Technical Report

NetApp ONTAP AI Reference Architecture for Financial Services Workloads

Solution Design

Karthikeyan Nagalingam, Sung-Han Lin, NetApp
Jacci Cenci, NVIDIA

December 2019 | TR-4807

In partnership with

NVIDIA

Abstract

This reference architecture offers guidelines for customers who are building artificial intelligence infrastructure using NVIDIA DGX-1™ systems and NetApp® AFF storage for financial sector use cases. It includes information about the high-level workflows used in the development of deep learning models for financial services test cases and results. It also includes sizing recommendations for customer deployments.

NetApp®

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Executive Summary

The NVIDIA DGX™ family is composed of the world's first integrated artificial intelligence (AI) systems that are purpose-built for enterprise AI. NetApp® AFF storage systems deliver extreme performance and industry-leading hybrid cloud data-management capabilities. NetApp and NVIDIA® have partnered to create the NetApp ONTAP® AI reference architecture. This partnership provides customers with a turnkey solution for AI and machine learning (ML) workloads with enterprise-class performance, reliability, and support.

This reference architecture offers guidelines for customers who are building AI infrastructure using DGX-1™ systems and NetApp AFF storage for financial sector use cases. It includes information about the high-level workflows used in the development of DL models for financial services test cases and results. It also includes sizing recommendations for customer deployments.

The target audience for the solution includes the following groups:

- Infrastructure and enterprise architects who design solutions for the development of AI models and software for financial use cases such as credit card fraud analysis.
- Data scientists who are looking for efficient ways to achieve DL development goals.
- Executive and IT decision makers who are interested in achieving the fastest time to value from AI initiatives.

2 Solution Overview

2.1 Credit Card Fraud Detection Use Case

According to a recent study, U.S. credit card fraud rose to $9 billion per year in 2016 and is expected to increase to $12 billion by 2020. Many banks have used rules-based expert systems to catch fraud. However, these methods have become too easy to beat. To improve their defenses, the financial services industry is relying on increasingly complex fraud detection algorithms, including ML algorithms such as classifiers, linear approaches, and support vector machines.

Some companies have pioneered more advanced AI techniques such as deep neural networks and autoencoders. Autoencoders are a type of neural network that takes an input, boils down (encodes) it to its core features in an unsupervised manner, and then reverse encodes the data to recreate the input.

The financial services sector generates a wide variety of data types. Analysis can include transaction history data from banks; smartphone data; real-time structured and unstructured data; a client's behavior, location, and buying habits; and speech data from banking call centers. The different data types contribute to different aspects of financial services, including credit decisions, risk management, fraud prevention, trading, and personalized banking. Model training requirements vary for distinct data types, and the achievable performance on compute and storage resources also varies. The goal is always to saturate the GPUs and provide the highest throughput at the lowest latency from the storage side.

This technical report addresses challenges in the training phase. For this report, the base credit card fraud dataset from Kaggle was used as a foundation and was then magnified by using generative adversarial networks (GANs). Autoencoders from the Keras library with the TensorFlow back-end program were used to detect and validate the fraudulent credit card transactions dataset that resides on the NetApp storage system. Then a model was trained and used to identify instances of fraud. In workflows such as this, NetApp and NVIDIA technologies help deliver best-in-class performance to reduce the time to insight.

3 Solution Technology

The NetApp ONTAP AI architecture, powered by DGX systems and NetApp cloud-connected storage systems, was developed and verified by NetApp and NVIDIA. This reference architecture gives IT organizations the following advantages:

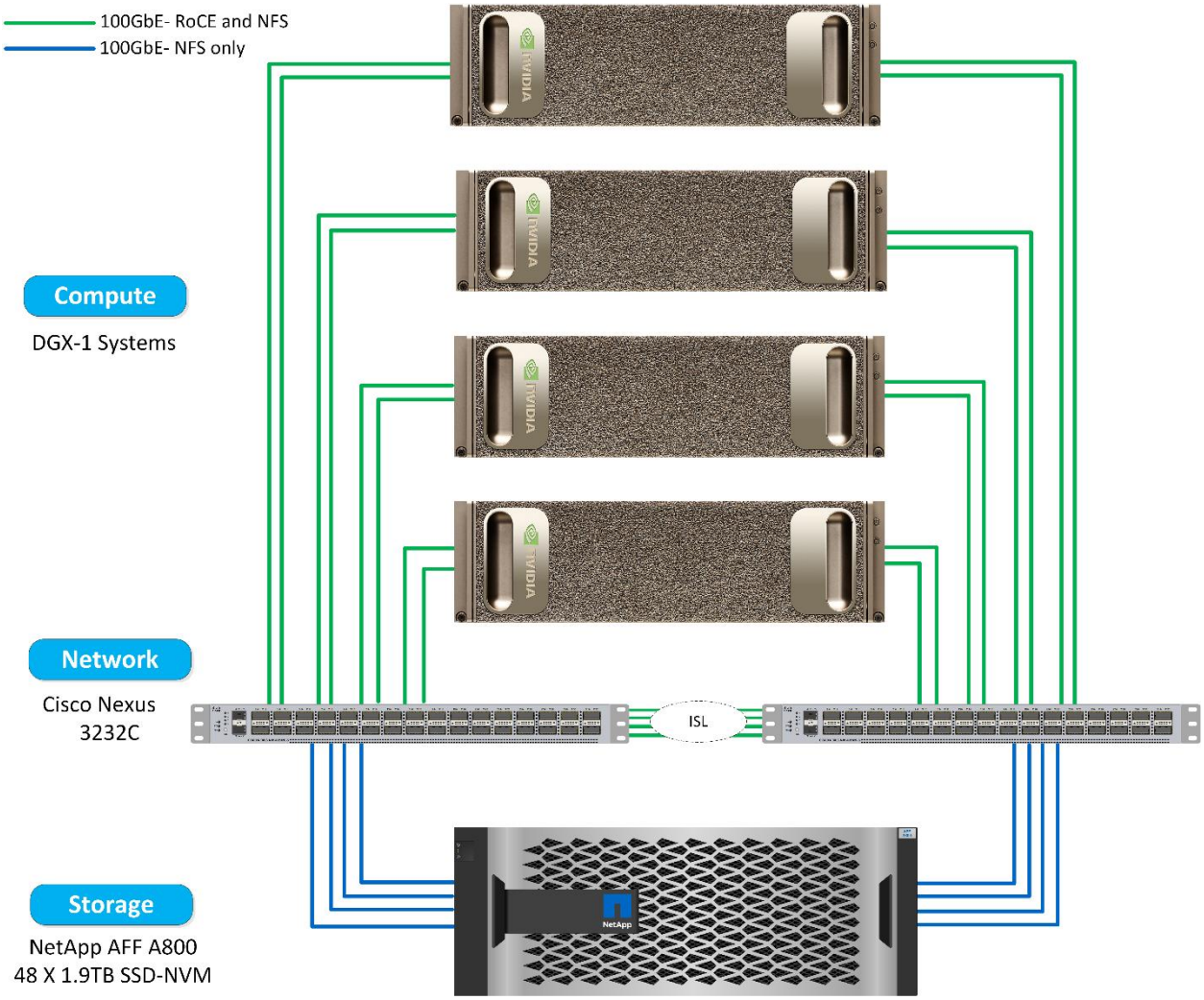
- Eliminates design complexities
- Allows independent scaling of compute and storage
- Enables customers to start small and scale seamlessly
- Offers a range of storage options for various performance and cost points

NetApp ONTAP AI tightly integrates DGX systems and NetApp AFF A800 storage systems with state-of-the-art networking. NetApp ONTAP AI with DGX systems simplifies artificial intelligence deployments by eliminating design complexity and guesswork. Customers can start small and grow their systems in an uninterrupted manner while intelligently managing data from the edge to the core to the cloud and back.

The AFF A800 storage system has been verified with nine DGX-1 systems and three NVIDIA DGX-2™ systems. Furthermore, by adding more network switches and storage controller pairs to the ONTAP cluster, the architecture can scale to multiple racks to deliver extremely high throughput, accelerating training and inferencing. With this flexible approach, the ratio of compute to storage can be altered independently based on the size of the data lake, the models that are used, and the required performance metrics. For detailed information about ONTAP AI with DGX-1 systems, see NetApp Verified Architectures NVA-1121 and NVA-1138. For information about ONTAP AI with DGX-2 systems, see NVA-1135.

This solution was validated with one AFF A800 storage system, four DGX-1 systems, and two Nexus 3232C 100Gb Ethernet (100GbE) switches. As illustrated in Figure 1, each DGX-1 system is connected to the Nexus switches with four 100GbE connections that are used for inter-GPU communications by using remote direct memory access (RDMA) over Converged Ethernet (RoCE). Traditional IP communications for NFS storage access also occur on these links. Each storage controller is connected to the network switches with four 100GbE links.

Figure 1) ONTAP AI financial services solution topology.



3.1 Hardware Requirements

This solution was validated using four DGX-1 systems and one AFF A800 storage system. This configuration is consistent with the 18kW DGX-1 rack design described in the NVIDIA DGX POD Data Center Reference Design.

Table 1 lists the hardware components that are required to implement the solution as tested. The hardware components used in a specific customer implementation should be based on the sizing guidance in Section 5.

Table 1) Hardware requirements.

Hardware	Quantity
DGX-1 systems	4
NetApp AFF A800 system	1 high-availability (HA) pair, including 2 controllers and 48x NVMe SSDs (3.8TB or more)

Cisco Nexus 3232C network switches	2

3.2 Software Requirements

Table 2 lists the software components that are required to implement the solution as tested.

Table 2) Software requirements.

Software	Version or Other Information
NetApp ONTAP data management software	9.5
Cisco NX-OS switch firmware	7.0(3)I6(1)
NVIDIA DGX OS	4.0.4 - Ubuntu 18.04 LTS
Docker container platform	18.06.1-ce [e68fc7a]
Container version	netapp_tf_19.03 based on nvcr.io/nvidia/tensorflow:19.03-py2
ML framework	TensorFlow 1.13.3
Horovod	0.16
OpenMPI	3.1.3
Benchmark software	TensorFlow benchmarks [7b9e1b4]

4 Credit Card Fraud Detection Training and Validation

The model training performance of this solution was validated using credit card transactions datasets for European customers. The following sections contain information about the datasets and the testing results.

4.1 Credit Card Transactions Dataset

This dataset contains transactions made with credit cards by European cardholders over the course of two days in September of 2013. The dataset contains 492 fraudulent transactions out of 284,807 transactions total. Due to the need for confidentiality, dataset features cannot contain original values that would permit the identification of individual credit card users. Rather than containing revealing background information about the data, the dataset contains numerical input variables that are the result of a principal component analysis (PCA) transformation.

Features V1 through V28 are the principal components obtained from the PCA transformation. The three features that have not been transformed with the PCA are Time, Amount, and Class. The Time feature contains the seconds elapsed between each transaction and the first transaction in the dataset. The Amount feature contains the real value used for example-dependent, cost-sensitive learning. The Class feature has a value of 1 for fraudulent transactions and 0 for normal transactions. The dataset size is 144MB.

Exploring the Dataset

Before we proceeded to the training, we wanted to analyze the data first to see if we could find areas of interest that would help us build the training model. Moreover, we wanted to focus only on interesting features, without having our analysis washed out by extraneous information.

We started with the features that are not changed by PCA: Time and Amount (Figure 2). We graphed the normal and fraudulent transactions separately in order to visualize their distributions independently. As shown in the figure, fraudulent transactions can happen at any time, so we excluded the Time feature for training purposes. Unlike the Time feature, the histogram of the Amount feature shows that fraudulent transactions tend to involve relatively small amounts. Therefore, Amount could be a useful feature for transaction classification. However, in order to distinguish the fraudulent transactions from normal transactions, more information from other features was required.

Figure 2) Histograms for the Time and Amount features.

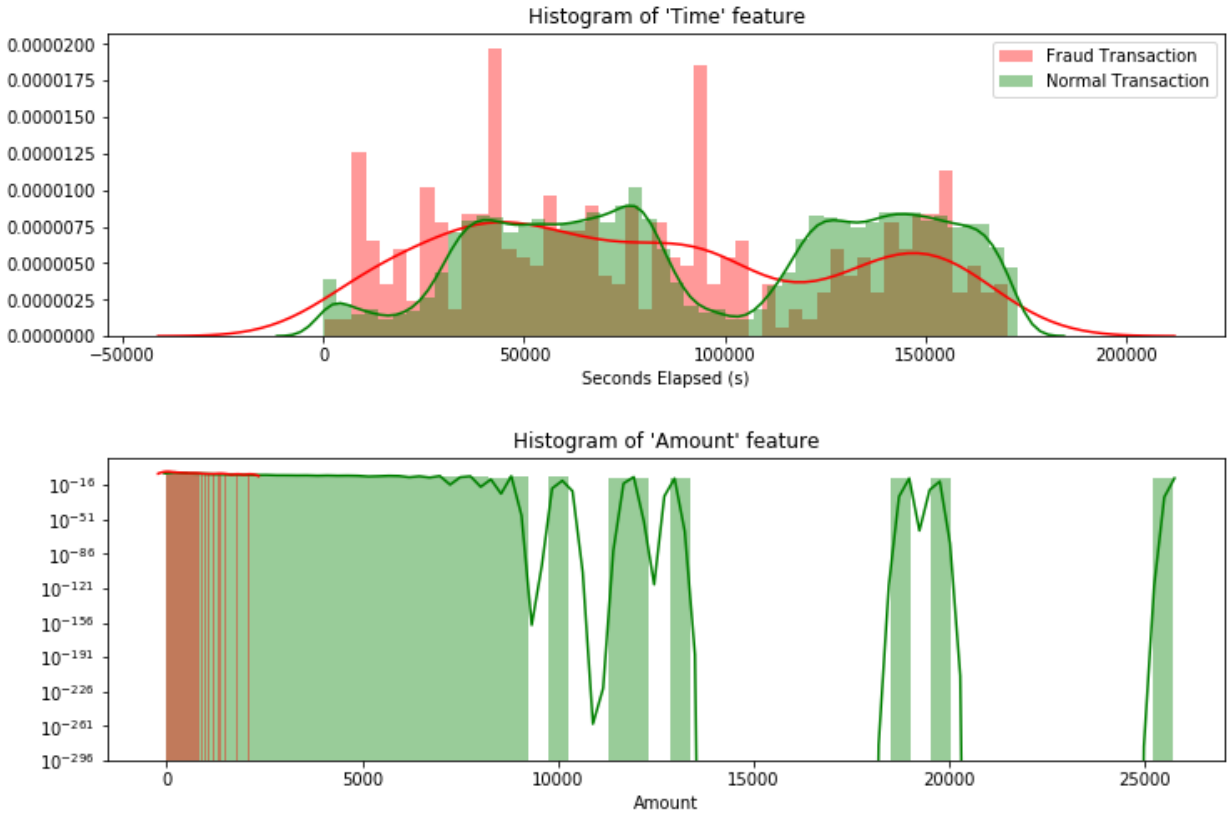
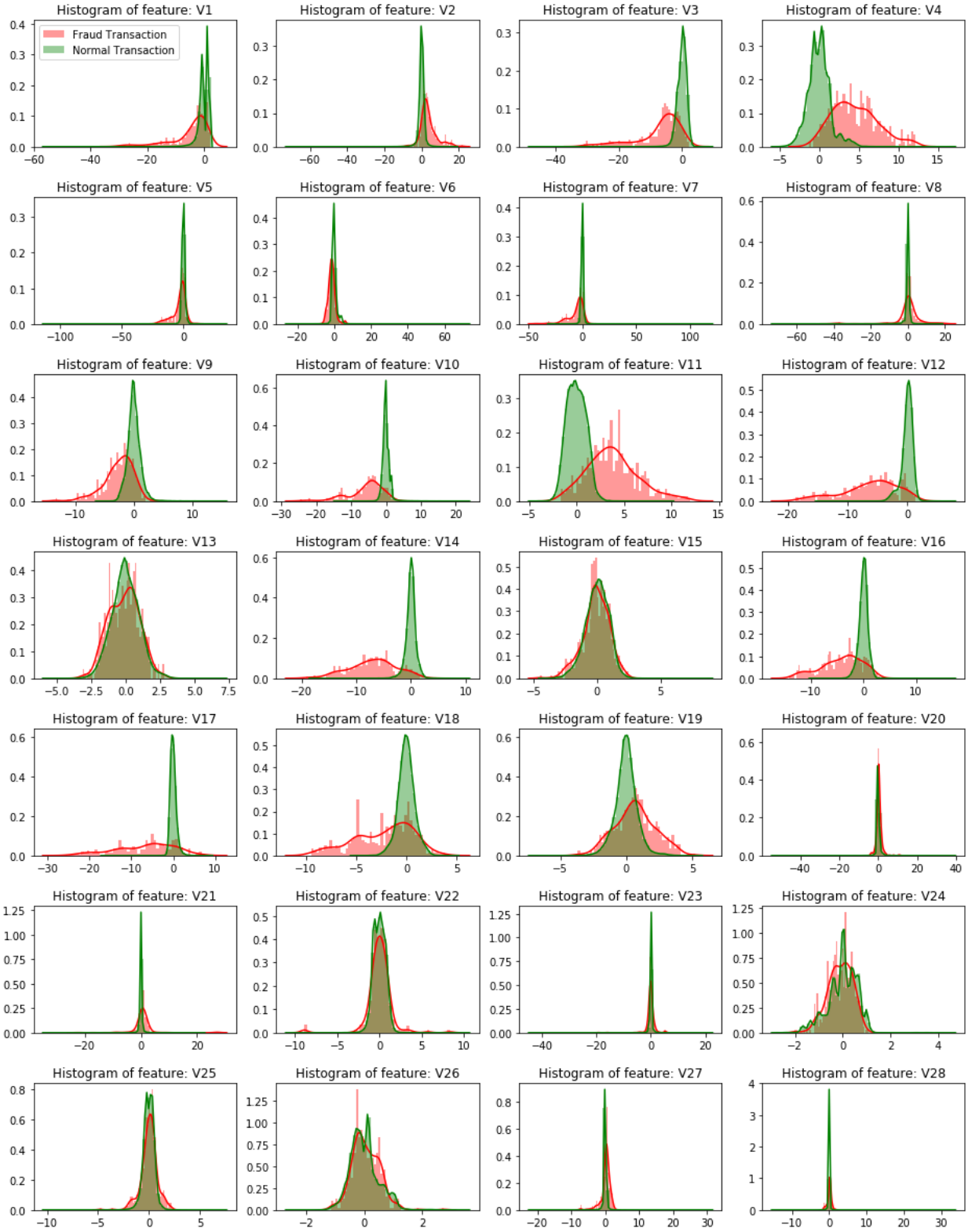


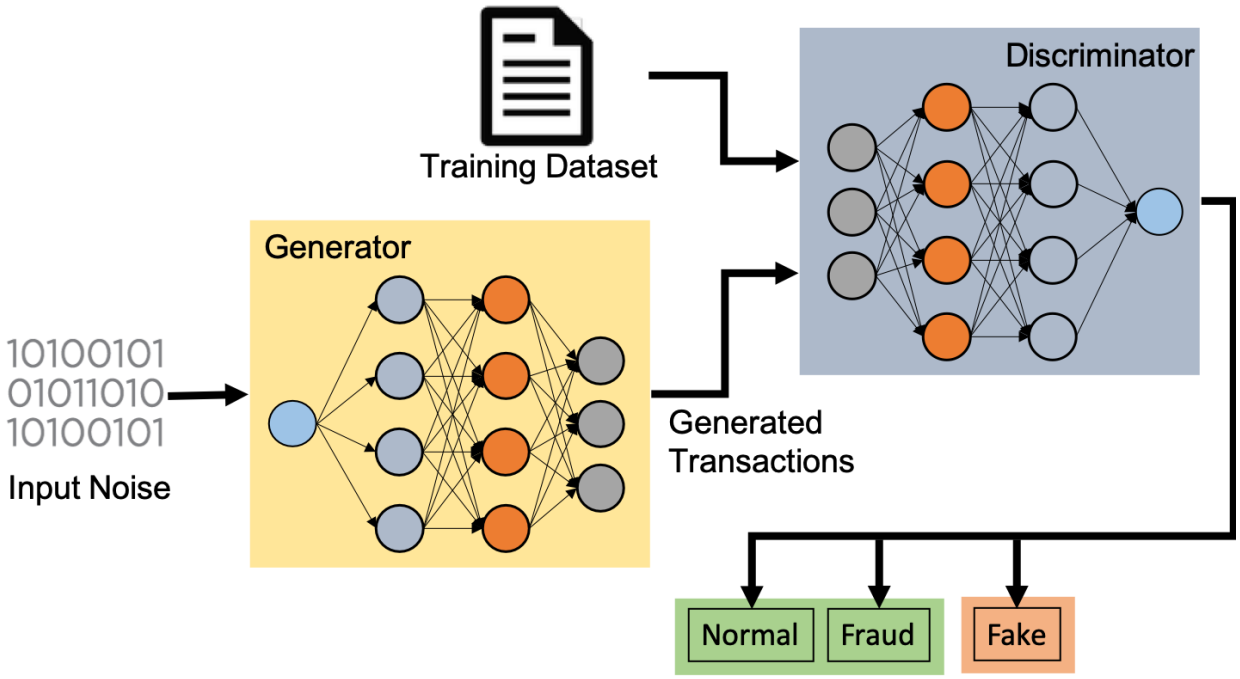
Figure 3 shows the distribution differences between normal and fraudulent transactions for the other 28 features. These distributions show a lot of overlap for each feature. Therefore, we needed a mechanism to correlate those features to better classify normal and fraudulent transactions. In this work, we chose an autoencoder neural network operating in an unsupervised manner for anomaly detection.

Figure 3) Histogram of V1, V2, ..., V28 features from the original dataset.



Even though we can create a reasonably good model to classify fraudulent transactions with the original dataset, we might improve our detection with a larger number of fraudulent transactions. To do so, we generated more realistic normal and fraudulent transactions by using GANs.

Figure 4) Semi-supervised learning GAN architecture for credit card transactions.



4.2 Generative Adversarial Networks

A GAN is a type of neural network that learns to mimic any distribution of data, including images, music, speck, and prose. Each GAN has two components, a generator and a discriminator. The generator is used to generate new data instances by feeding the network a small sample of random noise as input. The discriminator evaluates each instance of data to decide whether the input comes from the generator or from the true training set. These two components play a game with the following steps:

- The generator tries to fool the discriminator by generating realistic data.
- The discriminator tries to distinguish between real and fake data, which guides the generator to produce more realistic data.

Training Phase

The most common way of using GANs is in a supervised manner. Developing a supervised GAN model usually requires a very large amount of data. However, the credit card transaction dataset contains very few fraudulent transactions (less than 1%). We needed to create a model capable of learning from this small amount of data. To this end, we adopted the technique of semi-supervised learning by using both labeled and unlabeled data to train a classifier. The goal is to combine both sources of data to train neural networks to learn an inferred function that can map a new transaction to its desirable outcome.

Figure 4 illustrates the semi-supervised GAN for generating credit card transactions. We transformed the discriminator into a three-class classifier. The first two classes are for the individual class probabilities of the credit card fraud dataset (normal and fraud), and the third class is for all the fake transactions that come from the generator. We then set up the losses to instruct the discriminator to do the following:

- Distinguish between real and fake transactions to help the generator learn to produce realistic images.
- Use the generator's transactions, along with the labeled and unlabeled training data, to help classify the dataset.

The generator takes random noise numbers as input to generate transactions based on the feedback from the discriminator in order to minimize losses.

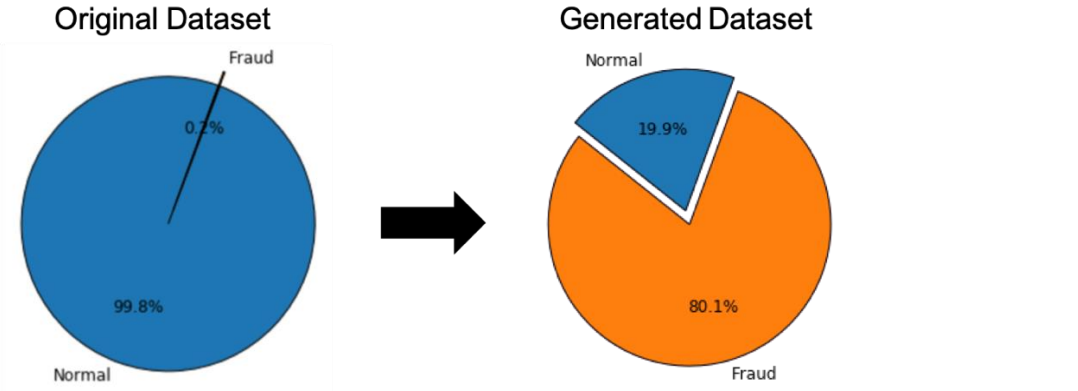
Data Generating Phase

After achieving a certain level of losses, we ended the training phase and used the trained model to generate the new dataset. In this phase, we set aside the real dataset and only used random noise to generate transactions. We still fed the generated transactions into the discriminator and stored only the transactions that could be classified as normal or fraudulent into the new dataset.

4.3 Data Generated by GANs

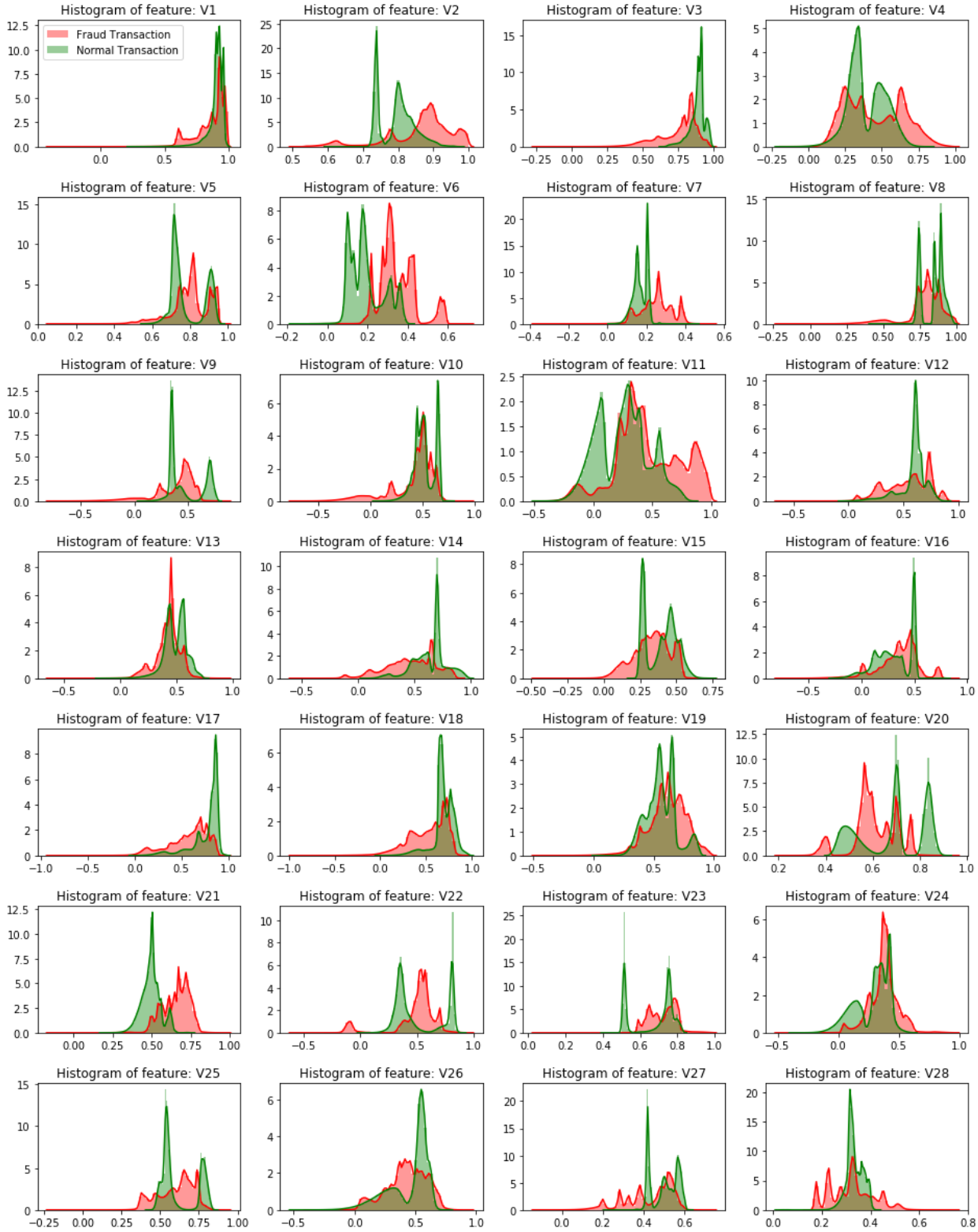
Using the mechanism just described, we generated a new 1.5TB dataset with a much higher proportion of fraudulent transactions. Figure 5 compares the original dataset and the generated dataset, with fraudulent transactions now filling 80% of the dataset.

Figure 5) The composition of the original dataset and the generated dataset.



As with the original dataset, we also determined the distribution differences between normal and fraudulent transactions for the 28 features (see Figure 6). In the figure, observe that many features still follow the same distributions as the original dataset. However, some of the features, for instance V5, have two peaks on their probability curves. This observation is a result of mode collapse, in which the generator becomes stuck on some modes and exhibits poor diversity for the generated samples. Mode collapse is usually an architectural problem, which we will address in future analyses.

Figure 6) Histograms for the V1, V2, ..., V28 features from the GAN-generated dataset.

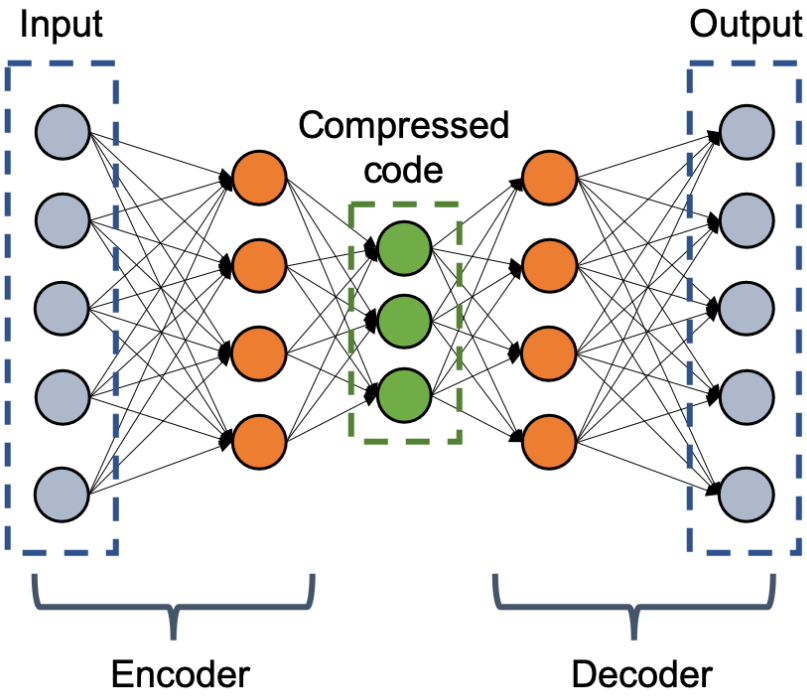


4.4 Accuracy Prediction Using Autoencoder Neural Networks

In the previous section, we used a GAN model to generate more datasets. The discriminator trained during this exercise can then be used directly on the dataset to classify the transactions. However, we would like to use another technique to achieve a similar goal. For this purpose, we built an autoencoder model from scratch.

An autoencoder is an unsupervised artificial neural network that learns to compress data into a reduced encoded representation and then learns to reconstruct the original input. Figure 7 shows an abstract architecture of an autoencoder.

Figure 7) Autoencoder architecture.



An autoencoder contains three main components:

- **Encoder.** This component learns to reduce the input dimensions and compress the input data into an encoded representation.
- **Compressed code.** The output of the encoder, which usually has the lowest possible dimensions of the input data.
- **Decoder.** This component learns to reconstruct an encoded representation of the original input data.

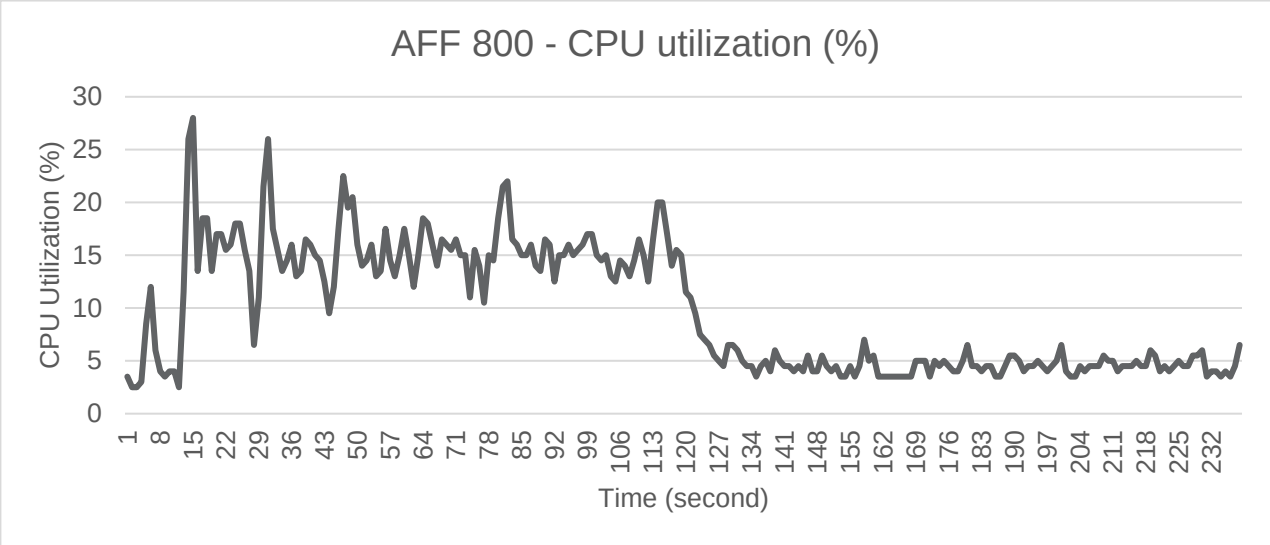
The goal of this training model is to minimize the difference between the original input data and the reconstructed data, which is usually called the reconstruction loss.

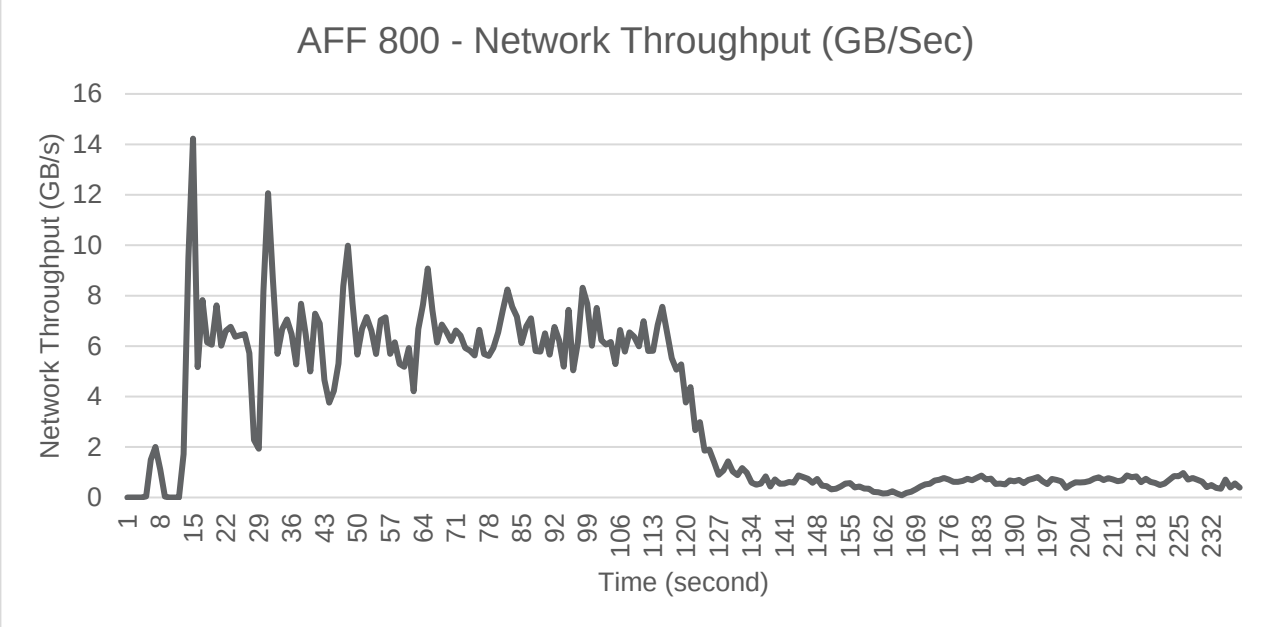
To better understand the performance of this solution, an autoencoder model was developed with a two-layer encoder and a two-layer decoder. We trained the autoencoder model with the generated dataset and validated the model with the original credit card fraud dataset.

Figure 8 illustrates the network throughput and CPU utilization from an AFF A800 storage system and four DGX-1 systems. This figure shows a throughput spike of around 14GB/s at the beginning and then a sustained throughput of around 800MB/s for the remainder of the training run. This spike results from the caching behavior of the training pipeline. After filling up the buffer, the throughput represents the amount of data consumed by the training model during the real training time. As shown in Figure 8, the sustained

CPU utilization for the AFF A800 storage system is below 8%, indicating significant remaining headroom for additional DGX-1 systems that can be used to run mixed workload like NLP.

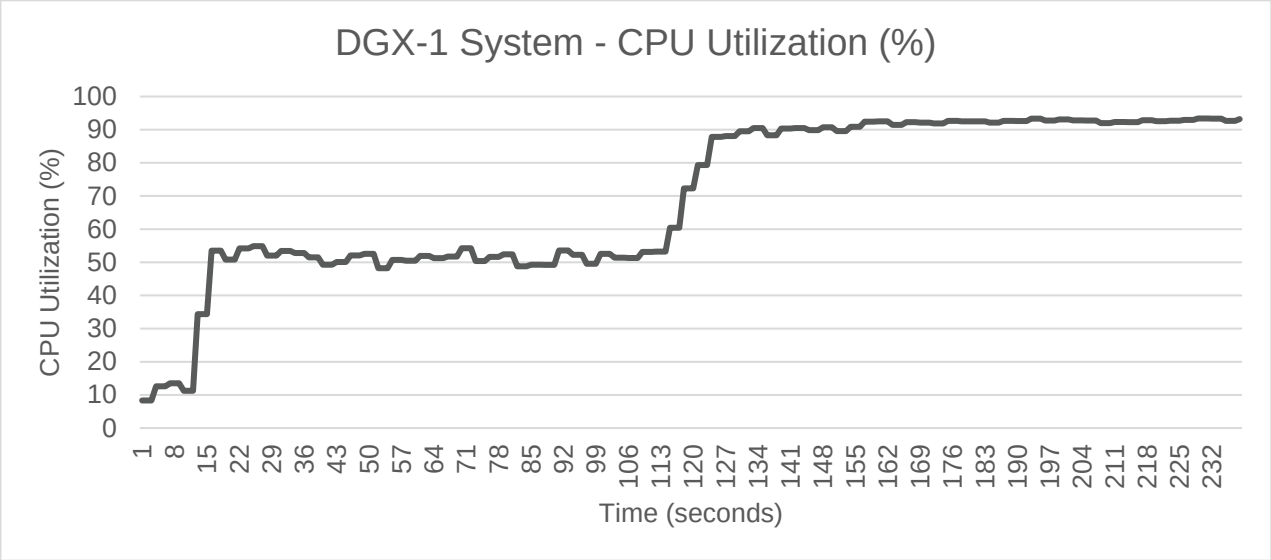
Figure 8) AFF A800 storage system CPU utilization and network throughput for the credit card transactions dataset.

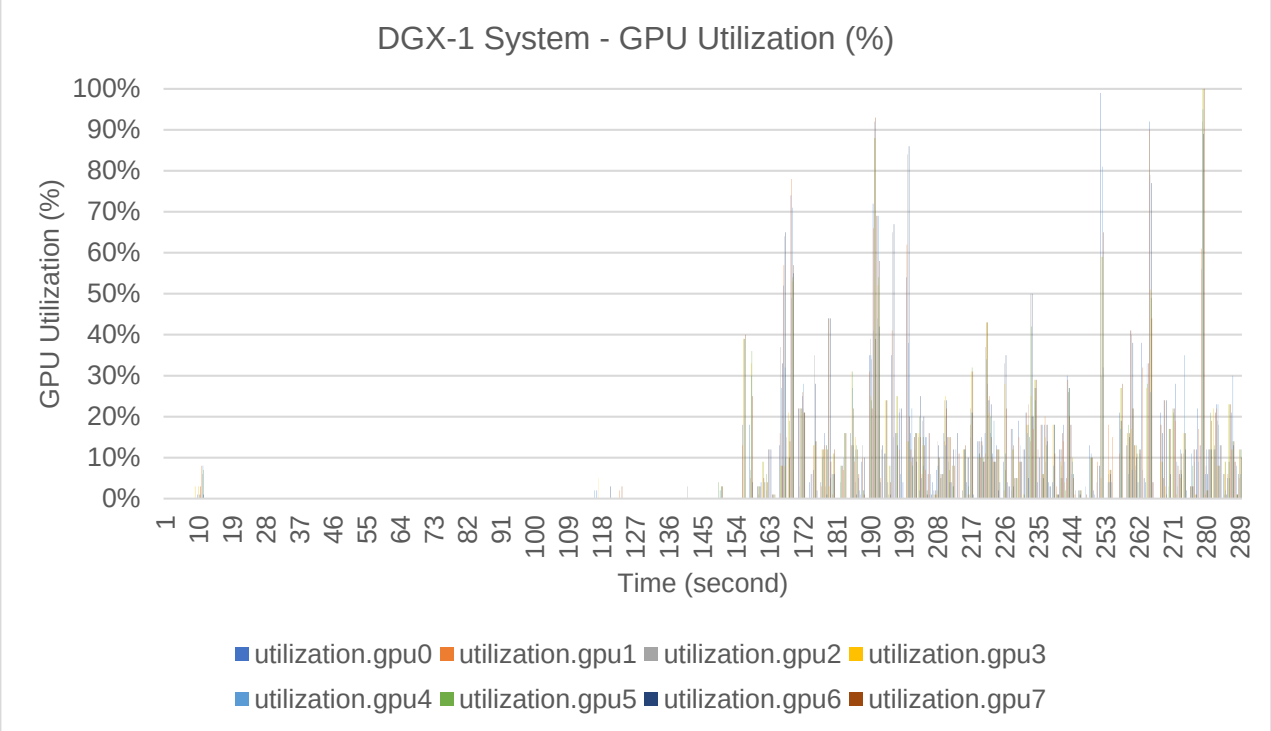




To better explain what happened during training, Figure 9 shows the CPU and GPU utilization for a single DGX-1 system. This figure shows that the throughput spike comes when the DGX-1 systems can use all their CPU cycles to prefetch data. After the prefetch phase, CPUs on the DGX-1 systems start to decode and structure the input, and the CPUs are fully utilized. However, this leads to fewer CPU cycles for fetching data, resulting in lower sustained throughput. To this end, GPUs on the DGX-1 systems cannot be fully utilized because the CPUs cannot feed data fast enough to meet demand.

Figure 9) DGX-1 system CPU and GPU utilization for the credit card transactions dataset.





5 Solution Sizing Guidance

As validated in this solution, as well as in NVA-1121, most AI training workloads require storage read throughput of roughly 1.5GB/s per DGX-1 system. Based on the test results above using four DGX-1 systems, the AFF A800 storage system could support 12 DGX-1 systems with the same workload characteristics. The synthetic performance benchmarks performed in NVA-1121 show that the AFF A800 storage system delivers up to 25GB/s of read performance and thus can easily support 12 DGX-1 systems with the model training workload demonstrated here.

The data used in financial services is the combination of a real dataset and a GANs-generated dataset. We saw an efficiency savings of 1.3:1 for the financial services data using NetApp storage features like compression, deduplication, and compaction. The storage savings are based on the type of data and uniqueness. NetApp AFF A800 systems support a variety of SSD capacity options, ranging from 100TB to 800TB per A800 storage system. Customers can choose the SSD size that meets their capacity requirements without any effect on performance.

For organizations that require more than 12 DGX-1 systems, NetApp ONTAP supports storage clusters of up to 24 nodes, enabling linear scaling of capacity and performance as DGX-1 systems are added to the environment.

For detailed sizing for specific workload requirements, consult a NetApp technical representative.

6 Conclusion

The ONTAP AI reference architecture is an optimized platform for the development of ML models for financial credit card fraud detection and many other use cases. With the accelerated compute power of DGX-1 systems and the performance and data management of NetApp ONTAP, ONTAP AI enables a full range of data pipelines that spans the edge, the core, and the cloud for successful financial services projects.

The models and datasets tested in this solution show that ONTAP AI can easily support the workload requirements for model training with credit card transactions datasets. DGX-1 system CPU and/or GPU resources were effectively utilized based on the tested workload, while the AFF A800 storage system delivered all the required storage performance with headroom to spare.

Acknowledgements

The authors gratefully acknowledge the contributions that were made to this technical report by our esteemed colleagues from NVIDIA, Darrin Johnson and Robert Sohigian. The authors would also like to acknowledge the contributions of key NetApp team members David Arnette, Santosh Rao, and Erik Mulder.

Our sincere appreciation and thanks go to all these individuals, who provided insight and expertise that greatly assisted in the creation of this paper.

Where to Find Additional Information

To learn more about the information that is described in this document, see the following resources:

- NVIDIA DGX-1 systems
  - NVIDIA DGX-1 systems
    https://www.nvidia.com/en-us/data-center/dgx-1/
  - NVIDIA Tesla V100 Tensor core GPU
    https://www.nvidia.com/en-us/data-center/tesla-v100/
  - NVIDIA GPU Cloud
    https://www.nvidia.com/en-us/gpu-cloud/
- NetApp AFF systems
  - AFF datasheet
    https://www.netapp.com/us/media/ds-3582.pdf
  - NetApp Flash Advantage for AFF
    https://www.netapp.com/us/media/ds-3733.pdf

  - ONTAP 9.x documentation
    http://mysupport.netapp.com/documentation/productlibrary/index.html?productID=62286
  - NetApp FlexGroup technical report
    https://www.netapp.com/us/media/tr-4557.pdf
- NetApp ONTAP AI
  - ONTAP AI with DGX-1 and Cisco Networking Design Guide
    https://www.netapp.com/us/media/nva-1121-design.pdf
  - ONTAP AI with DGX-1 and Cisco Networking Deployment Guide
    https://www.netapp.com/us/media/nva-1121-deploy.pdf
  - ONTAP AI with DGX-1 and Mellanox Networking Design Guide
    http://www.netapp.com/us/media/nva-1138-design.pdf
  - ONTAP AI with DGX-2 Design Guide
    https://www.netapp.com/us/media/nva-1135-design.pdf
- ONTAP AI networking
  - Cisco Nexus 3232C series switches
    https://www.cisco.com/c/en/us/products/switches/nexus-3232c-switch/index.html
  - Mellanox Spectrum 2000-series switches
    http://www.mellanox.com/page/products_dyn?product_family=251&mtag=sn2000
- Machine learning frameworks and tools
  - TensorFlow: An Open-Source Machine Learning Framework for Everyone
    https://www.tensorflow.org/
  - Horovod: Uber's Open-Source Distributed Deep Learning Framework for TensorFlow
    https://eng.uber.com/horovod/
  - Enabling GPUs in the Container Runtime Ecosystem
    https://devblogs.nvidia.com/gpu-containers-runtime/
- Datasets and benchmarks
  - Credit card dataset
    https://www.kaggle.com/mlg-ulb/creditcardfraud

    https://blogs.oracle.com/datascience/fraud-detection-using-autoencoders-in-keras-with-a-tensorflow-backend
  - GANs
    https://skymind.ai/wiki/generative-adversarial-network-gan
    https://github.com/eriklindernoren/Keras-GAN/blob/master/gan/gan.py

    https://www.geeksforgeeks.org/generative-adversarial-network-gan/
  - Financial fraud detection under IoT
    https://www.hindawi.com/journals/scn/2018/5483472/
  - TensorFlow benchmarks
    https://github.com/tensorflow/benchmarks
  - Auto encoders for fraud detection
    https://www.datascience.com/blog/fraud-detection-with-tensorflow

Version History

Version	Date	Document Version History
Version 1.0	December 2019	Initial release

Refer to the Interoperability Matrix Tool (IMT) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer's installation in accordance with published specifications.

Copyright Information

Trademark Information

TR-4807-1219