “Request limit increase” next to “g2.2xlarge”.

You will not be charged for requesting a limit increase. You will only be charged once you actually launch an instance.

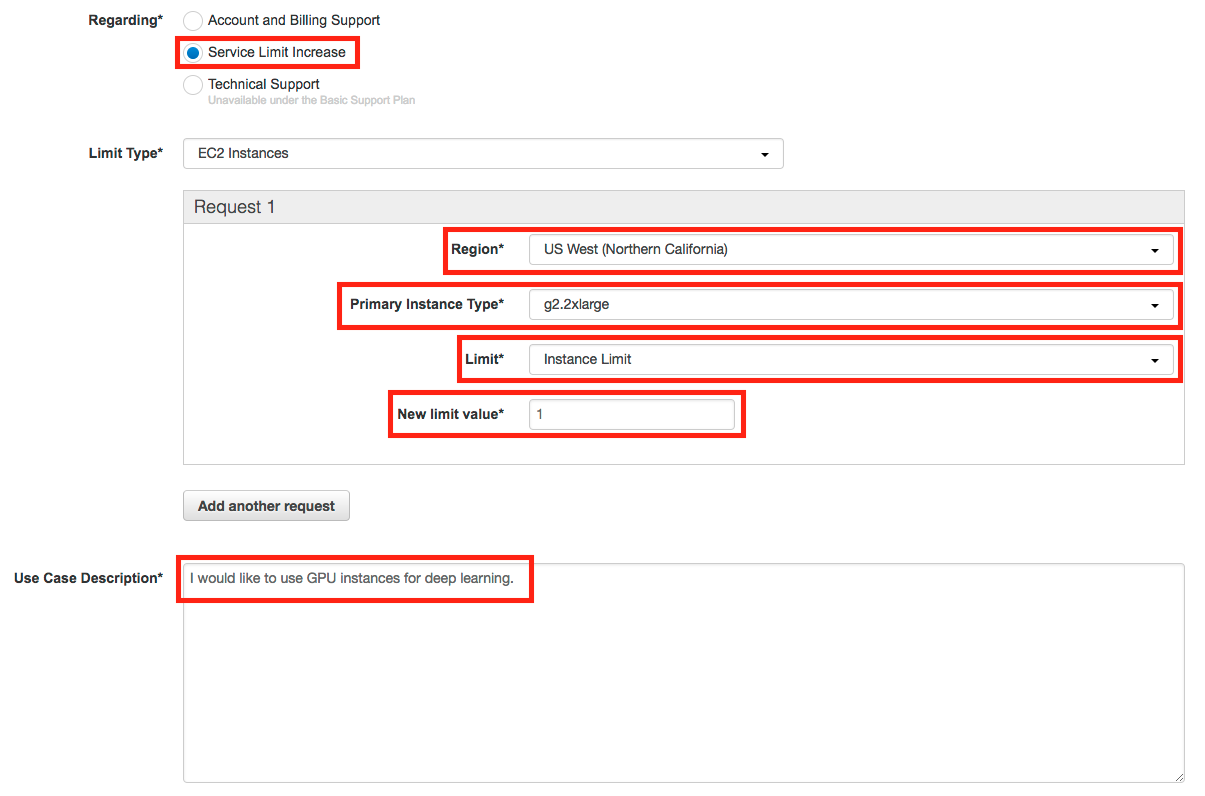


On the service request form, you will need to complete several fields.

For the “Region” field, select the region closest to you.

For the “New limit value”, enter a value of 1 (or more, if you wish).

For the “Use Case Description”, you can simply state: “I would like to use GPU instances for deep learning.”



Note: If you have never launched an instance of any type on AWS, you might receive an email from AWS Support asking you to initialize your account by creating an instance before they approve the limit increase.

**4. Wait for Approval**

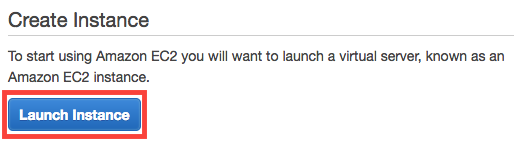
You must wait until AWS approves your Limit Increase Request. AWS typically approves these requests within 48 hours.

**5. Launch an Instance**

Once AWS approves your Limit Increase Request, you can start the process of launching your instance.

Visit the EC2 Management Console: [**https://console.aws.amazon.com/ec2/v2/home**](https://console.aws.amazon.com/ec2/v2/home)

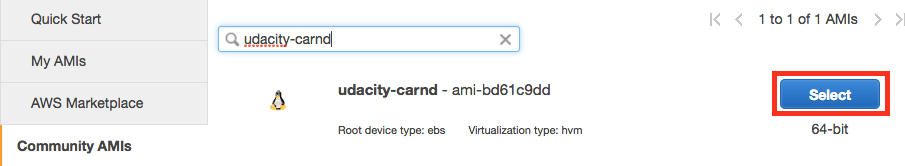
Click on the “Launch Instance” button.



Before launching an instance, you must first choose an AMI (Amazon Machine Image) which defines the operating system for your instance, as well as any configurations and pre-installed software.

We’ve created an AMI for you!

Search for the “udacity-carnd” AMI.

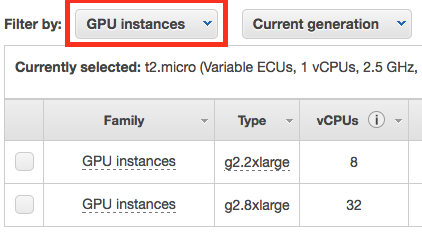


Click on the “Select” button.

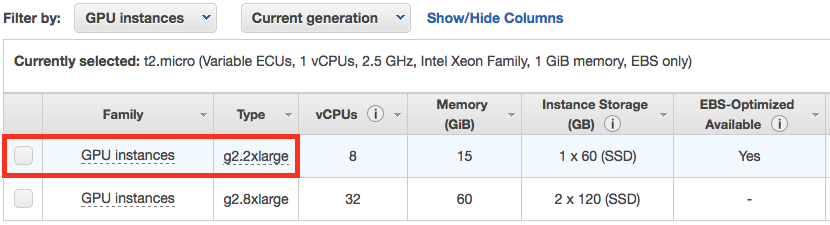
**6. Select the Instance Type**

You must next choose an instance type, which is the hardware on which the AMI will run.

Filter the instance list to only show “GPU instances”:



Select the g2.2xlarge instance type:



Finally, click on the “Review and Launch” button:

https://d17h27t6h515a5.cloudfront.net/topher/2016/December/584633cb_review-and-launch/review-and-launch.png

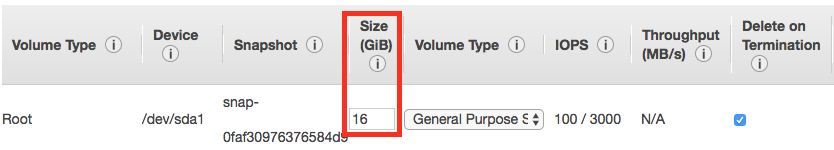
**7. Add Storage**

Your instance is now configured and ready for launch, but it would probably help to add storage above and beyond the 8GB that come with the g2.2xlarge instance by default.

Click on “Edit Storage”:

https://d17h27t6h515a5.cloudfront.net/topher/2016/December/584633f6_edit-storage/edit-storage.png

Increase the storage size to 16 GB (or more, if necessary):



Click on the “Review and Launch” button again.

**8. Configure the Security Group**

Running and accessing a Jupyter notebook from AWS requires special configurations.

Most of these configurations are already set up on the udacity-carnd AMI. However, you must also configure the security group correctly when you launch the instance.

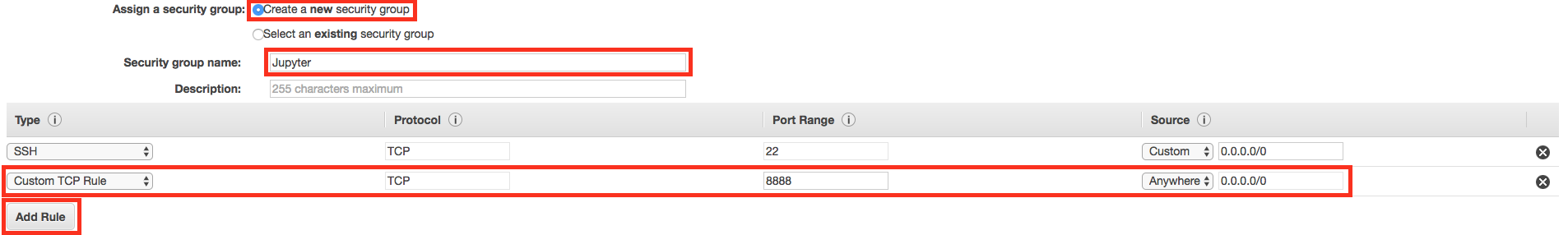
By default, AWS restricts access to most ports on an EC2 instance. In order to access the Jupyter notebook, you must configure the AWS Security Group to allow access to port 8888.

Click on "Edit security groups".

https://d17h27t6h515a5.cloudfront.net/topher/2016/December/58543293_edit-security-group/edit-security-group.png

On the "Configure Security Group" page:

1. Select "Create a **new** security group"
2. Set the "Security group name" (i.e. "Jupyter")
3. Click "Add Rule"
4. Set a "Custom TCP Rule"
   1. Set the "Port Range" to "8888"
   2. Select "Anywhere" as the "Source"
5. Click "Review and Launch" (again)



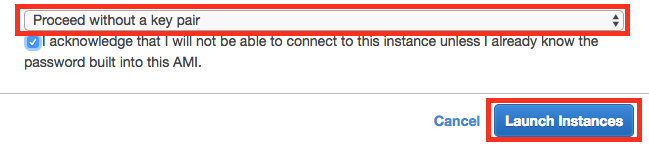
**9. Launch the Instance**

Click on the “Launch” button to launch your GPU instance!

https://d17h27t6h515a5.cloudfront.net/topher/2016/December/58463444_launch/launch.png

**10. Proceed Without a Key Pair**

Oops. Before you can launch, AWS will ask if you’d like to specify an authentication key pair.



In this case the AMI has a pre-configured user account and password, so you can select “Proceed without a key pair” and click the “Launch Instances” button (for real this time!).

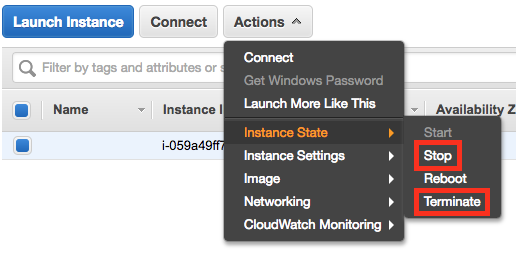
Next, click the “View Instances” button to go to the EC2 Management Console and watch your instance boot.

**11. Be Careful!**

From this point on, AWS will charge you for a running an EC2 instance. You can find the details on the [**EC2 On-Demand Pricing page**](https://aws.amazon.com/ec2/pricing/on-demand/).

Most importantly, remember to “stop” (i.e. shutdown) your instances when you are not using them. Otherwise, your instances might run for a day or a week or a month without you remembering, and you’ll wind up with a large bill!

AWS charges primarily for running instances, so most of the charges will cease once you stop the instance. However, there are smaller storage charges that continue to accrue until you “terminate” (i.e. delete) the instance.

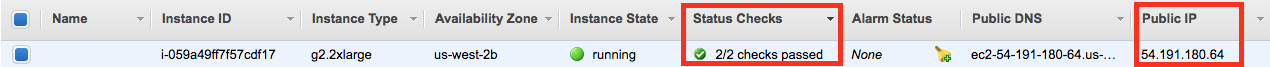


There is no way to limit AWS to only a certain budget and have it auto-shutdown when it hits that threshold. However, you can set [**AWS Billing Alarms**](http://docs.aws.amazon.com/awsaccountbilling/latest/aboutv2/free-tier-alarms.html).

**12. Log In**

After launch, your instance may take a few minutes to initialize.

Once you see “2/2 checks passed” on the EC2 Management Console, your instance is ready for you to log in.



Note the "Public IP" address (in the format of “X.X.X.X”) on the EC2 Dashboard.

From a terminal, SSH to that address as user “carnd”:

ssh carnd@X.X.X.X

Authenticate with the password: carnd

**13. Launch a Jupyter Notebook**

Congratulations! You now have a GPU-enabled server on which to train your neural networks.

Make sure everything is working properly by verifying that the instance can run the [**LeNet-5 lab solution**](https://github.com/udacity/CarND-LeNet-Lab/blob/master/LeNet-Lab-Solution.ipynb).

On the EC2 instance:

1. Clone the LeNet Lab repo: git clone https://github.com/udacity/CarND-LeNet-Lab.git
2. Enter the repo directory: cd CarND-LeNet-Lab
3. Activate the new environment: source activate carnd-term1
4. Run the notebook: jupyter notebook LeNet-Lab-Solution.ipynb

**14. Run the Jupyter Notebook**

From your local machine:

1. Access the Jupyter notebook index from your web browser by visiting: X.X.X.X:8888 (where X.X.X.X is the IP address of your EC2 instance)
2. Click on the "LeNet-Lab-Solution.ipynb" link to launch the LeNet Lab Solution notebook
3. Run each cell in the notebook

It took me 7.5 minutes to train LeNet-5 for ten epochs on my local CPU, but only 1 minute on an AWS GPU instance!

# Traffic Sign Classifier Project

In this project, you will use what you've learned about deep neural networks and convolutional neural networks to classify traffic signs. Specifically, you'll train a model to classify traffic signs from the [**German Traffic Sign Dataset**](http://benchmark.ini.rub.de/?section=gtsrb&subsection=dataset).

After you train your model, you will test it on new images of German traffic signs you find on the web.

## Setup

1. [**Download the dataset**](https://d17h27t6h515a5.cloudfront.net/topher/2017/February/5898cd6f_traffic-signs-data/traffic-signs-data.zip). This is a pickled dataset in which we've already resized the images to 32x32.
2. Clone the project and start the notebook.
3. git clone https://github.com/udacity/CarND-Traffic-Sign-Classifier-Project
4. cd CarND-Traffic-Sign-Classifier-Project

#### TensorFlow

If you have access to a GPU, you should follow the TensorFlow instructions for [**installing TensorFlow with GPU support**](https://www.tensorflow.org/get_started/os_setup#optional_install_cuda_gpus_on_linux).

Once you've installed all of the necessary dependencies, you can install the tensorflow-gpu package:

pip install tensorflow-gpu

#### Amazon Web Services

Instead of a local GPU, you could use Amazon Web Services to launch an EC2 GPU instance. (This costs money.)

1. [**Follow the Udacity instructions**](https://classroom.udacity.com/nanodegrees/nd013/parts/fbf77062-5703-404e-b60c-95b78b2f3f9e/modules/6df7ae49-c61c-4bb2-a23e-6527e69209ec/lessons/614d4728-0fad-4c9d-a6c3-23227aef8f66/concepts/f6fccba8-0009-4d05-9356-fae428b6efb4) to launch an EC2 GPU instance with the udacity-carnd AMI.
2. Complete the **Setup** instructions.

## Start the Project

1. Launch the Jupyter notebook: jupyter notebook Traffic\_Sign\_Classifier.ipynb
2. Follow the instructions in the notebook
3. Write your project report

## Project Submission

When you are ready to submit your project, collect the following files and compress them into a single archive for upload. Alternatively, upload your files to github to link to the project repository:

* The Traffic\_Sign\_Classifier.ipynb notebook file with all questions answered and all code cells executed and displaying output.
* An HTML or PDF export of the project notebook with the name report.html or report.pdf.
* Any additional datasets or images used for the project that are not from the German Traffic Sign Dataset. **Please do not include the project data set provided in the*traffic-sign-data.zip*file.**
* Your writeup report as a markdown or pdf file

You can view the complete rubric for the project [**here**](https://review.udacity.com/#!/rubrics/481/view).