HITACHI
Inspire the Next

ARCHITECTURE GUIDE

Deep Learning Platform with Pentaho DS225 for Accelerated Deep Learning

Reference Architecture Guide

By Ken Wood and David Pascuzzi

July 2019

Feedback

Hitachi Vantara welcomes your feedback. Please share your thoughts by sending an email message to SolutionLab@HitachiVantara.com. To assist the routing of this message, use the paper number in the subject and the title of this white paper in the text.

Revision History

Revision	Changes	Date
MK-SL-157-00	Initial release	July 8, 2019

Table of Contents

Deep Learning Platform with Pentaho DS225 for Accelerated Deep Learning

Reference Architecture Guide

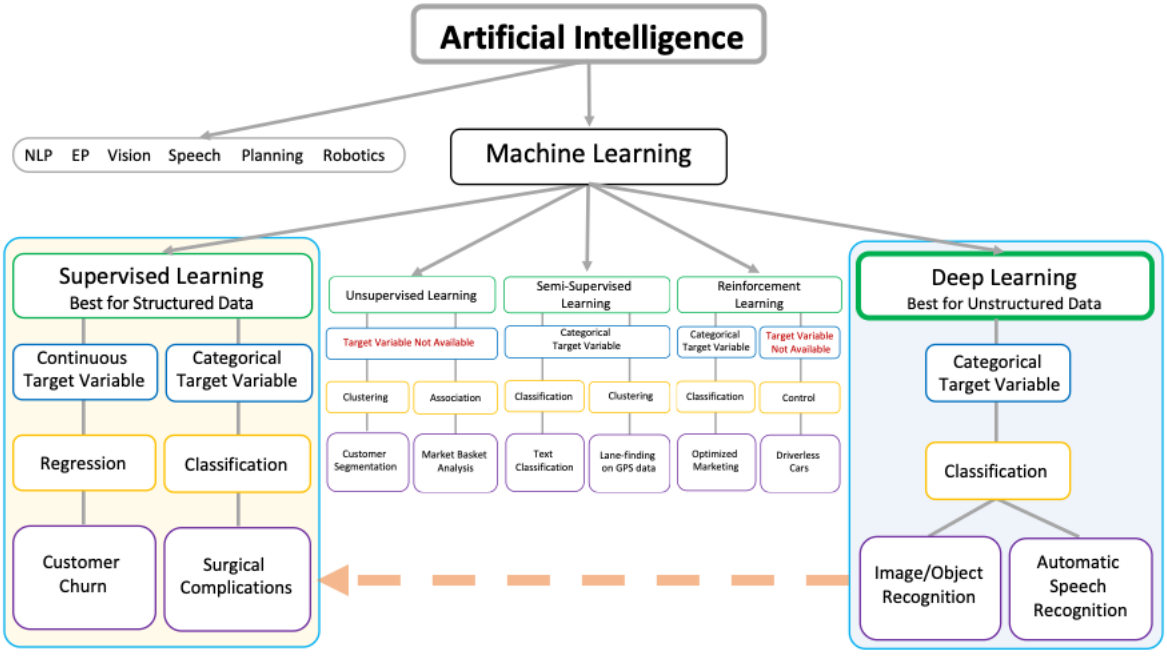
The field of Artificial Intelligence (AI) has many subdomains that play important roles in providing intelligence to machines that constantly strive to mimic or, in some sense, surpass human intelligence. While the combination(s) of these subdomains can lead to varying forms of simulated human intelligence, focusing on a single subdomain can create specialized intelligent solutions that can match or exceed human intelligence (and tenacity) in mundane and repetitive tasks. Machine learning (ML) and Deep Learning (DL) are two growing subdomains of AI that are gaining considerable attention in enterprise and consumer applications.

While ML has a robust reputation in the structured data analysis industry, DL is beginning to be viewed as a way to solve many new and interesting use cases. DL is a specialized subdomain in that it is used both in the AI subdomain of Computer Vision and is part of the ML subdomain. The inclusion of DL in the ML subdomain is a recent trend where DL can be treated and used similarly to ML, but for specialized use cases and datasets. This brings up a useful distinction between ML and DL based on the current understanding and best practices,

- Machine Learning is typically used and best suited in structured data analysis for classification and regression analysis.
- Deep Learning is typically used and best suited for unstructured data analysis for classification.

While this is not a hard and fast rule, it is currently considered a best practice that can yield the best results. Both ML and DL can be used to solve questions about unstructured and structured data, respectively. However, the results are typically less than optimal. This may change in the future as the state-of-the-art in specialized hardware configurations and economies of scale attempt to shift the overall “best practices” understanding.

Figure 1



This white paper describes the configuration and methodology of performing DL processing on a complex dataset using easy to use software tools and specialized hardware configurations designed to show the advantages in processing speeds and increased productivity of this type of compute intensive task.

Hitachi's solution is based upon the following key components:

- Hitachi Advanced Server DS225 – A compact and optimized 2U accelerator server, that delivers the compute, memory and storage needed for advanced analytics, artificial intelligence, and deep learning applications.
- Nvidia Tesla V100 - the most advanced data center GPU built to accelerate AI, High Performance Computing (HPC), and graphics.
- Cisco Switches – Industry leading switches.
- Deep Learning Frameworks – A library, tool, or environment that simplifies the task of developing deep learning related applications.
- Pentaho - Analytics platform that offers both data integration and visualizing in a scalable offering via the use of GUI based tools.
- Plugin Machine Intelligence – An experimental Machine Learning framework for Pentaho to do Machine learning and deep learning natively within the visual integrated development environment.

Advantages of Hitachi's solution:

- Provides a solution that can be run in a datacenter. Unlike solutions running on desktop GPUs. the DS225 with Nvidia Tesla V100 GPUs can run in a datacenter
- Resource sharing – with a datacenter solution, the server(s) can be shared between multiple data scientists
- Performance – GPU based systems can reduce training times from weeks to hours
- Multi-Tasking – when training you can assign a subset of GPUs to a process, allowing other training tasks to use the other GPUs
- Availability – redundant power supplies, RAID controllers, and other components help to ensure the availability of the solution
- Scalability – growing from one to four GPUs scales the performance of the system
- Integration with Pentaho
- Continued innovation from Hitachi Vantara Labs in the field of AI and other leading-edge technologies
- Backed by Hitachi

Note — Testing of this configuration was in a lab environment. Many things affect performance beyond prediction or duplication in a lab environment. These results were for the ResNet-50 benchmark. This benchmark was designed by team of data scientists and programmers to take full advantage of multiple GPUs, so it is possible that your machine learning code will not see the same level of performance improvements.

Note — The Plugin Machine Intelligence capability is an experimental prototype provided by Hitachi Vantara Labs. This Pentaho Data Integration plugin is freely available to download, use and test, but support is only provided by the Pentaho community and Hitachi Vantara Labs.

Introduction

Hitachi Advanced Server DS225

Hitachi Advanced Server DS225, shown in Figure 2, delivers unparalleled compute density and efficiency to meet the needs of your most demanding high-performance applications in the data center. DS225 takes full advantage of the ground-breaking Intel Xeon Scalable Processor family. By combining the Intel processors with up to four dual-width 300W graphic accelerator cards and up to 3 TB memory capacity in a 2U rack space package, this server stands ready to address the most challenging demands on today's IT infrastructure.

The DS225 server provides the reliability, availability and serviceability features demanded by your business-critical enterprise applications. The server's modular design simplifies cable routing and reduces service time. Redundant hotswap drives and power supplies provide a resilient architecture for important applications.

Figure 2

NVIDIA Tesla V100 Tensor Core Graphical Processing Unit

NVIDIA Tesla V100 Tensor Core, shown in Figure 3, is the most advanced data center GPU built to accelerate AI, High Performance Computing (HPC), and graphics. It is powered by the NVIDIA Volta architecture, comes in 16 and 32 GB configurations, and offers the performance of up to 100 CPUs in a single GPU. Data scientists, researchers, and engineers can now spend less time optimizing memory usage and more time designing the next AI breakthrough. Nvidia Tesla GPUs are a key component in the next generation of computing, including:

- Deep learning training
- Deep learning inference
- Unstructured data preprocessing
- Virtual desktop infrastructure
- High-performance computing

Figure 3

Pentaho

Pentaho Data Integration (PDI) allows you to ingest, blend, cleanse, and prepare diverse data from many sources. With visual tools to eliminate coding and complexity, Pentaho puts all data sources and the best quality data at the fingertips of businesses and IT users.

Using intuitive drag-and-drop data integration coupled with data agnostic connectivity, your use of Pentaho Data Integration can span from flat files and RDBMS to Hadoop to Spark, Internet of Things (IoT), and beyond. Go beyond a standard extract-transform-load (ETL) designer to scalable and flexible management for end-to-end data flows. PDI can be run on-premise and in the cloud, offers a native execution engine, and the ability to run PDI in a Spark cluster using an AEL-Spark engine.

Plugin Machine Intelligence

The Plugin Machine Intelligence (PMI) plugin for Pentaho Data Integration (PDI), from Hitachi Vantara Labs, brings advanced Machine Learning capabilities to the ETL (Extract, Transform and Load) tool in the form of native PDI steps, including Deep Learning. PMI is a plugin of plugins and a framework of frameworks.

PMI provides the ability to execute Machine Learning algorithms, including Deep Learning neural networks, that can be designed seamlessly into your PDI analytic flow transformations and Pentaho based solutions.

PMI includes the following Machine Learning steps:

- A no coding approach to Machine Learning and Deep Learning.
- Make ML and DL easier to use and deploy by combining it with our data integration tool as a suite of easy to consume steps, and ensuring these steps guide the Data Engineer through its usage.
- Combine ML, DL, and data integration together in one tool/platform. This powerful coupling between Machine Learning and data integration allows the Pentaho steps to receive row data as seamlessly as other steps in PDI.
- Execute DL algorithms that require high-end processing with GPUs.

PMI is an experimental plugin extension to the PDI application and is freely available for automatic download and installation from the Pentaho Marketplace.

Table 1 lists the ML and DL algorithms and execution “engines” included in PMI version 1.4.

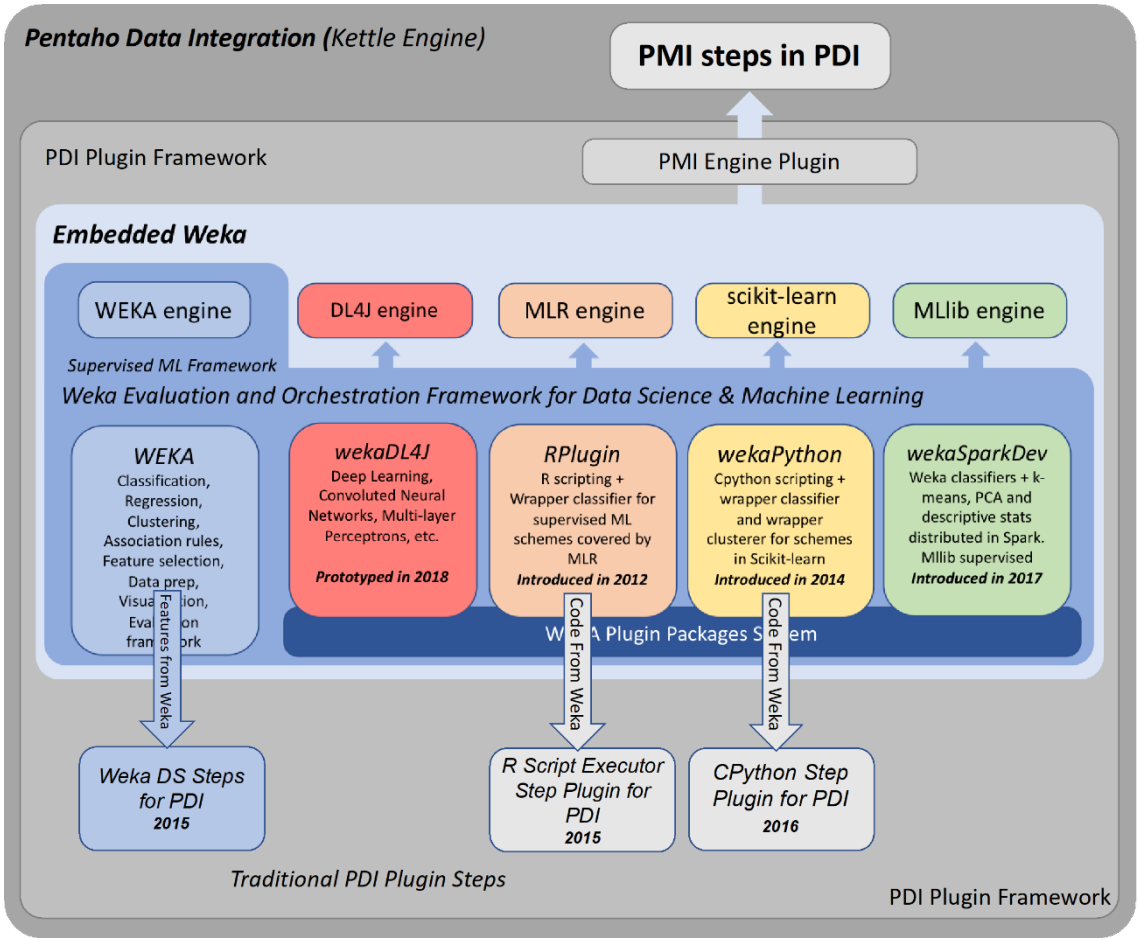
TABLE 1. EXECUTION ENGINE TO ALGORITHM IMPLEMENTATION IN PMI

Machine Learning Algorithm	Machine Learning Library Integrated in PMI 1.4				
	Python Sclkit-Learn	RMLR	Spark MLib	Weka	DL4j
Decision Tree Classifier	O	O	O	O	X
Decision Tree Regressor	O	O	O	O	X
Deep Learning Network	X	X	X	X	O
Gradient Boosted Trees	O	O	O	O	X
Linear Regression	O	O	O	O	O
Logistic Regression	O	O	O	O	O
Multi-Layer Preceptron Classifier	O	O	X	O	X
Multi-Layer Preceptron Regressor	O	O	X	O	X
Naïve Bayes	O	O	O	O	X
Naïve Bayes Multinomial	O	X	O	O	X
Naïve Bayes Incremental	X	X	X	O	X
Random Forest Classifier	O	O	O	O	X
Random Forest Regressor	O	O	O	O	X
Support Vector Classifier	O	O	O	O	O
Support Vector Regressor	O	O	O	O	X

O = Available, X = Not Available

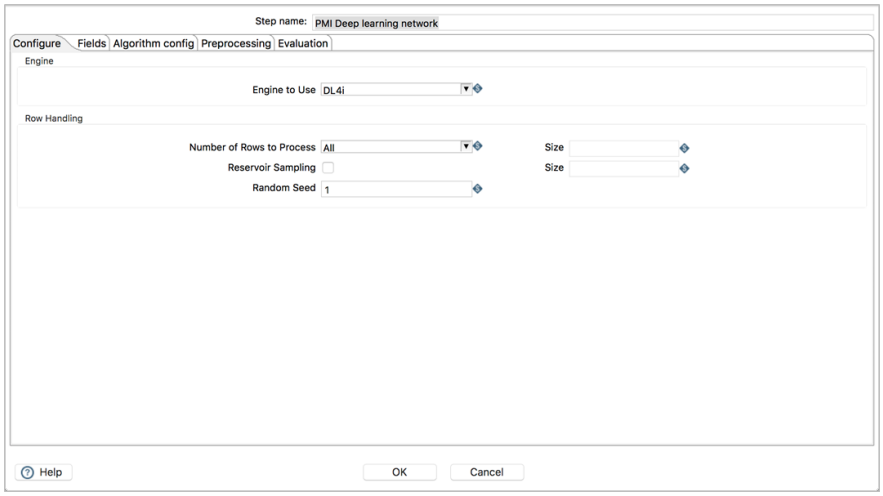
Additional and new ML algorithms can be added to the PMI framework beyond what is included with the standard Pentaho Marketplace version, and additional frameworks for different types of Machine Learning schemes can be added. PMI is designed to be extensible. Today, only Supervised Machine Learning is supported. Figure 4 shows how PMI plugs into the PDI framework.

Figure 4



As part of the Supervised Machine Learning framework, PMI supports DL through the inclusion of the Deep Learning for Java (DL4j) library. Deep Learning is a Supervised Machine Learning scheme focused on image processing and classification. Specifically, the graphical user interface for configuring and tuning the DL algorithm in PDI with PMI is shown in Figure 5.

Figure 5



Deep Learning

The following are the main components of DL dataset and model management processing:

- Data analyses, preparation and cleaning – Data is cleaned and analyzed and enhanced to identify the results before it is used in machine learning. Often you will have three sets of data: training data, validation data, and testing data.
- Model Management
  - Model development – A mathematical model is developed to approximate the data.
  - Model training – During model training the training and validation data is processed against the model. This teaches the model how to behave with real data. This is an iterative process and the model may require modification via adjusting of hyper parameters, adding or removing layers, or changing the algorithm used. It will then be retrained, and this process repeated until it is accurate enough for your purposes.
  - Inference – Inference is the task of getting an answer with live data from the deep learning model.

A deep learning framework is used to simplify this process. PMI and DL4j is used as the DL framework for this project. With a framework, you no longer need an advanced degree in mathematics and years of programing experience to develop a DL application.

This white paper is focused on PDI, PMI and DL4j as the DL framework, but other DL frameworks can be deployed with the DS225 configuration. Some of the popular DL frameworks that are GPU accelerated are:

- TensorFlow is an open source software library for high performance numerical computation. Its flexible architecture allows easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices. Originally developed by researchers and engineers from the Google Brain team within Google's AI organization, it comes with strong support for machine learning, deep learning, and the flexible numerical computation core is used across many other scientific domains.
- PyTorch is an open source machine learning library for Python, based on Torch, used for applications such as natural language processing. It is primarily developed by Facebook's artificial-intelligence research group.

These can be also be included in the PMI DL framework but were not integrated with PMI at the time of the release of this white paper.

Operating System

This solution supports running on the common operating systems associated with machine learning:

- Ubuntu

  Ubuntu is an open source software operating system that runs from the desktop, to the cloud, to all your internet connected things.

- Red Hat Enterprise Linux (RHEL)

  Using the stability and flexibility of Red Hat Enterprise Linux, reallocate your resources towards meeting the next challenges instead of maintaining the status quo. Deliver meaningful business results by providing exceptional reliability on military-grade security. Use Enterprise Linux to tailor your infrastructure as markets shift and technologies evolve.

- Microsoft® Windows Server® 2016

  Microsoft Windows Server is a multi-purpose server that increases the reliability and flexibility of your server or private cloud infrastructure.

Solution Details

Hitachi Advanced Server DS225

This solution supports multiple options to meet your needs. Table 2 shows the options used. Contact Hitachi Vantara Sales for a complete list of options. This solution is only supported on 3-Phase power.

TABLE 2. HITACHI ADVANCED SERVER DS225 CONFIGURATION

Component	Description
Advanced Server DS225	■ 1 × DS225 Chassis
Power Supply	■ 2 × 2200 watt 3-Phase
CPU	■ 2 × Intel 6154 Gold (18C 3.0GHZ, 200w)
Memory	■ 512 GB – 8 × 64 GB DDR4 R-DIMM 266MHz
Operating System Devices	■ 1 × 960 GB SATA 6 Gbps 1DWDP SFF SSD, S4500
Data Devices	■ 3 × 3.4 TB SSDs for data
Network Card	■ Mellanox ConnectX-4 LX EN Dual Port 10/25OCP Mezzanine GB Ethernet adapter ■ Mellanox ConnectX-5 Dual Port 100GB PCIe Ethernet Adaptor

When sizing memory it is recommended to have memory to be at least as large as the total memory for all graphic cards. Choosing a larger system memory size can have a significant impact on overall performance.

When performing ML/DL the same set of data is run through the learning code multiple times. If there is enough memory to allow the file system to cache the data there is minor impact to performance no matter how the files are accessed: local, remote, 25 GbE connection, or 100 GbE connection.

Storage is determined on individual needs. Often the operating system disks will provide enough storage to hold all the data used in the ML/DL system.

Management Server

To allow for management of the hardware and access to the management network an optional Hitachi Advanced Server DS120 Management Server may be included in the solution.

Networking

This solution supports having three networks as shown in Figure 6. The networks are:

- GPU Network – a 100 Gb network used for communication between the nodes when doing multi node training.
- OS Network –a 25 Gb network used for standard OS access by the data scientists, accessing the staging area, accessing corporate data, and all non-gpu communication.
- Management Network – a 1 Gb network used for out of band management.

Figure 6

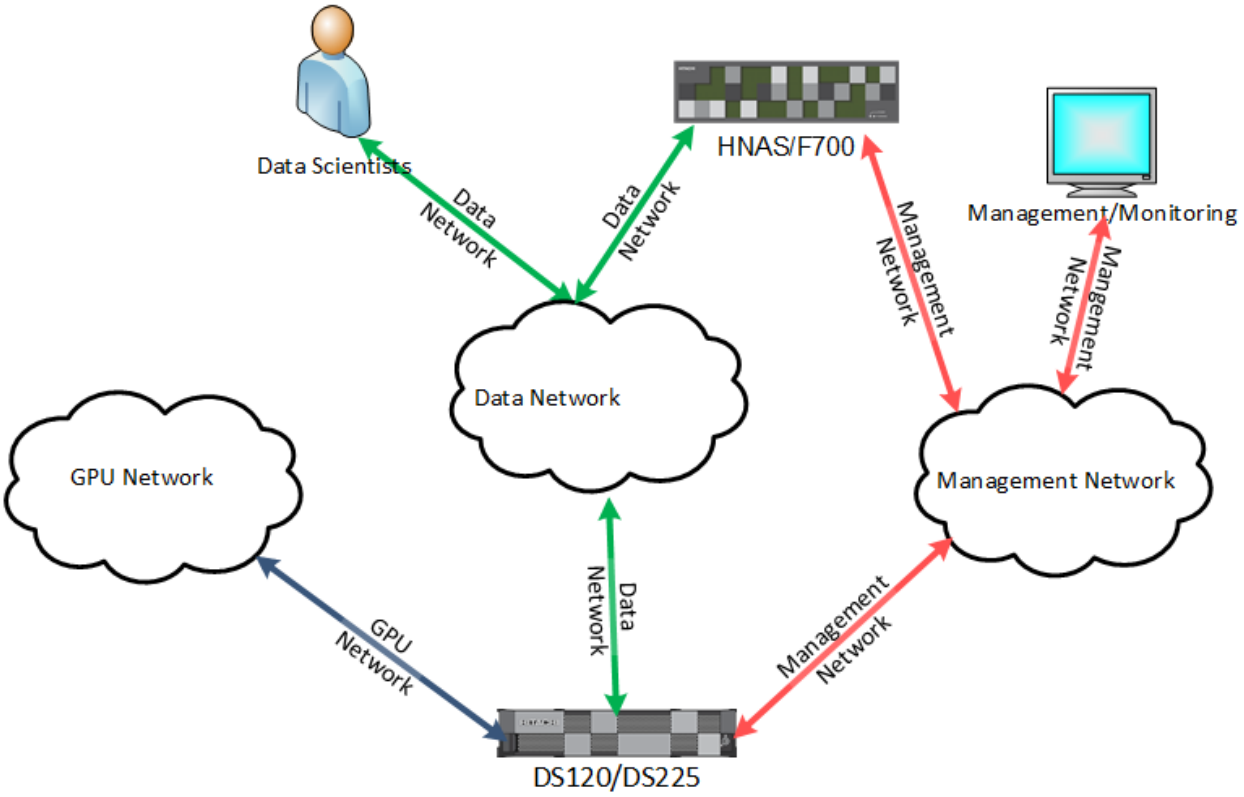


Table 3 lists the ToR switches and the optional spine/aggregate switches used for both networks.

TABLE 3. SWITCHES

Component	Description
Management Switch ToR	■ 1 × Cisco Nexus 3048
ToR Switches	■ 2 × Cisco Nexus 9336C used for both GPU and OS networks

Learning Data

There are many ways to store and access data. If you are using a cluster, then a shared method is needed:

- Local Storage – all data is copied to the local storage devices outside of the ML\DL code.
- NFS – this allows you store data in a central location.
- Remote Data – ML\DL code directly accesses remote data. This can be a remote database, a data stream, or HDFS.
- Shared Storage with a clustered file system – this solution uses shared SAN or shared NVMe over fabric. A clustered file system, like GFS2 is used so all nodes can access the data.
- Distributed storage and file system – GlusterFS or BeeGFS. These file systems spread the data across the local storage on all the nodes and allow you to treat it as one unit of storage.

Caching the data can improve performance. ML\DL will take one, usually small, set of data and process it tens, hundreds, or even thousands of time. Reducing the data access time for each loop by caching the data in memory or fast local disk can have a significant impact. If you are using a file system, the operating system can perform file level caching for the data that fits in memory.

You can have ML\DL cache the data. This would require coding to make use of these features. This has the following advantages:

- Works with all data sources
- Can cache prepared data, which will improve performance
- Can cache the data in whatever underlying method you prefer memory, local storage, SAN, or NVMe over Fabric

Whether using a single standalone, multiple standalone servers, or a cluster of servers; these options can meet your needs.

PMI Testing

Installing the Plugin Machine Intelligence and Deep Learning

Installation of PMI is done via the Pentaho Marketplace in spoon. By default, installation of PMI will automatically install:

- Weka for PMI
- Spark MLlib library
- DL4j library

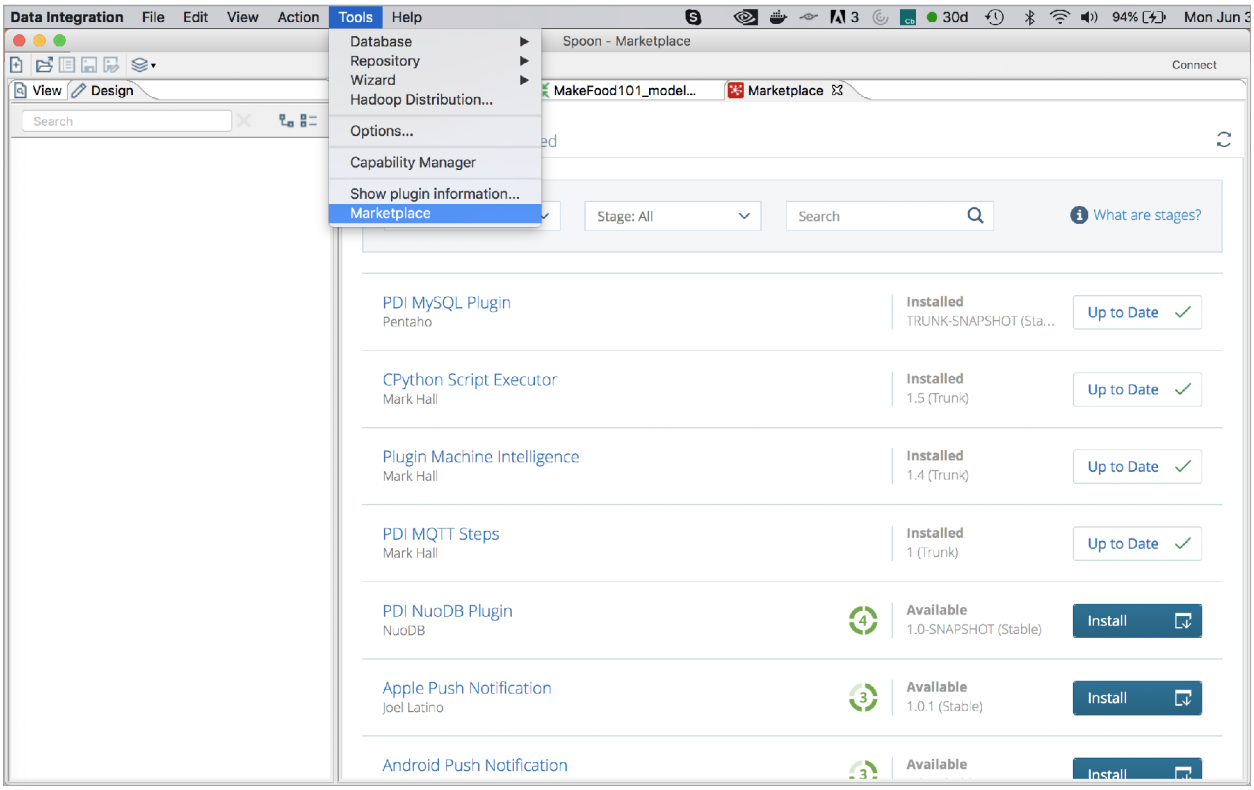
This will be available once spoon is restarted.

There are two additional machine learning libraries that can be used by PMI, Python Scikit-learn, and R MLR. However, these libraries must be downloaded, installed, and configured manually. Detailed instructions for installing and configuring PMI v1.4 including the instructions for manually installing and configuring Python Scikit-learn and R MLR for Linux, OSX, and Windows can be found on the Hitachi Vantara Community web site in the Pentaho Community space at this location:

- https://community.hitachivantara.com/community/products-and-solutions/pentaho/blog/2018/11/13/pmi-installation-and-developer-guides

Figure 7 shows where the Pentaho Marketplace option is and where to install PMI.

Figure 7



Since this white paper is only using the deep learning feature of PMI, DL4j, and this is automatically installed, installing and configuring Python Scikit-learn and R MLR is not required. There is a message when spoon is started and PMI is used stating that these two libraries cannot be found. This condition is normal and can be ignored.

Deep Learning Model Creation Performance

This is not a true benchmark application. This is a comparison of the time it takes to create a DL model using only the DS225 CPUs compared to using the same dataset, same Pentaho transformation and PMI using a configured NVidia Tesla V100 GPU. We accomplish this by creating the CPU-based DL model before configuring the GPU software components and drivers, CUDA and CNN, for the “PMI Deep Learning Network” step, thus making sure that the Pentaho transformation does not recognize the GPUs and therefore cannot access the GPU.

The configuration comparison between the two DL model training processes are shown in Table 4.

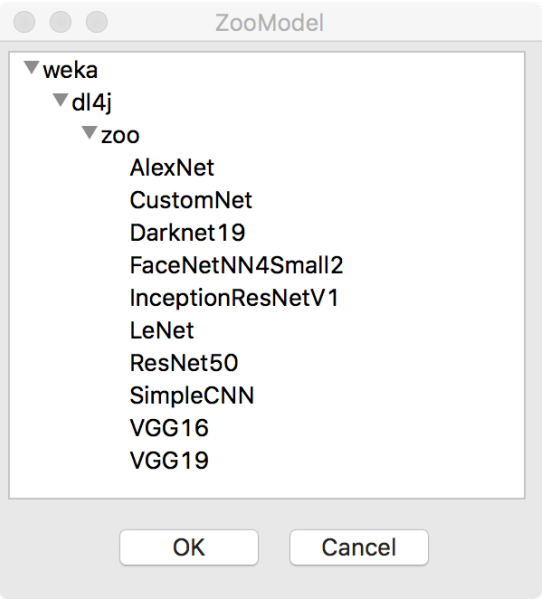
TABLE 4. SERVER CONFIGURATION DIFFERENCE BETWEEN CPU ONLY AND WITH GPU MODEL TRAINING

CPU Configured Training	GPU Configured Training
1 × DS225	1 × DS225
2 × Intel 6140 18x core; 135W 2.3 GHz	2 × Intel 6140 18 core; 135W; 2.3GH
25 × 64GB DIMMS – 1.5TB total	25 × 64 GB DIMMS- 1.5 TB total
	1 × NVIDIA TESLA V100

Deep Learning Model – AlexNet

For this performance comparison, we chose the AlexNet DL model. This is a good general-purpose DL model to experiment with overall. AlexNet is the name of a convolutional neural network (CCN) that is supplied by the ZooModel in DL4j library, the DL library used in PMI. There are 10 prebuilt models available in PMI V1.4 to choose from and train. Figure 8 shows a full list of the available DL models in the DL4j ZooModel.

Figure 8



The model training phase of most DL projects is typically the longest and most compute intensive part of using DL. One could argue that the data preparation phase is the most challenging part, but this white paper is addressing pure computational resources used to create a DL model.

The "Food-101" Dataset

There is an Internet dataset used for DL experimentation and education called "Food-101". This is a 101 class dataset for different food classes. The dataset download image is only 5 GB, but it contains 1000 images per class for a total of 101,000 images that needs to be preprocessed and processed to train a DL model. This dataset was chosen because it is complicated and compute intensive enough to illustrate a distinct advantage between training DL models on CPUs and GPUs. For illustrative purposes, Figure 9 lists the classifications of all the food classes in the Food-101 dataset.

There is more detailed information located at the following website for the Food-101 dataset:

- https://github.com/stratospark/food-101-keras

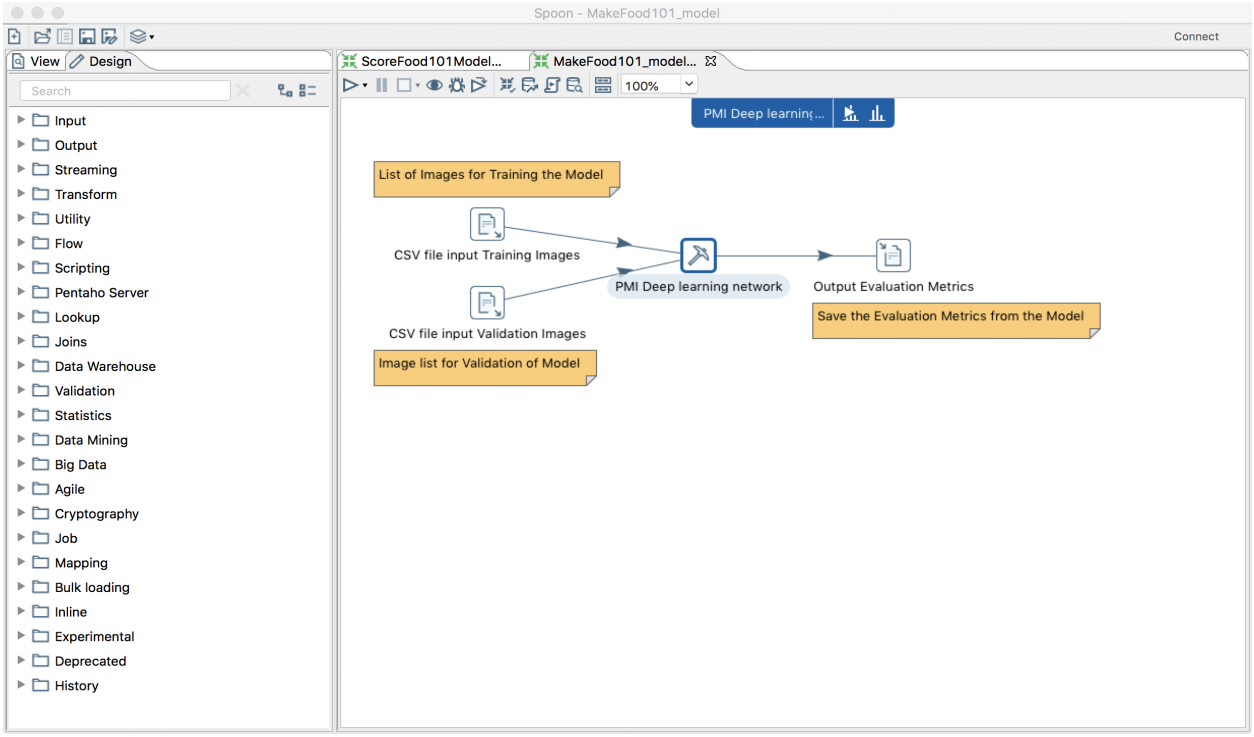
Figure 9

```
apple_pie            eggs_benedict            onion_rings
baby_back_ribs       escargots                oysters
baklava              falafel                  pad_thai
beef_carpaccio       filet_mignon             paella
beef_tartare         fish_and_chips           pancakes
beet_salad           foie_gras                panna_cotta
beignets             french_fries             peking_duck
bibimbap             french_onion_soup        pho
bread_pudding        french_toast             pizza
breakfast_burrito    fried_calamari           pork_chop
bruschetta           fried_rice               poutine
caesar_salad         frozen_yogurt            prime_rib
cannoli              garlic_bread             pulled_pork_sandwich
caprese_salad        gnocchi                  ramen
carrot_cake          greek_salad              ravioli
ceviche              grilled_cheese_sandwich  red_velvet_cake
cheesecake           grilled_salmon           risotto
cheese_plate         guacamole                samosa
chicken_curry        gyoza                    sashimi
chicken_quesadilla   hamburger                scallops
chicken_wings        hot_and_sour_soup        seaweed_salad
chocolate_cake       hot_dog                  shrimp_and_grits
chocolate_mousse     huevos_rancheros         spaghetti_bolognese
churros              hummus                   spaghetti_carbonara
clam_chowder         ice_cream                spring_rolls
club_sandwich        lasagna                  steak
crab_cakes           lobster_bisque           strawberry_shortcake
creme_brulee         lobster_roll_sandwich    sushi
croque_madame        macaroni_and_cheese      tacos
cup_cakes            macarons                 takoyaki
deviled_eggs         miso_soup                tiramisu
donuts               mussels                  tuna_tartare
dumplings            nachos                   waffles
edamame              omelette
```

PDI DL Training Transformation

When this performance test is complete, there will be 2 DL models, one from using the CPUs to build the model, and one from using the GPU to build the model. The same PDI transformation is used to train both DL models and is shown in Figure 10. At the beginning of the PDI transformation, there are two “CSV File Input” steps used to open and read comma separated value (csv) files that provide the location of the images to the “PMI Deep Learning Network” step for training and validating the model.

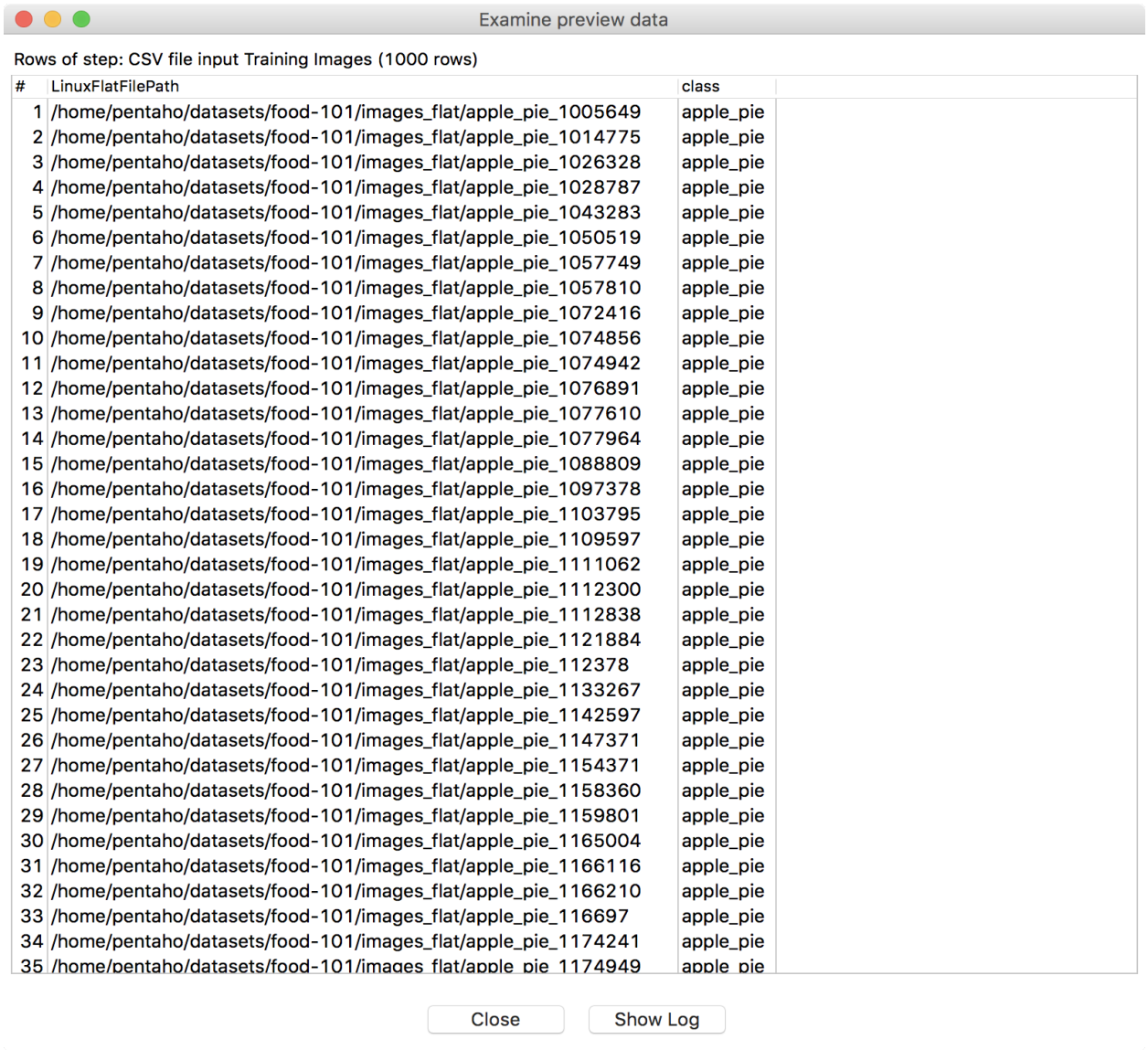
Figure 10



There is some data preparation that is required to be completed to the dataset from the original layout in order to put the dataset in a format and position to be processed. This is part of the data preparation phase of many ML and DL projects and is not covered in this white paper.

An example of a processed csv file that lists the location and name of the image file and the class of the image can be seen in Figure 11.

Figure 11



#	LinuxFlatFilePath	class
1	/home/pentaho/datasets/food-101/images_flat/apple_pie_1005649	apple_pie
2	/home/pentaho/datasets/food-101/images_flat/apple_pie_1014775	apple_pie
3	/home/pentaho/datasets/food-101/images_flat/apple_pie_1026328	apple_pie
4	/home/pentaho/datasets/food-101/images_flat/apple_pie_1028787	apple_pie
5	/home/pentaho/datasets/food-101/images_flat/apple_pie_1043283	apple_pie
6	/home/pentaho/datasets/food-101/images_flat/apple_pie_1050519	apple_pie
7	/home/pentaho/datasets/food-101/images_flat/apple_pie_1057749	apple_pie
8	/home/pentaho/datasets/food-101/images_flat/apple_pie_1057810	apple_pie
9	/home/pentaho/datasets/food-101/images_flat/apple_pie_1072416	apple_pie
10	/home/pentaho/datasets/food-101/images_flat/apple_pie_1074856	apple_pie
11	/home/pentaho/datasets/food-101/images_flat/apple_pie_1074942	apple_pie
12	/home/pentaho/datasets/food-101/images_flat/apple_pie_1076891	apple_pie
13	/home/pentaho/datasets/food-101/images_flat/apple_pie_1077610	apple_pie
14	/home/pentaho/datasets/food-101/images_flat/apple_pie_1077964	apple_pie
15	/home/pentaho/datasets/food-101/images_flat/apple_pie_1088809	apple_pie
16	/home/pentaho/datasets/food-101/images_flat/apple_pie_1097378	apple_pie
17	/home/pentaho/datasets/food-101/images_flat/apple_pie_1103795	apple_pie
18	/home/pentaho/datasets/food-101/images_flat/apple_pie_1109597	apple_pie
19	/home/pentaho/datasets/food-101/images_flat/apple_pie_1111062	apple_pie
20	/home/pentaho/datasets/food-101/images_flat/apple_pie_1112300	apple_pie
21	/home/pentaho/datasets/food-101/images_flat/apple_pie_1112838	apple_pie
22	/home/pentaho/datasets/food-101/images_flat/apple_pie_1121884	apple_pie
23	/home/pentaho/datasets/food-101/images_flat/apple_pie_112378	apple_pie
24	/home/pentaho/datasets/food-101/images_flat/apple_pie_1133267	apple_pie
25	/home/pentaho/datasets/food-101/images_flat/apple_pie_1142597	apple_pie
26	/home/pentaho/datasets/food-101/images_flat/apple_pie_1147371	apple_pie
27	/home/pentaho/datasets/food-101/images_flat/apple_pie_1154371	apple_pie
28	/home/pentaho/datasets/food-101/images_flat/apple_pie_1158360	apple_pie
29	/home/pentaho/datasets/food-101/images_flat/apple_pie_1159801	apple_pie
30	/home/pentaho/datasets/food-101/images_flat/apple_pie_1165004	apple_pie
31	/home/pentaho/datasets/food-101/images_flat/apple_pie_1166116	apple_pie
32	/home/pentaho/datasets/food-101/images_flat/apple_pie_1166210	apple_pie
33	/home/pentaho/datasets/food-101/images_flat/apple_pie_116697	apple_pie
34	/home/pentaho/datasets/food-101/images_flat/apple_pie_1174241	apple_pie
35	/home/pentaho/datasets/food-101/images_flat/apple_pie_1174949	apple_pie

The “PMI Deep Learning Network” step has 2 key configuration tabs that need to be setup for this PDI step. These are the “Fields” tab and the “Algorithm” tab.

Figure 12 shows the “Fields” tab in the “PMI Deep Learning Network” step and how the data from the two “CSV File Input” steps are configured, as well as how to set up the “class” field and the classes. The “class” field must be set to “Nominal” regardless of the defaults that might be preset when the **Get Fields** button is clicked. All 101 classes must be entered manually (copy and pasted) as they are provided in the csv files in the **Nominal Values** field. **Training Data Source** and **Separate Test Set Source** must be selected for the associated incoming data from the CSV File Input steps.

Figure 12

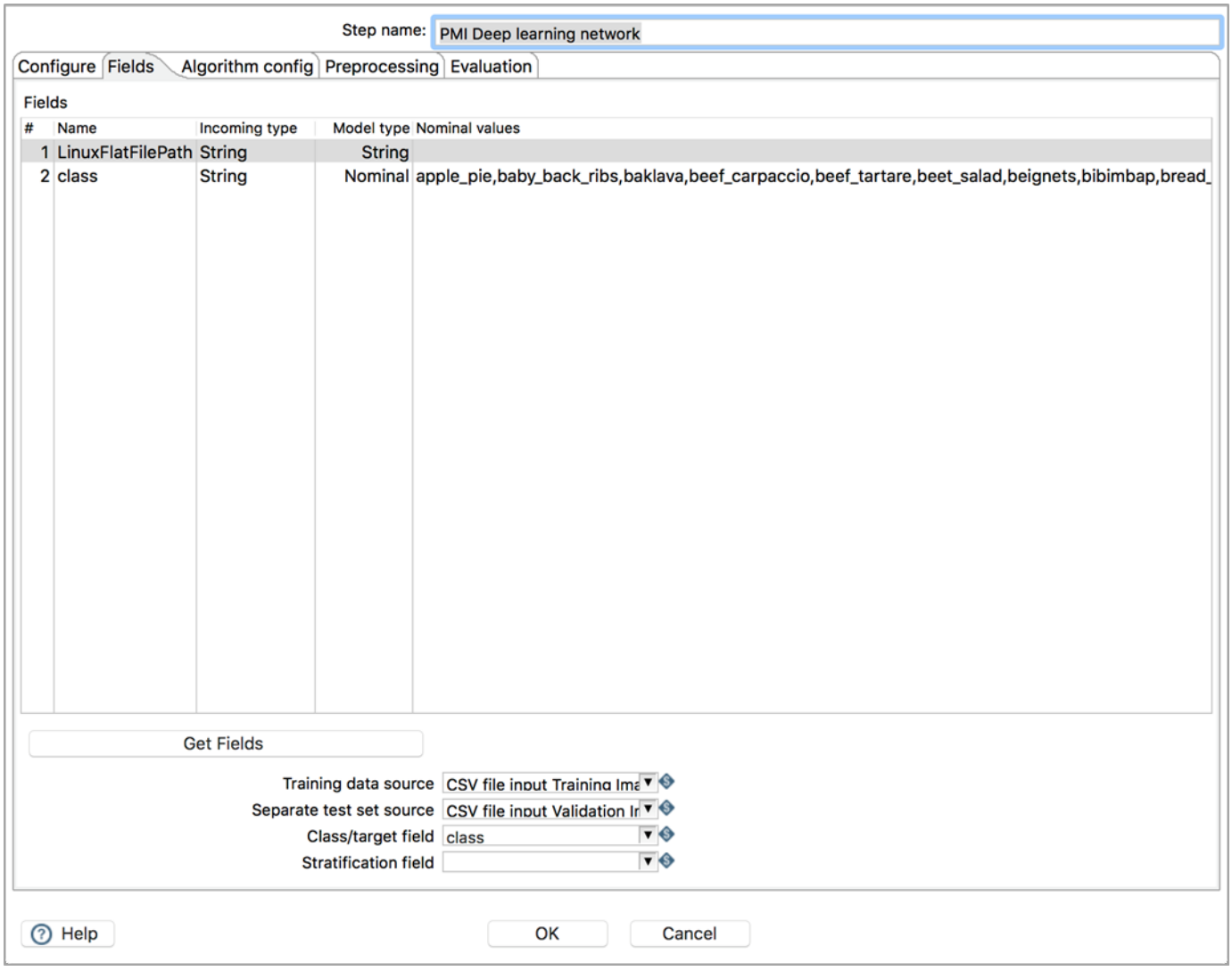
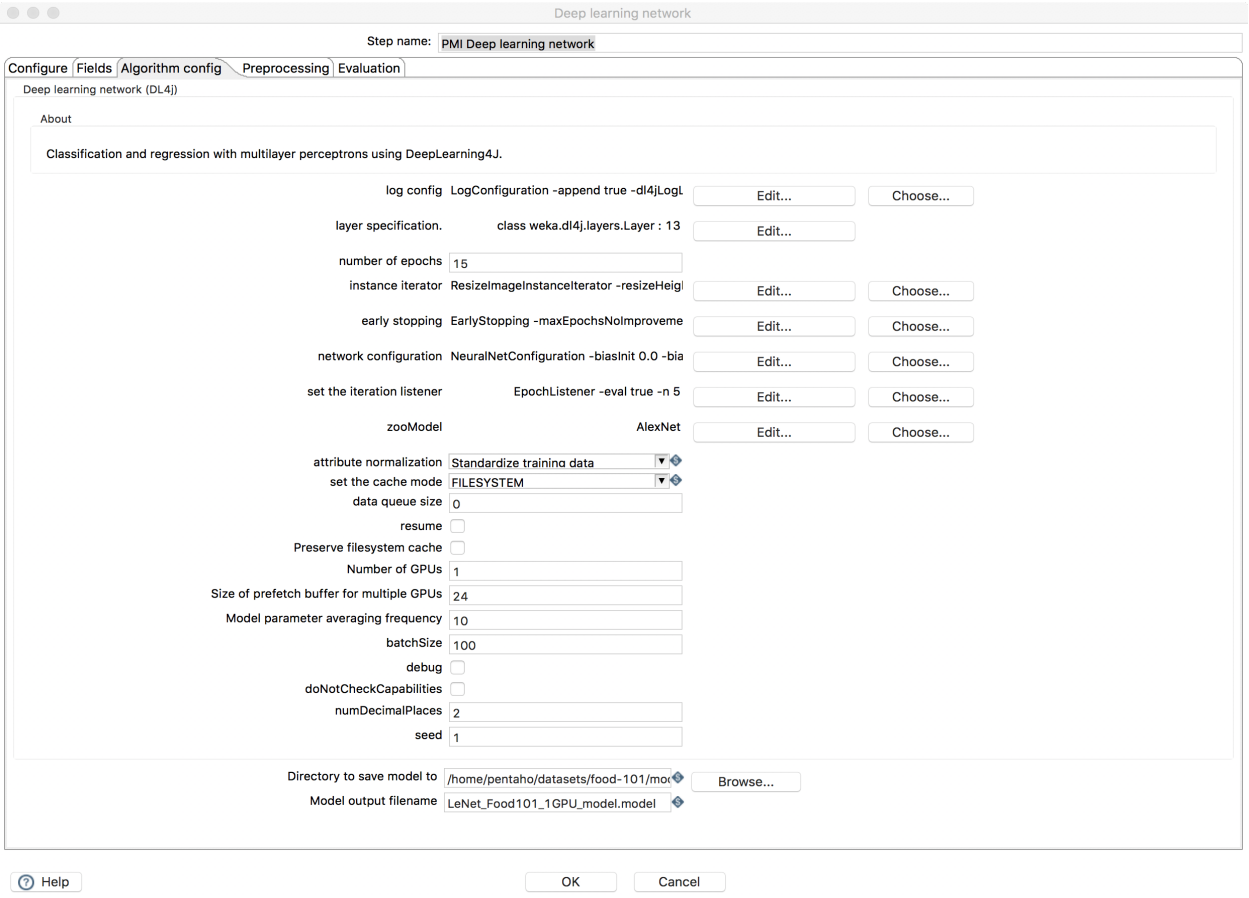


Figure 13 shows the tunable parameters and setup of the **Algorithm Config** tab. By design, the default settings in all PMI algorithms should execute without issues and should result in a model when complete. The skillset required to tune these parameters are part of the role a data scientist plays in getting the maximum accuracy from this phase of any DL project. In this case, the following settings need to be changed or entered for this project:

- The number of times the dataset is passed through the neural network setting – **Number of epochs** field
  - 15 is used here for the purpose of this project
- **AlexNet** is used and selected for – **ZooModel**
- Location to store the DL model – **Directory to save model to**
- Name of the model file – **Model output filename**

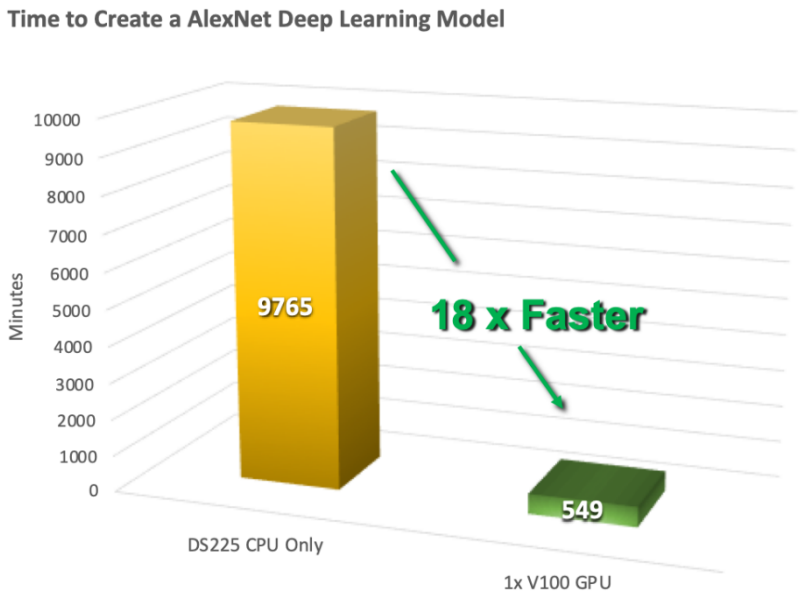
Figure 13



Performance

Figure 14 illustrates the difference between training a DL model on the CPUs versus training the DL model on a GPU. As can be seen, the amount of time required to train and build a DL model with all available CPUs takes 9,765 minutes, 162.75 hours, or 6.78 days – almost 1 week. While using the 1 GPU to build and train the exact same DL model and the same PDI transformation takes 549 minutes or 9.15 hours. This results in an almost 18 times increase in productivity.

Figure 14



Conclusion

This white paper demonstrates that the combination of Pentaho, Plugin Machine Intelligence from Hitachi Vantara Labs, and the DS225 server with NVidia Telsa V100 GPU (for compute-intensive processing) makes for a very powerful data science development environment that can dramatically increase the productivity of the data scientist and time to market.

Hitachi Vantara

Corporate Headquarters
5355 Augustine Drive
Santa Clara, CA 96054 USA
HitachiVantara.com | community.HitachiVantara.com

Contact Information
USA: 1-800-446-0744
Global: 1-858-547-4526
HitachiVantara.com/contact

MK-SL-157-00, July 2019