

Getting started with Google Cloud

The Google Cloud Platform (GCP) is a cloud service offering compute power. In this class you will receive a sign up link giving you access to 150 USD in compute resources, which gives you about 280 hours of compute time with a NVIDIA Tesla K80 GPU.

This guide will teach you how to get started on GCP to set up a virtual machine instance with all the required frameworks.

IMPORTANT!!! Remember to turn off your instance when you are done working on it! If not, it will continue running and drain your credit!

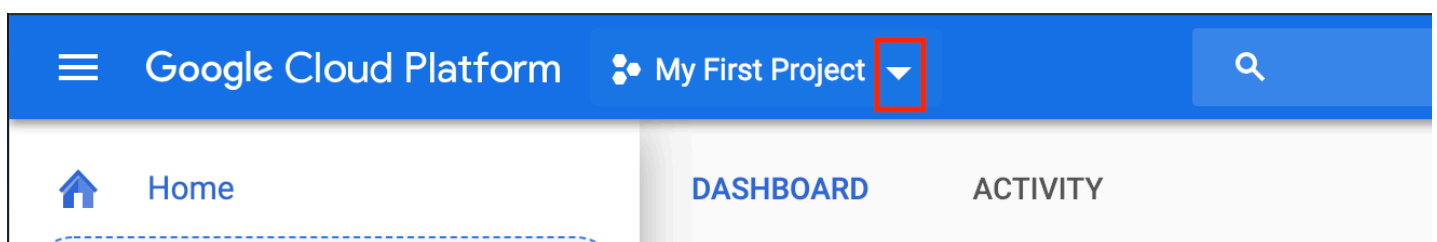
1. Sign up

Go to <https://cloud.google.com/free/> to sign up for google cloud. This will give you 300\$ in credits if you have never used it before. GCP sign up might require credit card signup, but they will not withdraw anything from your account while you have credit on GCP.

2. Create your first project

Go to <https://console.cloud.google.com> to view the GCP console panel. Here you can see your available credit (in the billings tab), current running engines and more.

In the top, click on the project bar as shown in the image. Then create a new project, call it TDT4265



3. Create your deep learning environment

Here we will set up your virtual environment for your project. This procedure works for both Tensorflow and Pytorch

Go to: <https://console.cloud.google.com/marketplace/details/click-to-deploy-images/deeplearning>

and click "LAUNCH ON COMPUTE ENGINE". Then, select the project we created in the previous step.

Fill your compute preferences here. We recommend a setup as shown on the image below, but you are

freely available to choose a different setup. The NVIDIA P100 is significantly faster than the K80, but it comes at a higher cost. Also, the P100 has about 4GB more VRAM than the K80. You can read more about the performance of these GPU's here: <https://www.tensorflow.org/guide/performance/benchmarks>. The performance of the GPU's are about the same for Pytorch and Tensorflow.

Remember, if you want to use Tensorflow instead of Pytorch, you need to change the framework to "Intel optimized TensorFlow 1.12".



New Deep Learning VM deployment

Deployment name

pytorch-gpu1

Zone ?

GPU availability is limited to certain zones. [Learn more](#)

europa-west1-b

Machine type ?

4 vCPUs

26 GB memory

[Customize](#)

GPUs

The number of GPU dies is linked to the number of CPU cores and memory selected for this instance. For this machine type, you can select no fewer than 1 GPU die.

[Learn more](#)

Number of GPUs

1

GPU type

NVIDIA Tesla K80



Machines with GPUs can't migrate on host maintenance

Framework

Choose the primary machine learning framework you will be using. If the library you would

Choose the primary machine learning framework you will be using. If the library you would like to use is not listed, choose the base image, which provides core packages.

PyTorch 1.0 + FastAi 1.0 (CUDA 10.0)

GPU

☒ Install NVIDIA GPU driver automatically on first startup? 





I want to use NVIDIA GPUs with this image. Please fetch NVIDIA GPU drivers from a third-party location and install them on my behalf (requires internet access on the VM).

NB: You might encounter an error telling you that the GPU quota is exceeded. This can be fixed by going to <https://console.cloud.google.com/iam-admin/quotas>, and select "GPUs (all regions)" as metric. If this limit is 0, you need to edit it by clicking "Edit Quota" in the top of the page. This will send a request to the GCP team, and you will receive a reply within short time (took 5 minutes for me). After you received a reply, perform the previous step again!

4. Launch your VM instance

Go to <https://console.cloud.google.com/compute/instances>, which gives you a list of your VM instances. Hopefully, you will see the VM we created in the previous step. In my case, the instance was running after the first step. If it is not, you can start it by clicking the dots on the right side of the list.

SSH is used to connect to the VM instance. GCP offers several ways to connect through SSH, as shown in the image below. The red annotation shows you how to start and stop the instance.

Name	Zone	Recommendation	Internal IP	External IP	Connect
<input checked="" type="checkbox"/>  pytorch-10-vm	europe-west1-b		10.132.0.2 (nic0)	None	SSH  
<input type="checkbox"/>  pytorch-gpu-vm	europe-west1-b		10.132.0.3 (nic0)	None	SSH

CLICK THIS TO START/STOP INSTANCE

- Open in browser window
- Open in browser window on custom port
- View gcloud command
- Use another SSH client

Whichever way you choose to connect with, you will end up in a bash window. From this point, it should be straightforward to run your code on the cloud. Transfer your code to the cloud (we recommend git for this), and start your training script: `python name_of_train_script.py`.

If you are not familiar with bash, these are some quick tutorials to get your started. It should be enough for this course:

- [Git quick tutorial](#)
- Run a python script: `python name_of_my_python_file.py`
- [Bash navigation tutorial](#)

IMPORTANT!!! Remember to turn off your instance when you are done working on it! If not, it will continue running and drain your credit.

5. Tips & Tricks

Working on a remote server can be a pain, but these tricks will (hopefully) make your life a bit better.

First of all, **DO NOT DEVELOP YOUR CODE ON CLOUD**. This will just drain your GCP credit and it is always better to work on your local machine! Only use the cloud when you are sure that your code is working, and you want to train your network for a longer time.

To transfer your code to the server, we recommend you to use **git**.

If you want to work directly on the server, we recommend you to take a look at [sshfs](#). This is a tool that is able to mount a folder on a remote server to your computer, such that you can use VSCode/Pycharm/.. to edit your code. This might be a bit complicated to set up, but it is worth it in the end!

Useful tools:

- `nvidia-smi` : This shows the GPU Usage for both VRAM and core utilization
- `htop` : See the CPU utilization
- `vim` : Text editor in bash. It is a bit complicated to get familiar with, but is useful for quick bug fixes
- `nano` : Similar to vim, but a bit simpler to get started with if you're not familiar with vim.

Launching Jupyter Notebook on the server

Stanford CS231n course has a good tutorial on how to launch a jupyter notebook on the cloud. This can be found [here](#) (skip to the chapter named "Using Jupyter Notebook with Google Compute Engine")