A Cloud Service Provider Platform for Big Data and Machine Learning Workloads

Executive Summary

This was a lot of hype recently about Machine Learning and DNN. People usually just associate the processing of machine learning to 1) Selecting a Model; 2) Using Training Data and backward propagation techniques to build the model; and 3) Selecting the best Model and verify against Test Data.

Although deep learning provides a new approach to solve many complex problems in image analysis, speech recognition, and natural language processing, the process of using DNN to do Machine Learning is rarely a standalone workload. The process of Machine Learning actually includes:

1. The process of Extracting and pre-processing of training data from real life scenarios (Videos, conversations streams, Manufacturing line data, Medical records, Chess games, etc)

2. Selecting a model and use backward propagation tools (Tensorflow, Caffe, etc) to iteratively derive the optimal standard deviation between the model and test data
3. Uploading the “optimal model” to the inference engine and collect more training data and repeat the process to further improve accuracy.

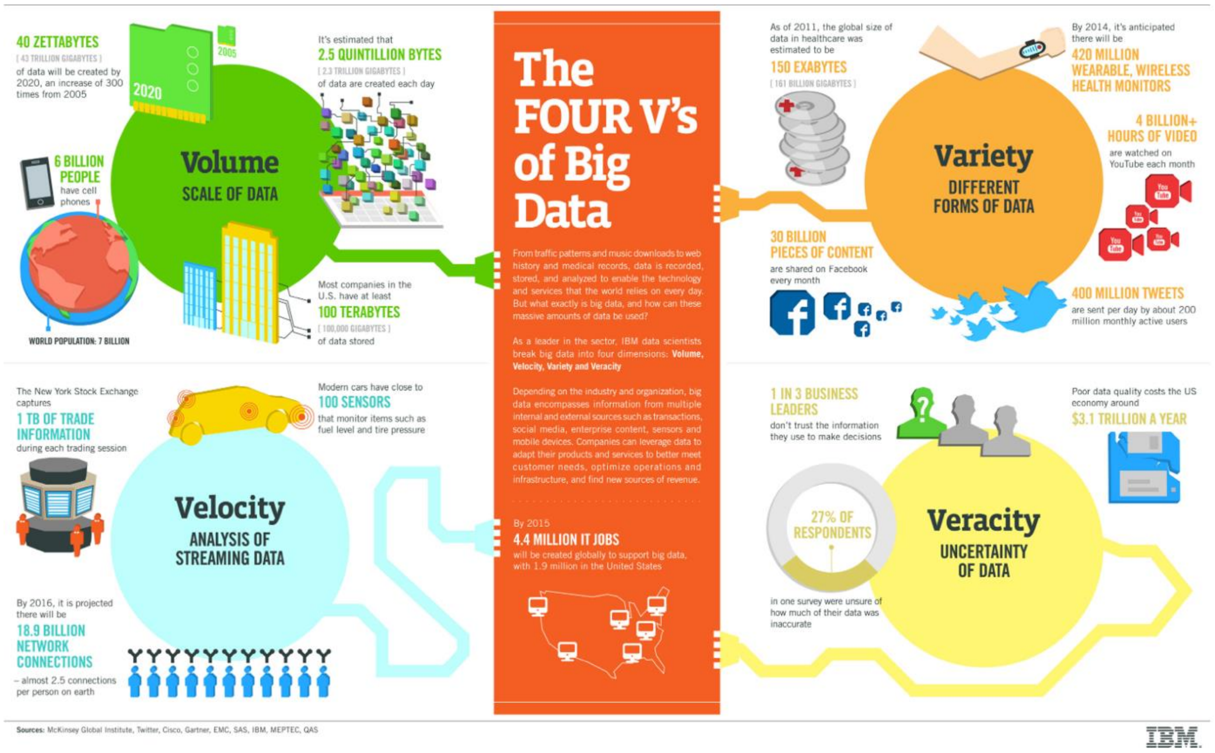
The objective of this article is to go deeper into each of these Machine Learning phases, understand its theory, and derive a platform that’s suited for the entire Machine Learning process.

Introduction

Legacy HPC & Big Data Computing

Let’s start with the traditional HPC computing and Big Data processing. HPC computing has been around for decades and everyone has been talking about Big data analysis since the dawn of the Century because the Internet has created an explosive growth of data in terms of both content and user behavioral statistics.

In the early stages, HPC computing is primarily for scientists and has relatively little commercial meanings. However, the birth of World Wide Web and easy access to powerful search engines has led to explosive information growth in the 90’s. People start to recognize the power of social networking and data mining of people’s behavior on the Internet has profound commercial implications.


40 ZETTABYTES
[43 TRILLION GIGABYTES]
of data will be created by 2020, an increase of 300 times from 2005
2020
2005
It's estimated that
2.5 QUINTILLION BYTES
[2.3 TRILLION GIGABYTES]
of data are created each day
Volume
SCALE OF DATA
6 BILLION PEOPLE
have cell phones
WORLD POPULATION: 7 BILLION
Most companies in the U.S. have at least
100 TERABYTES
[100,000 GIGABYTES]
of data stored
The New York Stock Exchange captures
1 TB OF TRADE INFORMATION
during each trading session
Modern cars have close to
100 SENSORS
that monitor items such as fuel level and tire pressure
Velocity
ANALYSIS OF STREAMING DATA
By 2016, it is projected there will be
18.9 BILLION NETWORK CONNECTIONS
– almost 2.5 connections per person on earth
As of 2011, the global size of data in healthcare was estimated to be
150 EXABYTES
[161 BILLION GIGABYTES]
By 2014, it's anticipated there will be
420 MILLION WEARABLE, WIRELESS HEALTH MONITORS
Variety
DIFFERENT FORMS OF DATA
4 BILLION+ HOURS OF VIDEO
are watched on YouTube each month
30 BILLION PIECES OF CONTENT
are shared on Facebook every month
400 MILLION TWEETS
are sent per day by about 200 million monthly active users
1 IN 3 BUSINESS LEADERS
don't trust the information they use to make decisions
Poor data quality costs the US economy around
$3.1 TRILLION A YEAR
27% OF RESPONDENTS
in one survey were unsure of how much of their data was inaccurate
Veracity
UNCERTAINTY OF DATA
By 2015
4.4 MILLION IT JOBS
will be created globally to support big data, with 1.9 million in the United States
IBM.

The FOUR V's of Big Data

From traffic patterns and music downloads to web history and medical records, data is recorded, stored, and analyzed to enable the technology and services that the world relies on every day. But what exactly is big data, and how can these massive amounts of data be used?

As a leader in the sector, IBM data scientists break big data into four dimensions: **Volume, Velocity, Variety and Veracity**

Depending on the industry and organization, big data encompasses information from multiple internal and external sources such as transactions, social media, enterprise content, sensors and mobile devices. Companies can leverage data to adapt their products and services to better meet customer needs, optimize operations and infrastructure, and find new sources of revenue.

Sources: McKinsey Global Institute, Twitter, Cisco, Gartner, EMC, SAS, IBM, MEPTEC, QAS

The most well-known characteristics of Big Data is the 4 V's of Big data – Volume, Velocity, Variety, and Veracity. The technology behind how to tackle these problems are also well understood. In the early 2000's, thanks to a bunch of Engineers at Google & Yahoo!, they have developed and released the open source Hadoop platform. The Hadoop platform primarily includes a set of data extraction tools, a data repository (HDFS), a Map Reduce framework, and a YARN task scheduler. Since then, most of the popular data collection, analysis, and visualization applications were developed with the Hadoop ecosystem and pretty much people think they have the big data problem well in control.

Cloud & HPC

Besides Big Data, another paradigm shift brought on by the Internet era is Cloud computing and Mobile computing. This is primary due to two reasons. Firstly, with massive amount of data collected, people start to realize there's no way we can store all the data locally inside your workstation. Instead of bringing the data to the computer, we virtualize computing and bring the computing power (i.e. server) to the data. The virtualization technology primarily contributed to this paradigm by creating computing power on-demand in the cloud to process the Big Data.

The other reason Cloud becomes popular is massive availability of smart phones and mobile computing devices. Because people are moving around all the time, they would like to access their data anytime, anywhere with any device. Mobile computing and Cloud also allows people “context” to become a relevant factor in their engagement experience. By “context”, I meant people’s location, their habits, as well as the environment around them. These ‘environmental’ factors can affect greatly the result of people’s search, as well as their social collaboration behavior. In short, the availability of data anywhere, any time, and context-sensitive computing, has dramatically changed how people interact with computing in the last 2 decades.

Although Cloud and Big Data technologies have matured for over 2 decades, it’s apparent the two solutions have not done much leverage of each other. Most HPC computing still are being done on workstations and physical machine clusters. That's until recently, with the popularity of AI and Machine Learning. We will talk about that next.

Machine Learning (DNN) workload

In 2015, Google’s DeepMind team in London developed a new program called “AlphaGO”. It became the first computer Go program to beat a human professional Go player without handicaps on a full-sized 19×19 board. Since then, the Machine Learning paradigm using DNN (Deep Neural Network) has caught on fire and everyone is talking about it. DNN is based on a new theory called Neuro-evolution. It basically tries to learn from a very large dataset and do forward supervised learning and backward propagation to come up with an algorithm (aka ‘model’) so the machine can derive its own conclusion to new incoming data. DNN and Artificial Intelligence and Machine Learning seems to open up numerous new possibilities.

While the concept above is relatively straight forward, the machine learning process is very compute intensive. Kind of similar to the Big Data 4V problem, machine learning is highly Iterative process that requires a lot of Matrix and Vector computations. This is where the Cloud becomes very interesting for machine learning. Matrix and Vector computation requires a lot of CPU processing that are better handled by GPU’s instead of the traditional CPU’s. GPU’s are very expensive that it is not practical for an individual and a workstation to possess all the GPU power for a Machine Learning process. Hence, it becomes much more efficient and cost effective to have the GPU computing available on-demand in a commonly shared environment (the cloud) for Data Scientists and AI Engineers.

This is key. Let me repeat the Morale of the Story here:

- AI and ML can benefit by Matrix & Vector processing
- Matrix & Vector computations can best be done on GPU’s

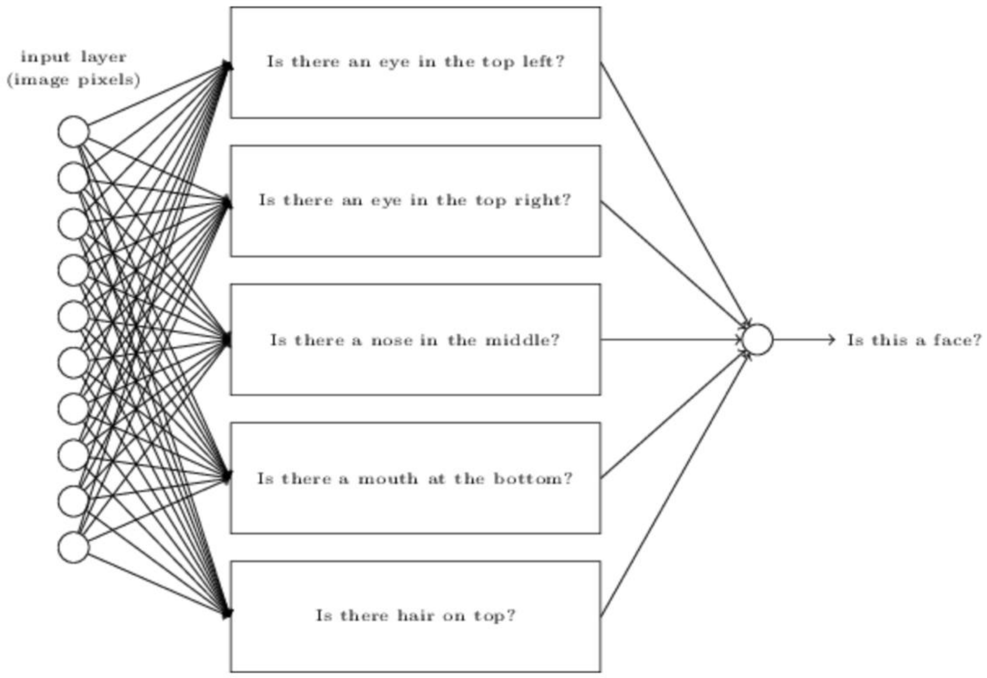
- GPU are expensive, dedicating a GPU to a few workstations and VM's is not very cost effective
- Dockers & Kubernetes allows GPU's to be effectively shared by a large group of user with very low penalty on context switch
- "Utility Computing" is a fancy name for Batch processing, but it serves the very well for the purpose of sharing workload among Data Scientists and AI Engineers.
- An ideal Cloud environment needs to have both "Capacity computing" as well as "Utility computing" modes

We will next go into the theory on DNN and providing GPU as a Service.

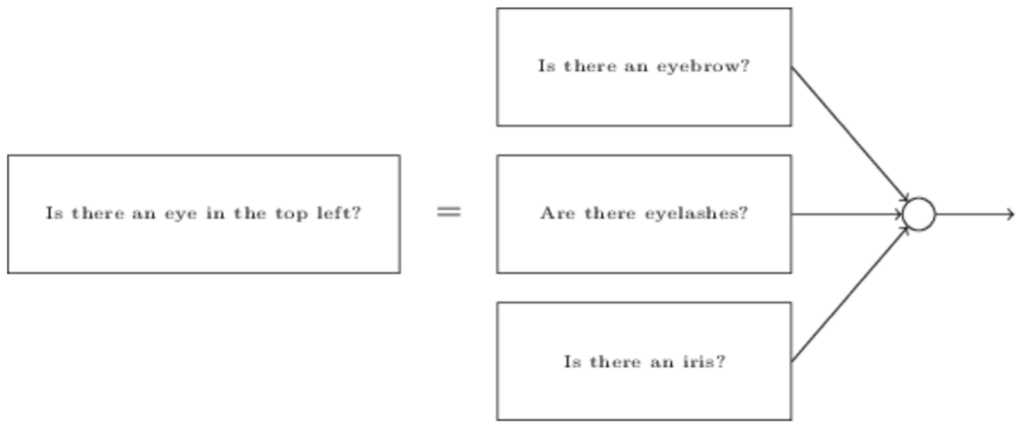
How does Deep Neural Network work?

The concept of Machine Learning using Deep Neural Network is actually not that complicated, although the mathematics behind it can be. This process is called **Supervised Learning.** Basically, you use a set of training data to recursively fine tune a mathematical model, until you are comfortable to use the model and let the computer predict the outcome of additional test data.

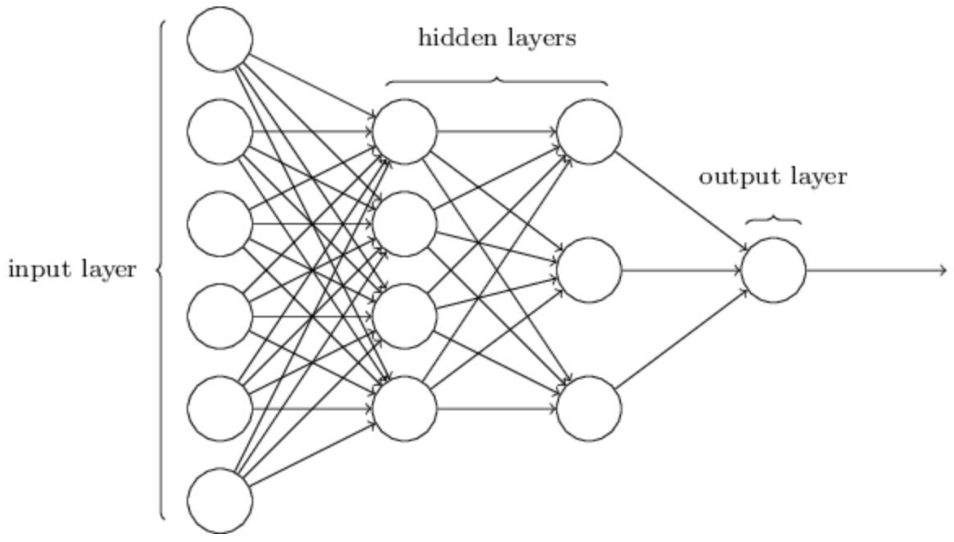
Let's consider an example. If you want to determine if a picture is the face of a human being, you can potentially break down the picture into different sections and ask the following questions:


input layer
(image pixels)
Is there an eye in the top left?
Is there an eye in the top right?
Is there a nose in the middle?
Is there a mouth at the bottom?
Is there hair on top?
Is this a face?

Obviously, these questions are not detail enough since you would easily have misinterpreted the above picture as the face of a human. Therefor, one would have to further break down each subsection and ask more detail questions.



These questions can be broken down, further and further through multiple layers. Ultimately, we'll be working with sub-sections that answer questions so simple they can easily be answered at the level of single pixels. The end result is a network which breaks down a very complicated question - does this image show a human face or not - into very simple questions answerable with the training data.

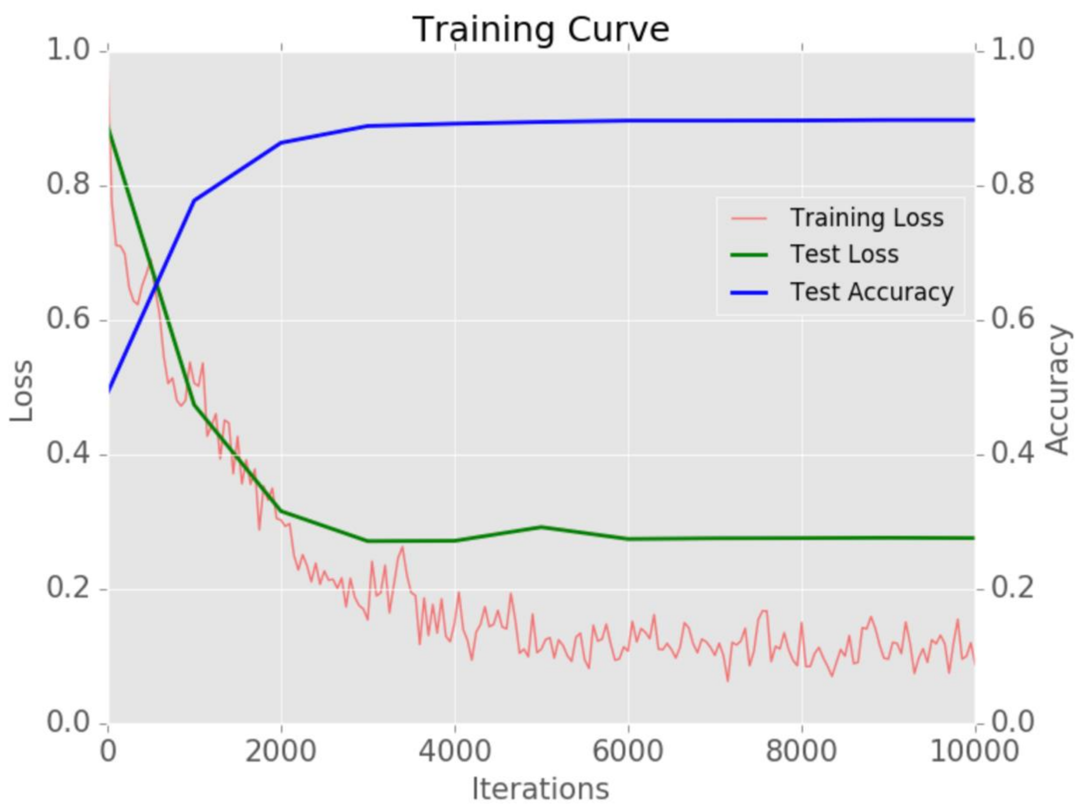


The detail mathematical theory behind DNN is beyond the scope of this article. The following is a brief explanation of how the concept works in laymen terms. The theory behind DNN is that the problem you want to train the machine to solve can be broken down into multiple finer grain problems. In terms of Machine Learning, each of these finer grain question is called a **Neuron**. Further, we can represent each of these neurons with a linear **Sigmoid function** as follows:

$$z = a_1 w_1 + \cdots + a_k w_k + \cdots + a_K w_K + b$$

Where $\mathbf{a_n}$ represents the **environment variables** you want to derive the answer to a finer grain question $\mathbf{Z}$ (represented by the Sigmoid function), and $\mathbf{W_k}$ represents the **weights** and $\mathbf{b}$ represents the **Bias**. The composite of the entire set of Sigmoid functions is call the Model you want to use to train the machine, and Machine Learning process is basically as follows:

1. Pick your initial **Model** (set of questions that compose the subject you want to derive) and the weights and bias for the Model. Each function within the model can have a different set of weights and bias.
2. Use a set of training data, go through each function in the model to derive Z and calculate the "distance" between Z and the actual value. Hence, the standard deviation of the summation of all Z's can be derived. This is also called the **Training Loss**.
3. Then you use **back propagation** to refine the training loss. Back propagation is a method used in artificial neural networks to calculate the error contribution of each neuron after the set of training data is processed. In the context of learning, back propagation commonly employs the gradient descent optimization algorithm to adjust the weight of neurons by calculating the gradient of the loss function.
4. Deep Learning means you go through this approximation process multiple iterations, each iteration is called a **hidden layer**. A recursive approximation scheme is applied to the computation of all the sigmoid functions by adjusting your weights and bias to make sure the training loss becomes smaller with each iteration (using gradient descent optimization).
5. Eventually, you arrived at "optimal" model that you will use the model to verify the outcome with additional testing data. The accuracy of the model is represented by the "distance" between the computer derive answer to the problem and the actual answer. This is also called the **Test Loss**.
6. If the accuracy is less than desired, then you will adjust the model and go back and try again. The following graph shows the relationship between Training Loss, Test Loss, and the accuracy of the model.



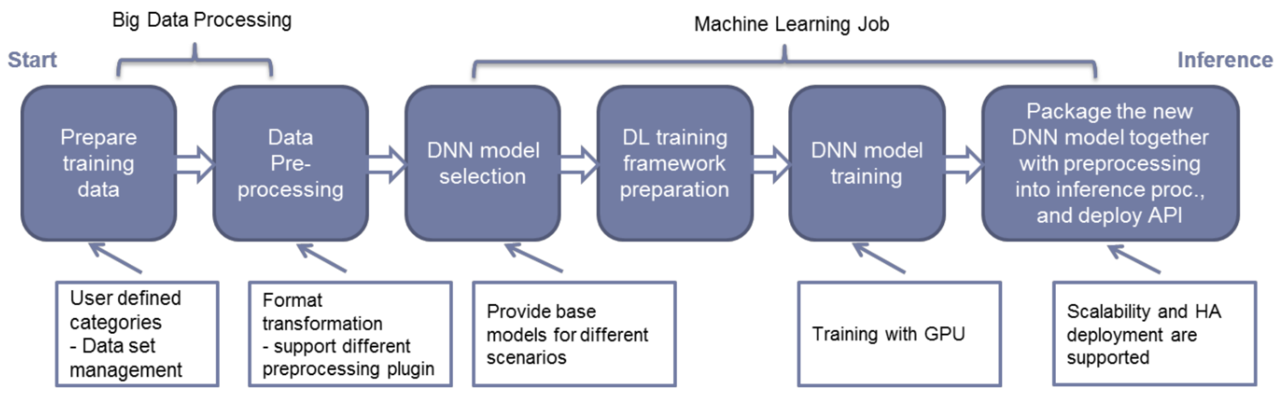
If you are getting a feeling that the Machine Learning process is very compute intensive, you are absolutely right. How do you get the set of training data? How do you pick the initial model? How do you determine the progressive values of weights and Biases? Fortunately, there are tools to help you with each step of these process. We are going to discuss that next.

Cloud Infrastructure for Artificial Intelligence

A Day in Life for an AI Engineer

It should be apparent from above that Machine Learning is a continuous process. Not only from the perspective that DNN is by its nature iterative (due to many hidden layers), but also rarely is the accuracy high enough in the first estimation of the model. In fact, even after a model is put into production for inference, the model can still be further refined and improved upon additional training data.

The following diagram shows the steps of a single iteration of a Machine Learning cycle. The process can roughly be separated into the Training Data Pre-processing phase, the Machine Learning phase, and the production (inference) phase. Each phase needs its own computing cluster that's composed of different data processing tools. We will go into each of these computing cluster later. It is important to note that this iteration will be repeated even after the DNN model is packaged and loaded up for inference processing. So it is important the computing clusters for both training data process and machine learning to be made available on-demand quickly.


Big Data Processing
Machine Learning Job
Start
Inference
Prepare training data
Data Pre-processing
DNN model selection
DL training framework preparation
DNN model training
Package the new DNN model together with preprocessing into inference proc., and deploy API
User defined categories - Data set management
Format transformation - support different preprocessing plugin
Provide base models for different scenarios
Training with GPU
Scalability and HA deployment are supported

Before going into the two computing clusters, let's examine a unique aspect of computing that is common to both types of computing. That is using GPU's and virtual containers.

GPU as a Service

Graphic Processing Units has been around over 20 years. It was originally invented to primarily address the 3D graphic rendering problem of the game industry. The strength of GPU computing is due to its vector and matrix processing capability, as well as it ability to do extremely high speed concurrent memory access. These attributes fit nicely with the computing requirement of DNN.

However, GPU's are very expensive and AI engineers only needs a lot of GPU's when they are pre-processing Training data and doing Machine Learning. It makes a lot of economic sense to be able to host GPU's in the cloud to be shared among AI Engineers.

Fortunately, this is now made feasible with the Dockers and cluster managers like Kubernetes. Dockers are basically Linux Containers with its own virtualization context. The cluster manager will determine at docker initiation time the type and numbers of GPU's needed and attach them to the dockers before dispatching them. As soon as the job is completed, the GPU's will be detached from the dockers and made available to other jobs by the Cluster Manager. So this is as if each AI Engineer has its own computing environment with its own GPU's.

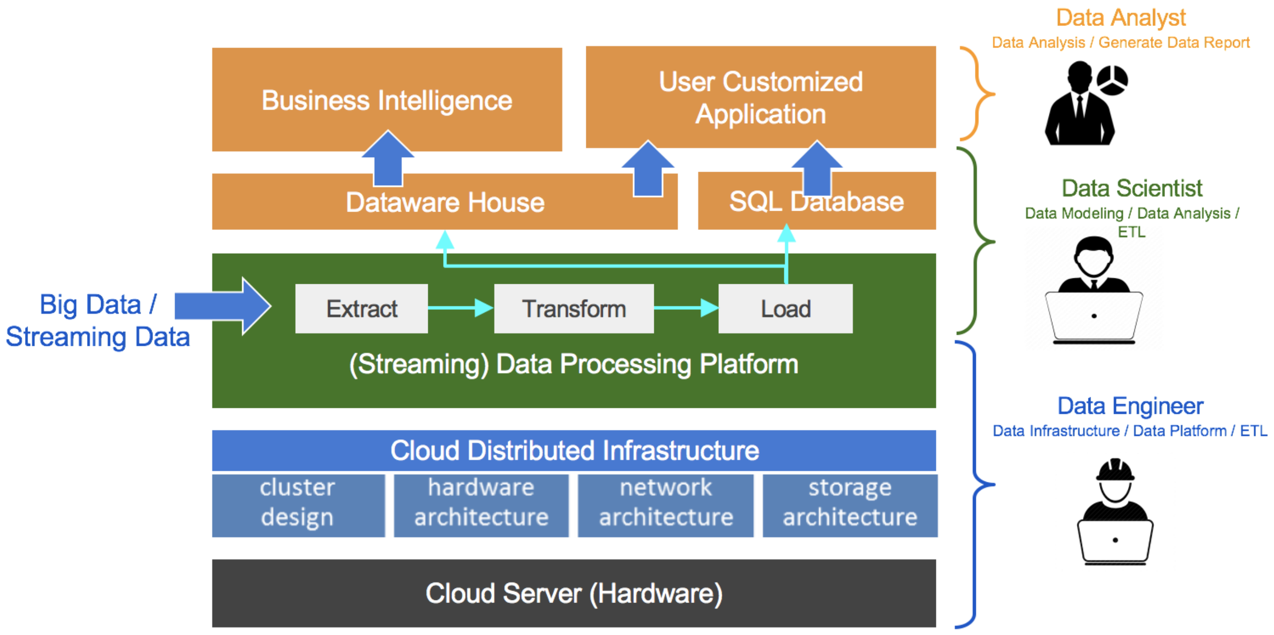
We have just defined a new paradigm of cloud computing, by offering and sharing GPU as a service among multiple computing clusters (tenants) using Dockers and Kubernetes.

Computing Infrastructure for Data Processing

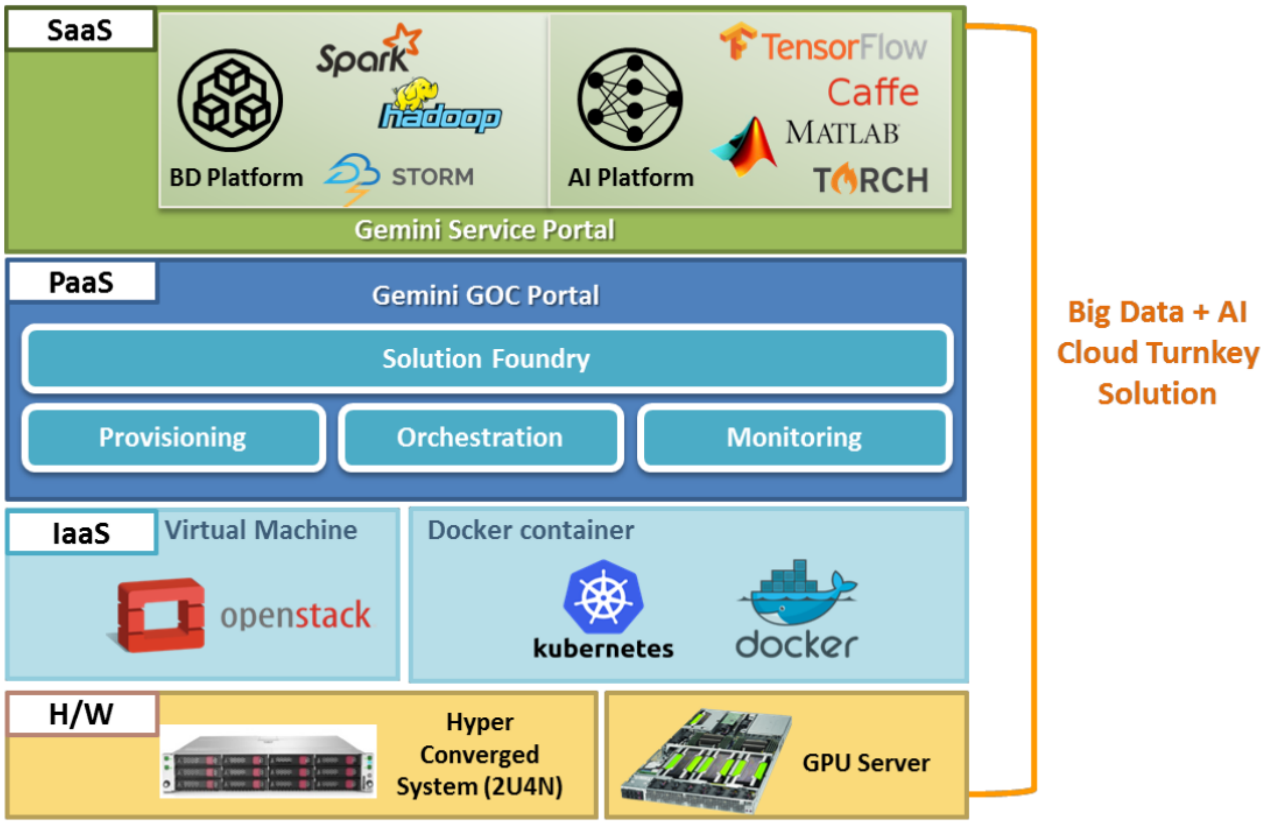
Some Machine Learning training data may need pre-processing before they can be used for the DNN process. As shown in the below diagram, there are actually 3 roles in typical big data processing:

1. A Data Engineer will build the computing cluster and load the appropriate tools. For example, Streaming data typical requires Kafka to stage the input stream, and the analysis will be done by Spark.
2. The Data Scientist is responsible for writing programs on the Jupyter IDE to Extract the relevant information from the raw data, transforming to the digital format useable by DNN, and load the data into a persistent volume (data warehouse or RDBMS) for further processing.
3. Finally, the Data Analyst is responsible for doing data slicing and aggregation to derive conclusions and generate reports

It should be noted that the process of building such computing clusters can be self-provisioned and orchestrated on demand using a Cloud OS with Big Data Service PaaS subsystem.


Data Analyst
Data Analysis / Generate Data Report
Business Intelligence
User Customized Application
Dataware House
SQL Database
Data Scientist
Data Modeling / Data Analysis / ETL
Big Data / Streaming Data
Extract
Transform
Load
(Streaming) Data Processing Platform
Data Engineer
Data Infrastructure / Data Platform / ETL
Cloud Distributed Infrastructure
cluster design
hardware architecture
network architecture
storage architecture
Cloud Server (Hardware)

Computing Infrastructure for Machine Learning



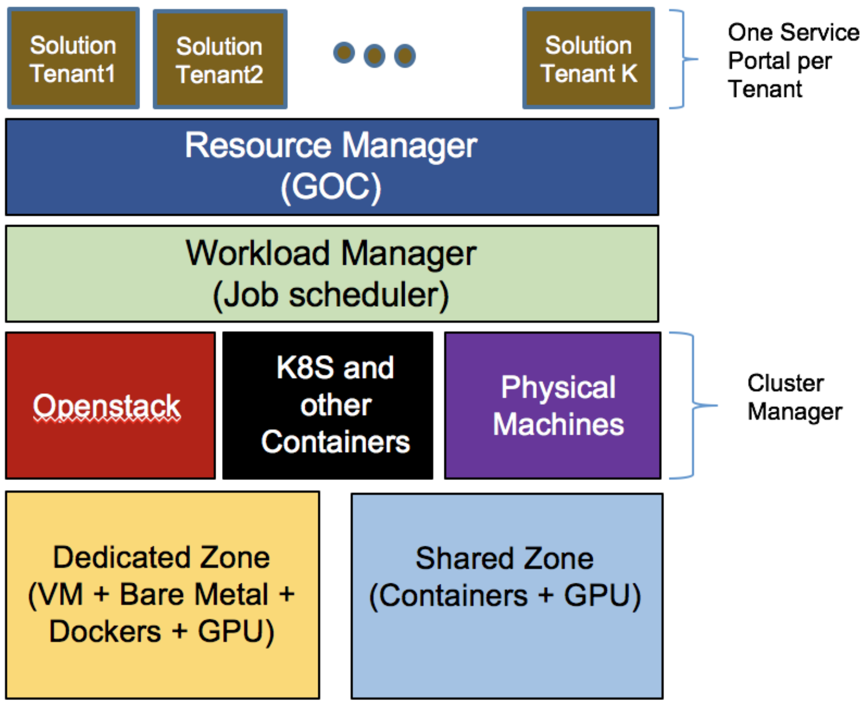
The computing cluster for Machine Learning is actually not that different from that of the training data pre-processing, with the exception that it needs to have GPU attached, and the software pre-loaded are back propagation tools such as Tensorflow, Jupyter, or Torch.

Just like the data processing cluster, an appropriate Cloud OS + Big Data PaaS subsystem can automatically provision the computing cluster for DNN machine learning. A big advantage of having both clusters on the same Cloud OS + PaaS subsystem is that the Persistent Volume loaded by the training data pre-processing cluster and be directly accessed by the Machine Learning cluster. Also, the hardware resource (CPU, Storage, Network, GPU) can be more effectively utilized since these 2 cluster are likely not running at the same time.

Service Provider Computing Platform for AI & Big data

Summarizing the above requirements and computing infrastructure, it makes sense for a service provider to host both Big Data and AI/ML on the same virtual infrastructure（AI 大資料運算及深度學習平台）. Gemini Open Cloud (GOC) has developed a platform for Service Providers to provision such computing clusters On-Demand. Built on top of a highly available Openstack Cloud OS, GOC manages multiple types of virtual resources (VM's, Dockers, bare metal machines) to provision and orchestrates computing clusters with the appropriate tools for Big Data, Machine Learning, and Inference processing. The computing clusters can be dedicated to specific users, or shared among a group of users in batch mode. The Gemini GOC platform consists of the following components:

1. A Cloud OS + PaaS layer with a service portal to allow its users to self-provision both types of computing clusters. This is shown as the **Resource Manager** in the following diagram.
2. GOC supports multiple tenants with heterogeneous virtual containers like VM, Dockers, Bare MetalServers, and GPU's.


Solution Tenant1
Solution Tenant2
Solution Tenant K
One Service Portal per Tenant
Resource Manager (GOC)
Workload Manager (Job scheduler)
Openstack
K8S and other Containers
Physical Machines
Cluster Manager
Dedicated Zone (VM + Bare Metal + Dockers + GPU)
Shared Zone (Containers + GPU)

3. Each type of virtual containers has its own respective **clustering manager**. For example, Openstack manages all the virtual machines, while Kubernetes manages the dockers.

4. A user can log into GOC and request a “virtual data center” for the exclusive use the user (e.g. data scientist). We call this the **Dedicated Zone**. Each dedicated zone will have its own **Service Portal**. GOC can create multiple dedicated zones that consist of its own isolated virtual resources and its own service portal.
5. GOC also comes with a **Workload Manager** that allows uses to submit batch jobs to a **Shared Zone** without its own dedicated VDC console. The workload manager will schedule the job to run when all its required resources becomes available. Otherwise, the job is queues
6. The workload manager actually supported multiple queues. Each queue can be associated with its set of **resource pools**. For example, a service provider can define 2 queues in the share zone. One zone is for Servers with GPU’s, while the other queue is associated with servers that has no GPU’s .
7. Finally, GOC PaaS layer has a subsystem that allows the service provider to use templates to define different type of cloud services that can be provisioned on this platform. We call this subsystem ‘**Solution Foundry’**.

For the dedicated zone, the business model is typically rent by capacity. Users wants to have their own service portal where they can define their own jobs and monitor their virtual work environment

On the other hand, the shared batch environment is rented based on utilization. The jobs are queued by the workload manager until the requested resources are available to execute the job.

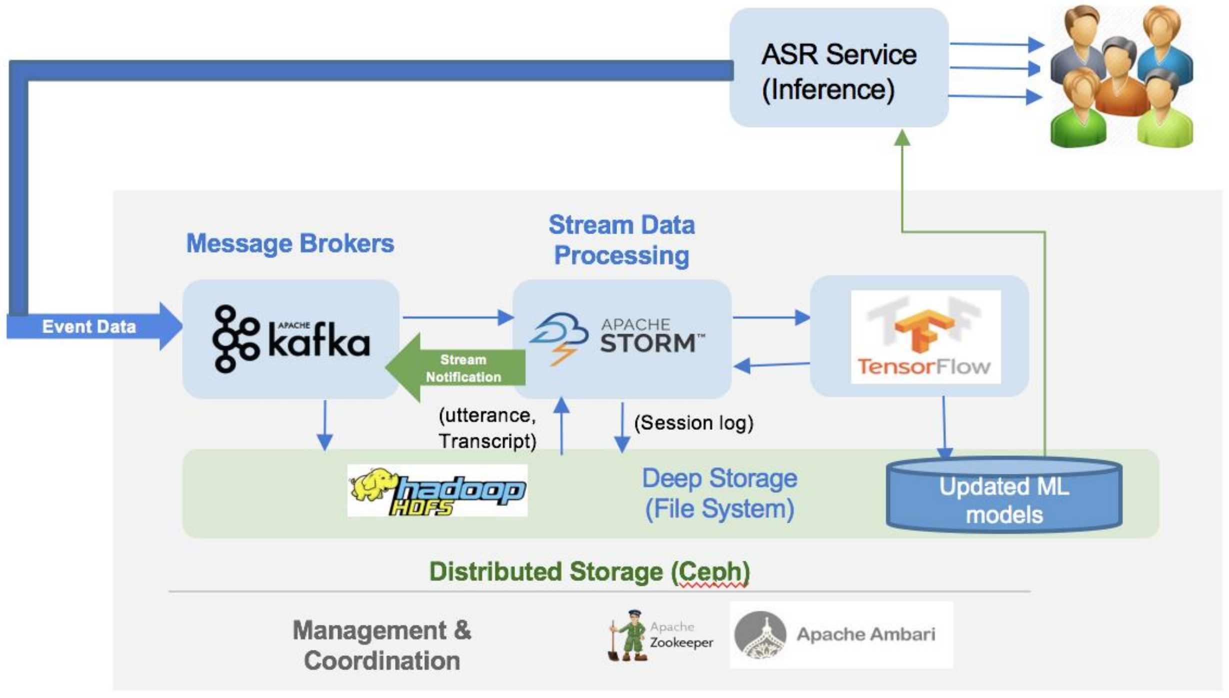
A Real Use Case Scenario

Let’s wrap up with a real-life use case scenario using the above Big Data + AI/Machine Learning Platform. The example here is about an Automatic Speech Recognition (ASR) Service. The ASR engine already is actually an inference engine using a DNN model derived from previous training data. But the accuracy of the speech recognition has room to improve. Here’s how the Engineer uses Big Data Computing Cluster and an DNN Machine Learning Cluster to continuously improve the accuracy of ASR:

- The ASR service engine records multiple simultaneous conversations streams between the Customers and ASR.
- These exchanges are forwarded to a Big Data Cluster using the Kafka system to stage the conversation streams.
- The Kafka system has multiple threads (channels), each channel collects a particular log stream of the speech service.

- Kafka channels are followed by Storm jobs for online data processing. For example, a Storm job joins the utterance log stream and transcription stream for each conversation.
- The joined result, namely session log stream, is fed to another Machine Learning cluster running Tensorflow application
- The Machine Learning uses the session log stream as training data to improve the ASR model. The Trainer writes the updated model to Ceph.
- Researchers might change the training algorithm, and run some experiment training jobs, which serve as testing to ASR service jobs
- Once the researchers determines the updated model has better accuracy, it will upload the updated model to the ASR inference engine.

The continuous refinement process is represented by the figure below:



Machine Learning ASR service

Summary

As we have shown above, the DNN supervised training methodology holds very promising approach to solve many difficult problems because it is not developing an algorithm, but rather to train the computer to develop a model to predict the outcome. Because DNN is an iterative approach using a lot of Matrix and Vector computations, it will benefit tremendously from the benefits of GPUs. However, it only makes economic sense for AI engineers to be able to easily access a hosted environment of many powerful GPU's.

A Cloud Service Provider can provide this shared hosted virtual environment by using a Workload Manager to allow AI engineers to submit job to a cluster of dockers with GPU's attached as virtual devices. Machine Learning jobs are submitted to the workload manager on job queues. The workload manager will determine if there are sufficient computing and GPU resources to dispatch the job. Once the job is completed, the pool of GPU's and computing resources will become available for the next job.

Finally, Machine learning is rarely a standalone job. Often times, the preparation of training data will require pre-processing by a Big Data cluster hosting Hadoop tools such as Map-Reduce, Kafka, Spark, and Storm. Because of the virtualized cloud environment, it is extremely easy and efficient for the Cloud service provider to host both Big Data Computing Clusters and Machine Learning Computing Clusters using the same hardware in his IDC. The cloud environment also allows the output of the Big Data Computing Cluster to be fed into the Machine Learning Computing Clusters without copying the data. This increases the efficiency and security concerns for the AI Engineers.

Gemini GOC is a product that allows a Service Provider to quickly set up a Private Cloud environment to host both Big Data and Machine Learning computing cluster. GOC consists of the Cloud OS, the clustering manager for different virtual resources, and a Resource Manager that provisions the computing clusters and orchestrate the cloud service (applications) running in them. Finally, GOC also comes with a workload manager that allows the service provider to set up a shared zone to allow user to submit batch job to utilize different resource pools.