|  |  |
| --- | --- |
|  |  |
| 1 | Deep Learning Workspace |
|  |  |

**Abstract**

*This document discusses Deep Learning workspace (DL workspace), an AI infrastructure effort that extends beyond DNN training, and aim to boost the productivity of AI scientists (both 1st party and 3rd party) in all lifecycles of their work. Our key observation is that AI scientist’s daily work extends far beyond neural training, but includes data exploration, DL inferencing, writing codes/script for DNN models, etc.. DL workspace is built to provide a coherent set of workspaces that is source controlled and linked together to provide all functionality and computation resource to support AI scientist’s daily work. Each DL workspace contains a specific DL task, which can be 1) interactive exploration, 2) DNN model training, 3) DNN evaluation, and 4) DNN deployment, etc.. The DL workspaces can be executed in: 1) batch mode, 2) background mode, and 3) interactive mode. DL workspace intelligently allocate proper computation (GPGPU) resource to suite all computation need of the AI scientists. DL workspace shields the AI scientist (either 1st party or 3rd party) from mundane system setup (e.g., building DNN training Docker, version control, using python notebook, allocating Docker to proper computing resource) so that they can concentrate their effort to push AI to the new frontier.*

**Revisions**

|  |  |  |
| --- | --- | --- |
| **Date** | **Summary** | **Author/Editor** |
| 2016/11/15 | First Draft | Jin Li, Hongzhi Li, Sanjeev Mehrotra, Sudipta Sengupta,  Kenneth Tran |

**Table of Contents**

[1. Introduction 2](#_Toc467587517)

[2. Scenario 2](#_Toc467587518)

[2.1. Interactive exploration 3](#_Toc467587519)

[2.2. Data curation 3](#_Toc467587520)

[2.3. DNN Training 3](#_Toc467587521)

[2.4. DNN Inference. 4](#_Toc467587522)

[2.5. Differential Neural Computing. 4](#_Toc467587523)

[3. DL workspace: architecture 4](#_Toc467587524)

[3.1. Docker 4](#_Toc467587525)

[3.2. Version control: Git, Docker, blob store. 4](#_Toc467587526)

[3.3. DNN training framework. 5](#_Toc467587527)

[3.4. Big data processing. 5](#_Toc467587528)

[3.5. Container orchestration. 5](#_Toc467587529)

# Introduction

*Deep neural network (DNN) learning driven by big data has become the state-of-the-art system for image/speech recognition, sequence learning and reinforcement learning. Moreover, recently proposed Differentiable Neural Computing (DNC) further introduced external content/location addressable memory to DNN, which allows DNN to manipulate complex data structures. More and more machine learning researchers are working actively in this exciting field, and are pushing the boundary of Artificial Intelligence every day.*

*We observe that the daily work of DNN scientists in the deep learning field involves much more than training large DNN models. Typically, the scientist spent a majority of his/her working hours in exploring data sets (e.g., curating data source [web/social network crawl] to generate the training/testing/validation set), fine tuning DNN models (via python notebook, writing codes for new DNN models and writing scripts for training). He/she will then develop a set of DNN models, train on relative small data sets, monitoring and observing the training results, and revising models based on the results of the training experiment. Large scale training on big dataset (which may require multiple GPUs, and/or even multiple machines worth of GPUs) is usually reserved to a few selected task when the model is proven mature for production.*

*We believe that AI Infrastructure need should extend beyond DNN training and be designed to allow DNN scientists (both 1st party and 3rd party) to manage their workspace flexibly to maximize their productivity through all lifecycles of their work, not just training alone. As such, we propose DL workspace (deep learning workspace). Each DL workspace contains a specific DL task, which can be 1) interactive exploration, 2) DNN model training, 3) DNN evaluation, and 4) DNN deployment, etc.. The DL workspaces can be executed in: 1) batch mode, 2) service mode, and 3) interactive mode. A DNN model training task is typically executed in batch mode. The background training task usually take days or even weeks to finish. As such, if system resource is tight, the DNN training task can simply be evicted to leave the system resource to other tasks with more urgent and/or interactive demand. DNN inferencing is typically executed in the service mode. In service mode, the number of computation resource needed by the DNN inference task may need to be automatically scaled according to the current workload. A third type of DL workspace is the interactive workspace. This type of workspace is normally used by DL scientist during daily work, in which they will explore data/DNN model interactively, e.g., via python notebook on a particular DL platform (e.g., Tensorflow, CNTK, Caffe). The AI infrastructure should usually strive to meet the resource requirement of the interactive DL workspace first, the service workspace second, and use the rest of the resource for background training tasks.*

# Scenario

Let’s discuss the daily life of Jason, a fictional user of the DL workspace. Jason could be either a first party developer (e.g., a data scientist in Bing), or a third party user of Azure. The DL workspace allows Jason to juggle among many of his daily tasks, and provide appropriate computing resource and environment to satisfy all his computing need.

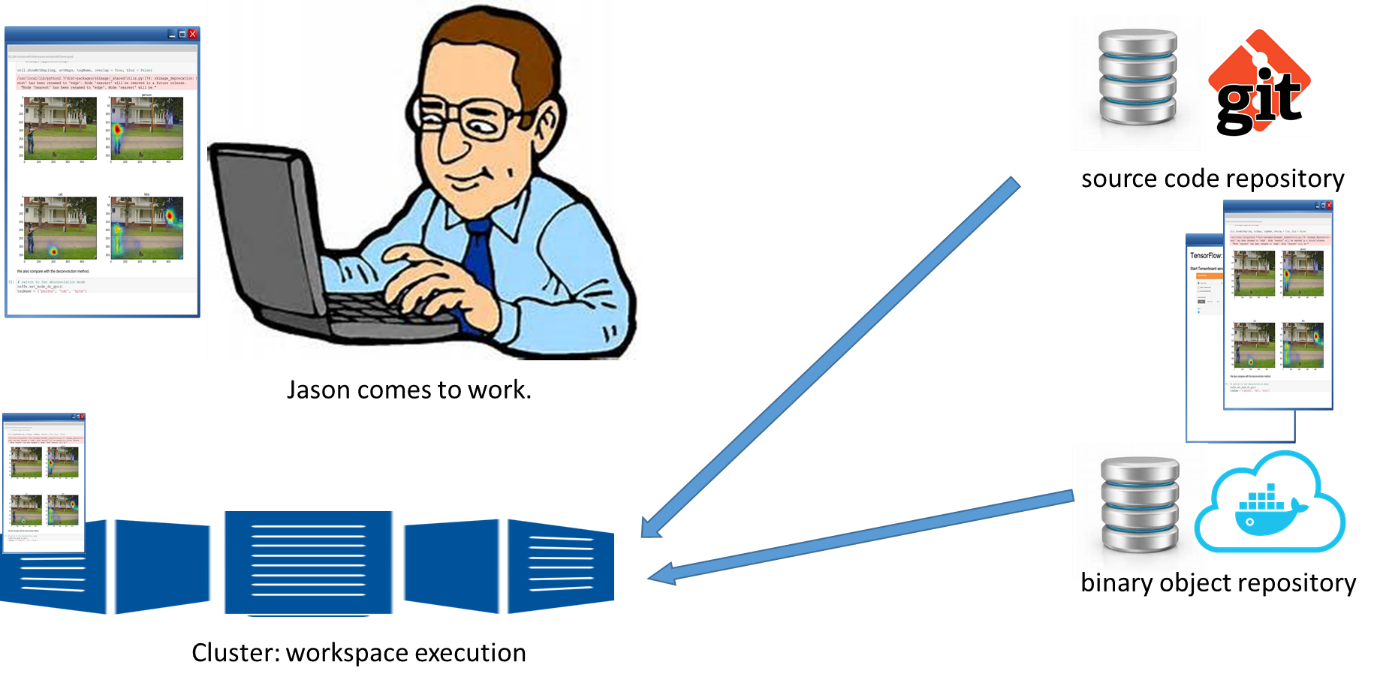


Figure AI Infrastrcture: Deep learning workspace.

## Interactive exploration

Jason came to work in the morning. He logged on to the organization’s Azure account, and retrieved an exploration workspace that he had worked on last night. The DL workspace dutifully obliged to Jason’ request, paused a low priority training workspace, checked out the workspace that he has saved (with appropriate version of data and binary program), and execute the workspace in a container that runs on the computing resource that is reserved by the organization’s Azure account. Jason had been using python notebook to explore a new DNN network model so that it could more reliably localize and recognize the video objects. The python notebook allowed him to quickly tune parameter of the DNN network, and instantly visualize the result of his work (e.g., showing the sparsity of the DNN network and its intermediate feature vectors).

When Jason left for lunch and/or left for home, he shut down the interactive workspace. The content of the workspace is archived for later rehydration, while DL workspace reclaimed the computing resource occupied so that it can be used for other task (e.g., DNN training).

## Data curation

Using a second python notebook, Jason was also curating new data that would feed into the DNN network model that he has designed. He wrote a Spark job that processed hundreds of millions of weakly labeled image, and cluster those images among certain labels. When the job is finished, he examined the clusters, eyeballing on the perspective labels, accepted and rejected certain clusters and their labels. The resultant labelled data was stored, and to be assigned for training/testing/validation.

## DNN Training

Jason has queued a number of DNN training experiments. Each training experiment came with specific DNN software (Tensorflow, Caffe, CNTK, etc.), DNN models used, and the training, testing and validation dataset. The AI infrastructure offered source code version control for all text contents (e.g., DNN models, training script, source code of DNN training software). Additionally, it also offered binary version control for other experimental components (e.g., container of the training experiment, data used, generated model, etc..) The version control system allowed Jason’s colleague to quickly repeat and/or follow any of Jason’s experiment.

Jason quickly examined the progress of the DNN training experiments through its associated dash board. Based on the result, Jason made adjustments, canceled certain training experiments that were not running well, and quickly forked a failed experiment, tuned a few of its parameter/code/data, and relaunched a new experiment. Jason also adjusted the token assigned to training workspace, thereby adjust the priority of the DNN training experiment that would be run in the system when resource became available.

## DNN Inference.

Jason noticed that certain DNN training experiments have finished. The output DNN model was managed by DL workspace and was version controlled. Jason examined the experimental results, and were satisfied with the performance of the trained DNN model. He decided to push out this new trained model for inference service. The DL workspace scaled down workspace that are running the old trained model, and started and scaled up workspace that are running the new trained model. The DL workspace also monitor the load (e.g., work queue and response time) of the inference workspace, and launch extra instance of the workspace if the load on the inference service increases. Jason monitored the dash board that tracked the inference service, and was relieved that the transition progressed smoothly.

## Differential Neural Computing.

Jason were also following the new development in differential neural computing. He was developing code that combined the DNN network with a data analytical tool, such as Spark/Prajna. He reserved a few customized interactive workspaces so that he could toy with the new computation model.

# DL workspace: architecture

We believe that the DL workspace can be quickly built by leveraging existing components. As times goes up, we could investigate on various optimization technologies, e.g., optimization of Docker deployment and blob storage (via P2P & deduplication).

## Docker

Each DL workspace will be implemented as a separate Docker. Docker container wraps up a piece of software in a complete file system that contains everything the software needs to run: code, runtime, system tools, libraries, etc.. It is a tool that can package an application and its dependencies in a virtual container that can run on any linux server, and enable flexibility and portability on where the application can run, whether on premises, public cloud, private cloud, bare metal, etc.. It avoids the overhead of starting and maintaining virtual machines, and greatly simplifies the creation and operation of tasks or workload queues and other distributed systems.

## Version control: Git, Docker, blob store.

There are strong desires to track all AI experiments. DL workspace will apply version control on all components of the workspace, including source code, DNN model files, scripts, training/testing/validation data, generated DNN models, and workspace runtime. It will use Git to manage text component of DL workspace (e.g., source code, Docker file, DNN model files, scripts, configuration, etc..) A private Docker registry will be used to manage the Docker container. Other binary data (e.g., training/testing/validation data) will be version controlled through a blob storage system, such as Azure.

## DNN training framework.

DL workspace can support multiple DNN framework, such as TensorFlow, Caffe, CNTK, etc.. With container (Docker) technology, the DL workspace will operate largely independent of the underlying DNN framework. As most DNN framework today support python notebook, it became a common interface that allow the DNN framework to interact with the DL workspace in a unified way. DL workspace can also leverage other DNN UI feature, e.g., TensorBoard, Caffe DNN network visualizer.

## Big data processing.

DL workspace will also allow other big data processing tool, e.g., Spark/Prajna, to be used in conjuncture with DNN models for both data curation and new DNN computation mode, e.g., the differential neural computing (DNC).

## Container orchestration.

Container orchestration tool, such as Kubernetes or Docker Swarm, provides enterprise-level framework for integrating and managing containers at scale. They will be used as a start point to schedule DL workspace for execution.