**Module 5**

**Deep learning with High Performance Computing**

**Outline**

* **Introduction**
* **Image Classification Example**
  + **Cats and Dogs Image classifier**
* **High Performance Computing**
* **GPUs stack up against CPUs**
* **How you can get an access to GPU**

**Introduction**

Machine learning (ML) is basically a mathematical and probabilistic model which requires tons of computations. Also, artificial neural networks (ANN) are another algorithmic approach from the early machine learning crowd that helped AI get closer to General AI. Andrew Ng put the “deep” in deep learning (DL) which describes all the layers in artificial neural networks.

The ML flow consists of four steps, from preprocessing input data, training the DL model, storing the trained DL model to deployment of the model. Among all these steps, training the DL model is the most computationally intensive task. Because training process requires a huge amount of data which are processed in hidden layers using wights, hence the large computational operations in terms of memory. Needing a lot of hardware for ML makes GPU (Graphics Processing Unit) as the heart of DL. Your laptop certainly does not have a GPU unless you buy a gaming laptop.

In this module, you can also recognize the limitation you will reach during the process of designing and implementing algorithms for resource-intensive tasks. You will discover that a bottleneck may appears during the training of a simple DL model which takes hours, days, and even years with billion parameters on a laptop, typically without GPU.

At the end of this module, you will see the role of High-Performance Computing (HPC) in building of the infrastructure of ML projects and the difference between CPU (Central Processing Unit) and GPU in terms of DL models. This module shows you HPC became an essential to build DL algorithms within data centers.

**Image Classification Example**

This part of module provides you a simple problem of image classification that leads to an impression that DL requires big systems to run execute. Before dive deep into hardware for this project, let’s see how usual performance computing in your laptop stuck in a CPU bottleneck for a long time. The goal is showing you that a CPU equipped system often is not fast enough to process a DL project.

We highly encourage you to try your own personal project which you are interested in to see difference of running time between a high-performance hardware and a normal personal laptop. You can use whatever libraries you like

**Cats and Dogs Image classifier**

You can access the Jupyter notebook[[1]](#footnote-1) here to see all details in the code. A fully connected convolutional neural network (CNN) with a single hidden layer for image classification model is used which can learn to recognize cats from dogs with at most 0.102 error rate. The library that is used for this project is fastai which is built on top of Pytorch and the [Oxford-IIIT pet dataset](https://www.tutorialspoint.com/jupyter/index.htm) with 37 categories of pet with roughly 200 images for each class.

According to the notebook, the training process took “2:24:00” minutes for four epochs. However, if you use a small GPU like what Google Colaboratory provids its users (you will see more details about Colab) the total running time for training is “01:46”

Table

Description automatically generated

With a free GUP of Google Colab took 1:46

Graphical user interface, text, application

Description automatically generated

Run on a personal computer almost 2 and half hours

Table

Description automatically generated

This is a simple example of why GPUs are more suited for DL projects. HPC allows developers and researchers to try the development of AI.

**High Performance Computing**

Think about this: how many times were you frustrated because you had a slow internet? In the DL research world, the same question can be asked when we were annoyed because of the insane time of iteration to train a model with many hyper-parameters. Furthermore, there is a continuous inflow of data which needs to be addressed for retraining models. HPC approaches is being used in develop, redesign, model products, and analyze large datasets. An HPC system consists of hundreds to thousands of physical servers, each powered by powerful processors. It uses parallel programming paradigms rather than sequential programming. HPC servers are large numbers of GPU cores and PU cores working in parallel, permitting complex workloads like DNN for machine translation tasks in natural language processing.

Here is [the link](https://towardsdatascience.com/a-beginners-tutorial-to-jupyter-notebooks-1b2f8705888a?v=c_55gZfUK1E) for a video from Dr. Andrew Ng talking about HPC and deep learning for more references. The following plot shows the reduction in time to train for a neural machine translation with a HPC approach.

Chart

Description automatically generated with low confidence

HPC is indeed the next frontier for enterprise AI and a competitive advantage in data intensive workloads.

**GPUs stack up against CPUs**

Parameters to determine whether to use a CPU or a GPU to train a DL model can be:

* Memory bandwidth,
* Dataset size,
* Optimization,
* Cost efficiency

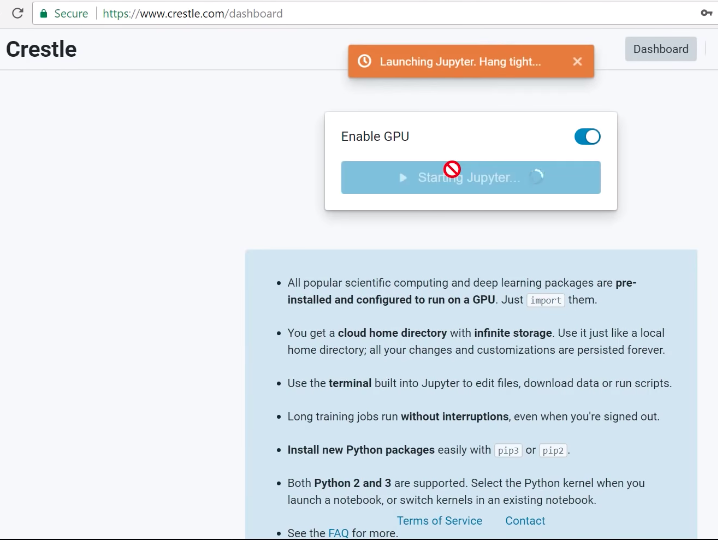
CPU takes up a lot of memory while training the model, however GPU uses Video RAM memory like Tesla and GeForce. The larger dataset comes with more complex computations and GPU can be an optimum choice. On the other hand, complex optimization techniques are difficult to implement in a GPU than a CPU and the power cost of GPU is higher than CPU.

So, deciding of using GPU depends on the tradeoff between speed, reliability, and cost. If your model is relatively small with a few hidden layers or if you are seeing 2-3 times performance gains in GPU-trained models, CPU can be enough. However, if your model uses many parameters and tones of calculations then you can consider investing in a GPU.

**How you can get an access to GPU**

You are going to need an NVIDIA GPU because it is the only company support CUDA-enabled GPU for general purpose processing. CUDA is the language and framework that almost all DL libraries and practitioners use. Therefore, you will need to rent a GPU for speeding up your compute-intensive applications. There are a couple of options when you choose using GPUs.

1. Crestle: it is easy to setup and use. GPU drivers and tensorflow come pre-installed. Crestle offers you 10 hours free trial (to this date March 2021). If you go to [www.crestle.com](https://cloud.google.com/) and sign up, enable GPU, and then click start Jupyter. The framework brings us into Jupyter notebook (one of the most popular tool in the data scientist toolbox)

Graphical user interface, text, application, email

Description automatically generated

1. Paperspace: is a GPU accelerated cloud platform. It runs on their own machine unlike Crestle. To start click on this link: [https://www.paperspace.com/console/machines](https://www.youtube.com/watch). You need to create a new machine and choose region, template, and storage. Payment is needed to use Paperspace’s GPU. You can see all detailed steps to start working with Paperspace.

A screenshot of a video game

Description automatically generated

1. Google cloud platform (Compute engine): [https://cloud.google.com/](https://www.datacamp.com/community/tutorials/tutorial-jupyter-notebook) currently offers 3-month free trial after that it’s about $0.04 hourly depend on your choices. You can choose your machine configuration and GPU type and its numbers. Pre-installed libraries like Keras, Pytorch, Tensorflow, Numpy, Pandas, and many others are available for users.

Text

Description automatically generated

1. Google Colaboratory (Colab): Colab resides within your Google Drive. You can create your own Google Colab by clicking on your Google Drive > New > Google Colaboratory. Google Colab provides two cores and 13 GB of RAM. You can choose the hardware accelerator between GPU and TPU. It might give you a Tesla K80 or a P100 or a V100 based on availability. In fact, Colab is a sub-part of Google Cloud for free! So, it can be a wise choice for trying your personal research. One of the drawbacks of using Colab is the limited maximum runtime which is 12 hours.

Graphical user interface, application

Description automatically generated

You can check this [Reddit post](http://www.crestle.com) for your reference about best GPUs for DL projects and top 10 GPU manufacturers [here](https://data-flair.training/blogs/deep-learning-project-ideas/).

**Write your first GPU-based project**

So far you have learned why accelerating DL projects is getting more attention and the role of HPC in computationally intensive training is the core of DL project. You are familiar with some famous platform that provide GPU. It is time to practice your own GPU-based project and experience challenges. You can either choose your interested topic and search for available dataset or look up into some well-known DL projects ideas, [here](https://www.upgrad.com/blog/exciting-deep-learning-project-ideas-for-beginners/) and [here](https://www.paperspace.com/console/machines).

References

# A Beginner’s Tutorial to Jupyter Notebooks. [https://towardsdatascience.com/a-beginners-tutorial-to-jupyter-notebooks-1b2f8705888a](https://www.robots.ox.ac.uk/~vgg/data/pets/)

# Jupyter Tutorial. [https://www.tutorialspoint.com/jupyter/index.htm](https://www.reddit.com/r/MachineLearning/comments/b95182/d_which_gpus_to_get_for_deep_learning_my/)

# Jupyter Notebook Tutorial: The Definitive Guide. [https://www.datacamp.com/community/tutorials/tutorial-jupyter-notebook](https://roboticsandautomationnews.com/2017/08/11/top-10-graphics-processing-unit-manufacturers-nvidia-clearly-in-the-lead/13709/)

1. Please see the references for provided online tutorials about Jupyter Notebook [↑](#footnote-ref-1)