Distributed Training of Deep Autoencoder for Network Intrusion Detection

Haripriya C[1], Prabhudev Jagadeesh M.P[2]
Research Scholar, JSS Academy of Technical Education, Bengaluru, Affiliated to VTU Belagavi, India[1]
Assistant Professor, Global Academy of Technology, Bengaluru, Affiliated to VTU Belagavi, India[1]
Professor, JSS Academy of Technical Education, Bengaluru, Affiliated to VTU Belagavi, India[2]

***Abstract*—The amount of data being exchanged over the internet is enormous. Attackers are finding novel ways to evade rules, investigate network defenses, and launch successful attacks. Intrusion detection is one of the effective means to counter attacks. As the network traffic continues to grow, it can be challenging for network administrators to detect intrusions. In huge networks connected with millions of computers Terabytes/Zettabytes of data is generated every second. Deep Learning is an effective means for analyzing network traffic and detecting intrusions. In this article, distributed autoencoder on the CSE-CIC-IDS2018 dataset is implemented by considering all the classes of the dataset. The proposed work is implemented on Azure Cloud using distributed training as it helps in speeding up the training process, thereby detecting intrusions faster. An overall accuracy of 98.96 % is achieved. By leveraging such parallel computing into the security process, organizations may accomplish operations more quickly and respond to risks and remediate them at a rate that would not be possible with manual human capabilities alone.**

Keywords*—Network intrusion detection systems; deep learning; autoencoders; cloud computing; distributed training; parallel computing

I. INTRODUCTION

Organizations must prioritize cybersecurity as they begin their digital transformation initiatives. The availability, confidentiality and integrity of the data has to be maintained irrespective of whether the deployment was on the premises or on the cloud. Therefore, while organizations want to ensure that they keep up with technological changes, they also need to prioritize security. In a matter of seconds, the amount of exchanged data can multiply dramatically, increasing the risk to sensitive data. And as the Internet of Things (IoT) expands beyond consumer electronics and into industrial machinery, there is concern over both the number of attacks and how sophisticated they are becoming. Human welfare is at risk as attackers are targeting hospitals, cars, and power plants. Given the stakes, it is crucial that IT directors put the practical approach in place to protect this valuable data. Security managers are implementing significant changes as they embrace new technology and security measures. Rather than making systems secure, the emphasis is on ensuring that the data they contain is secure. As a result of digital transformation, attackers are exploring vulnerabilities to launch attacks. Today, any vulnerability in the supply chain has a catastrophic effect that costs millions of dollars thus, immediately destroying the credibility.

Digitalization, connected cars, connected homes and IoT has enhanced the digital footprint. The world is transitioning from a physical to a digital experience. The ever increasing threat landscape brings in more challenges in the security domain. This led to the integration of Artificial Intelligence (AI) and Machine Learning (ML) in cyber security. To protect their organizations from novel attacks, companies are increasing their investments in cyber security automation. Few decades ago, companies used to invest very less on security. With the increase in the incidents and breaches post the covid pandemic period, the world has transitioned into a remote network which operates without geographical boundaries. Attackers are finding new ways to breach the security, posing significant threat to organizations. Deep Learning (DL) models can leverage cloud computing services. This paradigm shift helps train DL algorithms on distributed hardware more easily.

DL has been quite a game-changer for several companies and organizations during the recent years. As a result, it is not surprising that many specialized, cloud-based solutions have developed to aid data scientists in their work multiple ways. The global ML market is projected to grow from $7.3 billion in 2020 to $30.6 billion in 2024. This would lead to a substantial growth of 43% according to Forbes. To continue with the constantly changing business requirements of consumers, data scientists and ML engineers have to create more sophisticated models. ML experts have found MLaaS to be extremely useful and powerful in designing more sophisticated models.

Organizations have to leverage DL techniques to quickly identify, evaluate and treat risks to evade major attacks on their networks. The consequences after a network attack are devastating. Attacks might be one or more of the following;

- Loss to government, private and public parties
- Legal and regulatory impact
- Financial implications
- Impact to essential services
- Loss of trust
- Socio-psychological impact.

Decades ago, computing was completely self-managed with respect to hardware, software and configuration. The location of the data center was dedicated to a physical location and customization was based on requirements. Ring fence security was used to block or control external access. These

techniques are no longer suited in the present digital era as they are expensive because of the capital and operational costs. Also, the complexity with customization has increased. Another major disadvantage of traditional computing is their inability to scale. Thus, there is a complete paradigm shift towards cloud computing.

Traditional methods of storage are no longer suitable because of the following requirements.

- Exponentially growing data.
- Very expensive, not scalable and slow
- Loss of data.

Compared to traditional storage methods, there are numerous advantages of storing data on the cloud.

- Ease of use and access
- Easy sharing
- Disaster recovery
- Scalability and elasticity
- Cost efficient
- Security

There are numerous advantages of cloud computing over traditional computing

- Shared infrastructure that are logically separated.
- No access to the underlying hardware.
- Optional customization of software and configuration.
- Geographically dispersed points of presence
- Suited for performance and scalability.
- As-a-service (aas) subscription model.

MLaaS refers to a group of different cloud-based systems that combine ML tools with solutions to assist ML team in various tasks such deploying and run orchestration models, data pre-processing, model training and tuning predictive analysis for a variety of problem statements. It leverages cloud computing's flexibility to provide ML services on the go. The MLaaS industry is pretty substantial. Valued at $1.0 billion in 2019, it is anticipated to grow to $8.48 billion by 2025. Azure Machine Learning Studio offers a development environment. It helps create ML models both for entry-level and professional data scientists. Most actions in Azure ML Studio may be accomplished using a GUI interface, just like with Microsoft Windows.

The key contribution of this research is to perform distributed training. This research article emphasizes on the fact that training time can be drastically reduced when distributed training is used. As the size of the dataset increases, the traditional methods of training DL models using single machine are no longer suitable. In the context of NIDS, the main aim is to help the network administrators to detect intrusions at a faster rate. In terms of training time, the proposed distributed autoencoder model improves the performance by reducing the training time. This helps the network administrators to take necessary steps before the attacker is successfully able to launch an attack on the network.

The manuscript is organized as follows. Section II gives a detailed description of the literature review carried in the area of Network Intrusion Detection (NIDS). Section III gives the details of the proposed methodology. Section IV gives the details about the experimental setup. Section V gives the details on the results. Finally, Section VI discusses about conclusion, limitations and future enhancements.

II. Related Work

Ketulkumar et al. implemented ‘Multiclass Decision Forest’ on the UNSW-NB15 dataset using. The ML algorithm was run on Azure ML platform. The author has achieved an average overall accuracy of 96.33% [1].Youngrok et al. carried out their research on three datasets namely NSL-KDD, N-BaTot and IoTID20 datasets by using auto encoders. The authors opined that stacked encoders work better when their model size is increased [2]. Pooja Rana et al. implemented their FCM-ANN and SVM-ANN and FCM-SVM, SVM-ANN algorithms. The authors concluded that FCM-SVM methodology outperforms other classifiers on the UNSW_NB15 dataset. SVM-ANN methodology outperforms other classifiers on the NSL-KDD dataset [3]. Kanimozhi et al. implemented various ML algorithms and ANN (MLP) on the CSE-CIC 2018 dataset. The authors concluded that ANN (MLP) outperforms other ML Classifiers. Using ANN (MLP) the authors achieved an accuracy of 99.97% along with other metrics. The authors also used a Calibration curve [4]. Smitha Smitha Rajagopal et al. proposed a meta-classification approach. The authors tested their proposed model on UNSW NB-15, CICIDS 2017 and CICDDOS 2019 datasets [5]. Jan Lasky et al. carried out an extensive literature review on NIDS and concluded that DL techniques outperform shallow ML techniques [6].

Kanimozhi et al. used the combination of Random Forest and Decision Tree classifiers. The authors used ANN algorithm on the UNSW-NB15 dataset with an accuracy of 89% [7]. Zeeshan Ahmed et al carried out an extensive literature survey on ML and DL approaches used in NIDS. The authors conclude that a majority of the researchers detested their models on outdated datasets like KDD cup’99 and NSL-KDD. Another important finding from their work is most of the researchers have not addressed the class imbalance problem in their datasets. This affects the accuracy and detection rate of the minority attack classes [8]. Satish et al. carried out a detailed review of research trends in NIDS. The authors conclude that KDD Cup’99 is the most used dataset for NIDS. However, KDD Cup’99 does not reflect the current attacks. The authors encourage researchers to carry out their research using datasets that reflect the current day attacks [9]. Narayana et al implemented the Stacked Auto encoded- Deep Neural network (SAE-DNN) on KDD, NSL-KDD and UNSW-NB15 datasets. Their hybrid model on UNSW-NB 15 achieved an accuracy of 99.5% [10]. Sultan Zavrak et al. used Variational Auto Encoder (VAE). The metrics used by authors were Receiver Operating Characteristics (ROC) and Area Under ROC curve [11]. Sydney Mambwe et al. used Simple RNN,

LSTM and GRU on NSL-KDD and UNSW-NB15 datasets [12].

Zichan Ruan et al. used visualization algorithm to gain insights into the KDD99 cup dataset [13]. Matthias Langer et al give a taxonomy of Distributed Deep Learning Systems (DDLS) [14]. Asif et al. in their research paper detailed about the strategic importance of NIDS [15]. Abhishek Divekar et al. in their research work conclude that UNSW-NB 15 dataset is an alternative to KDDCup 99. The authors also highlighted that class imbalance of KDD-99 and NSL-KDD. Class imbalance hampers the efficacy of the classifiers on the minority class [16]. Mahdi Soltani et al. proposed model on Deep Intrusion Detection (DID) system. The authors evaluated their work on CIC-IDS2017 and CSE-CIC-IDS2018 datasets [17]. Joffrey L Leevy et al. carried out a detailed survey of IDS models on CSE-CIC-IDS2018 dataset. The authors opined that most of the researchers who carried out their work on CSE-CIC-2018 dataset did not address the class imbalance [18]. Sukhpreet et al. used eXtreme Gradient Boosting (XGBoost) on the NSL-KDD dataset. Jiyeon Kim et al used Convolutional Neural Network (CNN) and focussed only on DoS (Denial of Service) attacks. They used the KDD and CSE-CIC-IDS 2018 datasets [20]. Said Ouiazzane et al proposed snort signature based NIDS on the CICIDS2017 dataset. They implemented their proposed model using ML algorithms [21]. Haripriya et al carried out a literature review on NIDS datasets. They implemented Deep Autoencoder by including all the files of CSE-CIC-IDS2018 dataset. An overall accuracy of 97.79% was achieved [22-23].

Attackers are finding novel ways to launch sophisticated attacks. Studies from literature review show that there is an increasing necessity to not only detect intrusions accurately but also more quickly. Once intrusions on the network are detected, necessary steps should be taken so that the attacker is not successfully able to launch the attack. Considering the enormous amount of data generated on the network, training the DL algorithm on a single machine is no longer suitable. This research work mainly focuses on distributed training of deep autoencoder model to speed up the training.

III. Proposed Methodology

There are two means to achieve parallelism namely model parallelism and data parallelism. Model parallelism is when the same data is used for each thread but the model is split among them. In data parallelism, same model is used for each thread but operating on different portions of the data.

This research article focuses on data parallelism. Fig. 1 illustrates data parallelism [24]. Initially, a compute cluster with two nodes is created. Each node contains a replica of the model. However, each node processes a different portion of the data. The errors between each node's predictions for its training samples and the labeled outputs are individually calculated. Each node then modifies its model in response to the errors. This information is communicated to all the other nodes to update their related models. The worker nodes need to synchronize model parameters on every batch computation. This ensures the consistency of the training model.

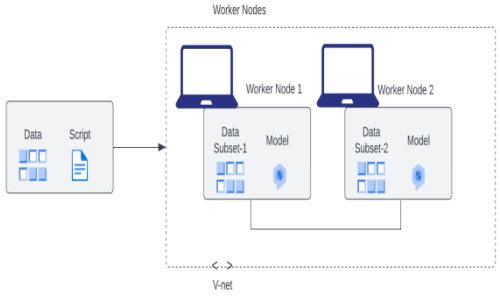

Fig. 1. Data parallelism.

A. Autoencoders

The proposed work uses autoencoders. An autoencoder is a special type of neural network architecture, used for unsupervised learning. The main idea of using an autoencoder is to learn a lower-dimensional representation for a higher-dimensional data.

B. CSE-CIC-IDS2018 Dataset

CSE-CIC-IDS 2018 dataset is used on the proposed autoencoder model [25]. The dataset is obtained from AWS S3 bucket. It consists of seven types of attack scenarios along with 80 features. The dataset was created as joint venture between the Communications Security Establishment (CSE) and Canadian Institute of Cybersecurity (CIC).

C. Preprocessing

Preprocessing helps network traffic data to be easily processable by the DL algorithm. It also helps speeding up the training process. Rows containing NAN and infinite values were dropped. Label encoding and one-hot encoding were used. The CSE-CIC-IDS2018 dataset suffers from class imbalance. The classifier tends to be more biased towards the majority class leading to inaccurate results. This has a major negative impact on the performance. The class imbalance was addressed Synthetic Minority Oversampling Technique (SMOTE). It is an augmentation technique for the minority class.

D. Configuration in Azure Machine Learning Studio

Fig. 2 illustrates the Azure ML Resource group. The following steps were carried out.

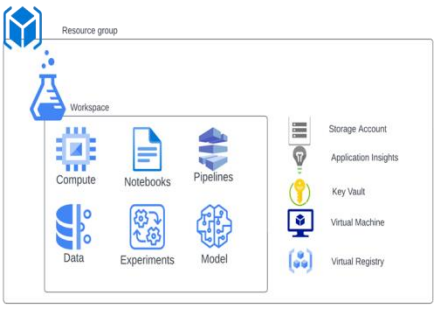

Fig. 2. Azure ML resource group.

- Sign in to Azure Machine Learning Studio and select create workspace. Give the workspace name. In addition to this provide subscription type, resource group and the region. After providing all the details a workspace is created.
- Microsoft Azure Blob (Binary Large Objects) was employed to store CSECICIDS 2018 dataset. Blob storage is best suited for large scale unstructured data.

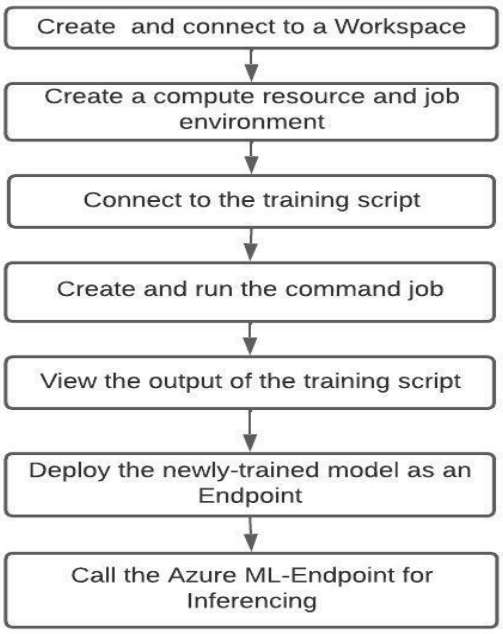

Fig. 3. Flowchart depicting model deployment on Azure cloud.

Fig. 3 clearly illustrates the different steps to be followed while deploying the DL model on the cloud.

IV. Experimental Setup

A. Details on the Implementation

Fig. 4 illustrates accelerated networking. The configured Virtual Machines (VMs) supports accelerated networking. It greatly improves network performance by reducing latency, jitter and CPU utilization. This is suitable in scenarios with most demanding network workloads.

Premium Solid State Drive (SSD) disks were used for storage. The main advantage of using SSD is that they have access speeds of 35 to 100 microseconds. Compared to Hard Disk Drives (HDD) they are 25 to 100 times faster. SSDs are more suitable in I/O intensive workloads and deliver high-performance and low-latency disk support for VMs.

Locally Redundant Storage (LRS) was used as illustrated in Fig. 5. The key advantage of using LRS is, it replicates the storage account three times within a single data center located in the primary region. Thus, the application is restricted to replicate data only within a country/ region. This is more suitable in scenarios where data governance requirements are being imposed. Table I illustrates the configuration of the VMs used in the training model.

Synchronous distributed training is implemented across the worker nodes, each having two CPUs. All variables and computations are replicated to every local device. In order to enable collaboration of multiple workers, it utilizes a distributed collective implementation (such as all-reduce).

First the ScriptRunConfig is created. This is used to specify the training script arguments, the environment and the cluster to run on. The training script in this experiment uses a Multi-Worker Distributed training of the Keras model. This can be done using the tf.distribute Strategy API. tf.distribute.Experimental.MultiWorkerMirroredStrategy() is used to leverage distributed training. The tensor flow configuration is used to run a multi-worker tensor flow job. The number of nodes in the training job is specified by setting the worker count variable. In tensor flow, to enable the training on multiple machines, the TF_CONFIG environment variable is used. This allows the training code on each Virtual Machine (VM) instance used in the training job, to gain access to information about the training job and the VM's operation. Thus, to comply with the requirements set forth by Tensor Flow for distributed training and to enable communication between VMs, TF_CONFIG environment variable must be present on all the VMs configured for the training job. Next, a distributed config is created by specifying the number of worker count. The number of worker nodes is set to two. The train - test split is set is to 70 % and 30 % respectively.

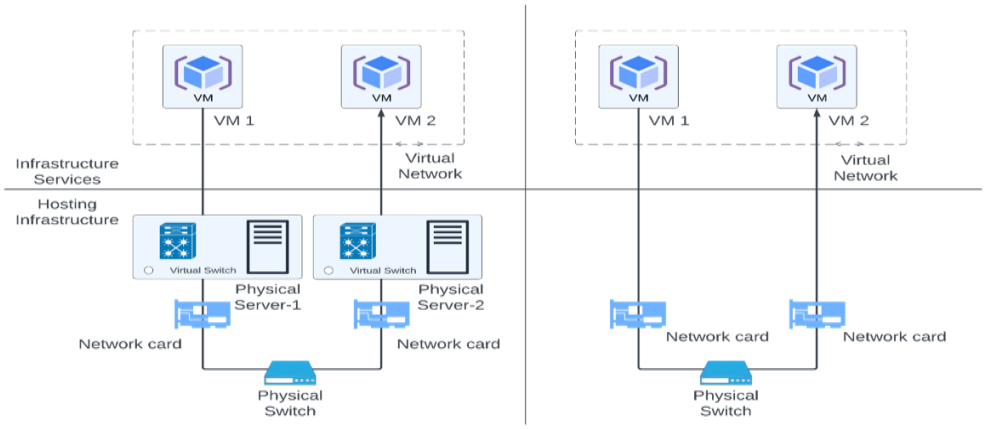

Fig. 4. Accelerated networking.

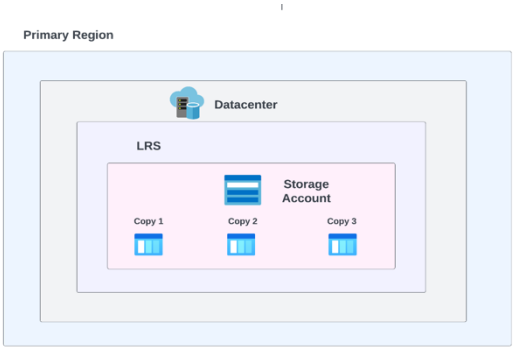

Fig. 5. Replication of storage account using LRS.

TABLE I. VM CONFIGURATION - DV2 11SERIES

VM Configuration	VM 1	VM 2
Size	Standard_D11_v2	Standard_D11_v2
vCPU	2	2
Memory GiB	14	14
Temporary Storage(SSD)GiB	100	100
Max temp storage throughput:IOPS/ ReadMBps / Write MBps	6000/93/46	6000/93/46
Maxdatadisks/ throughput IOPS	8/8*500	8/8*500
Max NICs	2	2
Expected network bandwidth (MBps)	1500	1500

TABLE II. HYPERPARAMETERS USED IN THE PROPOSED DISTRIBUTED DEEP AUTOENCODER

Sl. No.	Hyper Parameters	Value
1	Activation Function: Hidden layer	ReLU
	Output layer	Softmax
2	Batch Size	128
3	Number of Epochs	15
4	Loss function: Multi Classification	Categorical Cross Entropy
5	Optimizer	Adam
6	Learning Rate	0.01

Table II gives the details of hyper tuning parameters in training the proposed distributed autoencoder model.

V. RESULTS AND DISCUSSION

In our experimentation two scenarios are considered. In the first scenario, single VM is used for training. In the second scenario, two VMs are used to train the model.

Fig. 6 illustrates the time taken when the autoencoder model is run on a single VM without distributed training. Fig. 7 illustrates the time taken when the autoencoder model is run on two VMs by leveraging distributed training. Time taken to train is plotted on the X- axis and CPU Utilization is plotted on the Y-axis. In this research work, by using data parallelism, Deep Auto-encoder algorithm is trained on two VMs (distributed training), which speeds up the training process. An overall accuracy of 98.96% is achieved.

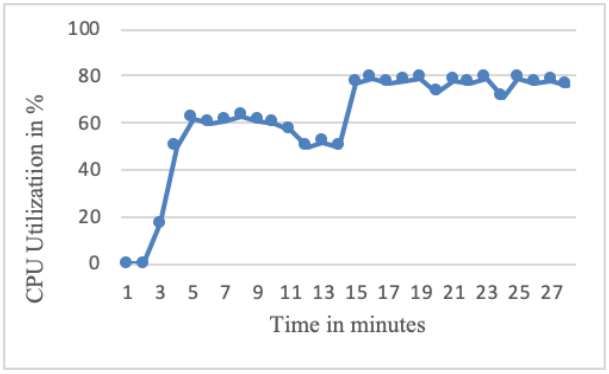

Fig. 6. CPU utilization and Time taken when a single node is used.

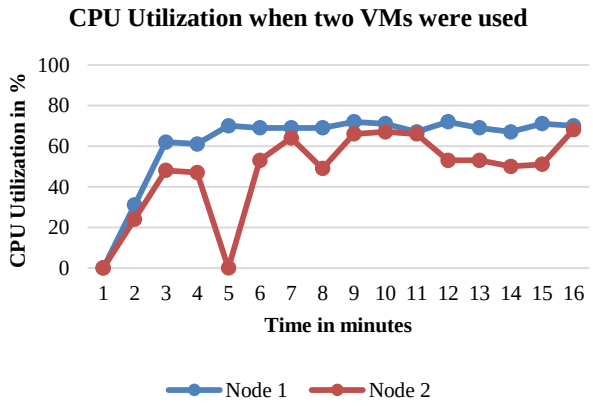

Fig. 7. CPU utilization and time taken when multiple nodes are used.

Fig. 8 depicts the training time when single node and distributed training is used. Time taken to train when single VM is used is 67 minutes while the time taken to train when two VMs (Distributed Training) is 43 minutes. 24 minutes reduction in terms of training time was observed when distributed training was used. Table III gives the comparative analysis of the proposed work with other techniques.

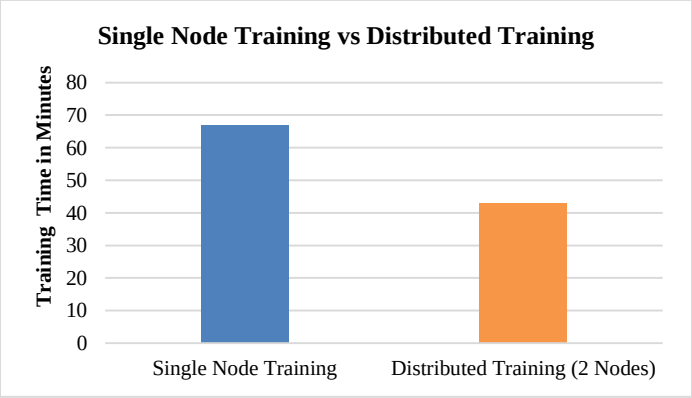

Fig. 8. Single node training vs. distributed training.

TABLE III. COMPARATIVE ANALYSIS

Sl.No	Authors	Algorithm Used	Dataset Used	Accuracy	Year	Attacks detected	Whether Distributed training was used?
1	V. Kanimozhi et al [4]	Artificial Neural Network and Multi-Layer Perceptron	CSE-CIC-IDS 2018	99.97%.	2019	Only Botnet attack	No
2	Alzughaibi et al [26]	Multi-Layer Perceptron with Back Propagation	CSE-CIC-IDS 2018	98.41 %	2023	All the attacks of the dataset	No
3	Farhan et al [27]	Deep Neural Network	CSE-CIC-IDS 2018	90 %	2020	All the attacks of. the dataset	No
4	Proposed Work	Deep Auto Encoder	CSE-CIC-IDS 2018	98.96 %	2023	All the attacks of the dataset	Yes

VI. CONCLUSION

With the pace at which internet is growing, the amount of data exchanged over it is increasing, paving way for novel network attacks. Detecting intrusions early to avoid network attacks is the need of the hour. Considering the huge size of dataset, classifying all the attacks is a compute-intensive task. The main contribution of this research work, is to perform distributed training on the autoencoder model using the latest benchmark dataset by classifying all the classes of the dataset. Performance on the model proved to more efficient when distributed training was used. For compute and data intensive tasks, DL combined with distributed training is the most appropriate solution. Promising results were achieved with an overall accuracy of 98.96% and 24 minutes reduction in training time when distributed training was used. To the best of our knowledge, this is indeed the first research work done in this area. As a future work, other DL algorithms suitable for distributed training for NIDS can be explored. The training time can further be reduced by using high configuration VMs. Also, services from various cloud service providers can be leveraged for distributing training. A comparative study of various ensemble methods for DL algorithms suitable for NIDS can also be carried out.

REFERENCES

[1] Chaudhari, Ketulkumar. (2018). Cyber Attack Classification in Microsoft Azure Using Deep Learning Algorithm. International Journal of Advanced Research in Electrical Electronics and Instrumentation Engineering. 7. 8168-8171. 10.2139/ssrn.

[2] Song Y, Hyun S, Cheong Y-G. Analysis of Autoencoders for Network Intrusion Detection. Sensors. 2021; 21(13):4294. https://doi.org/10.3390/s21134294

[3] Rana, Pooja & Batra, Isha & Malik, Arun & Imoize, Agbotiname & Kim, Yongsung & Pani, Subhendu & Goyal, Nitin & Kumar, Arun & Rho, Seungmin. (2022). Intrusion Detection Systems in Cloud Computing Paradigm: Analysis and Overview. Complexity. 2022. 10.1155/2022/3999039.

[4] Kanimozhi, V. & Jacob, Prem. (2020). Artificial Intelligence outflanks all other machine learning classifiers in Network Intrusion Detection System on the realistic cyber dataset CSE-CIC-IDS2018 using cloud computing. ICT Express. 7. 10.1016/j.icte.2020.12.004.

[5] Rajagopal, Smitha & Kundapur, Poornima & S., Hareesha. (2021). Towards Effective Network Intrusion Detection: From Concept to Creation on Azure Cloud. IEEE Access. PP. 1-1. 10.1109/ACCESS.2021.3054688.

[6] Lansky, Jan & Ali, Saqib & Mohammadi, Mokhtar & Majeed, Mohammed & Karim, Sarkhel & Rashidi, Shima & Hosseinzadeh, Mehdi & Rahmani, Amir. (2021). Deep Learning-Based Intrusion Detection Systems: A Systematic Review. IEEE Access. 9. 101574-101599. 10.1109/ACCESS.2021.3097247.

[7] Kanimozhi, V. & Jacob, Prem. (2019). UNSW-NB15 dataset feature selection and network intrusion detection using deep learning. International Journal of Recent Technology and Engineering. 7. 443-446.

[8] Ahmad, Zeeshan & Shahid Khan, Adnan & Shiang, Cheah & Ahmad, Farhan. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Transactions on Emerging Telecommunications Technologies. 32. 10.1002/ett.4150.

[9] Kumar, Satish & Gupta, Sunanda & Arora, Sakshi. (2021). Research Trends in Network-Based Intrusion Detection Systems: A Review. IEEE Access. PP. 1-1. 10.1109/ACCESS.2021.3129775.

[10] K. Narayana Rao, K. Venkata Rao, Prasad Reddy P.V.G.D., A hybrid Intrusion Detection System based on Sparse autoencoder and Deep Neural Network, Computer Communications,Volume 180, 2021, Pages 77-88, ISSN 0140-3664, https://doi.org/10.1016/j.comcom.2021.08.026.

[11] Zavrak, Sultan & Iskefiyeli, Murat. (2020). Anomaly-Based Intrusion Detection From Network Flow Features Using Variational Autoencoder. IEEE Access. 8. 108346-108358. 10.1109/ACCESS.2020.3001350.

[12] Sydney Mambwe Kasongo. 2023. A deep learning technique for intrusion detection system using a Recurrent Neural Networks based framework. Comput. Commun. 199, C (Feb 2023), 113–125. https://doi.org/10.1016/j.comcom.2022.12.010

[13] Ruan, Zichan & Miao, Yuantian & Pan, Lei & Patterson, Nicholas & Zhang, Jun. (2017). Visualization for big data security — A case study on KDD99 cup data set. Digital Communications and Networks. 3. 10.1016/j.dcan.2017.07.004.

[14] Langer, M., He, Z., Rahayu, W., & Xue, Y. (2020). Distributed training of deep learning models: A taxonomic perspective. IEEE Transactions on Parallel and Distributed Systems, 31(12), 2802-2818.

[15] Asif, Muhammad & Khan, Talha & Taj, Talha & Naeem, Umar & Sufyan, Muhammad. (2013). Network Intrusion Detection and its strategic importance. BEIAC 2013 - 2013 IEEE Business Engineering and Industrial Applications Colloquium. 140-144. 10.1109/BEIAC.2013.6560100.

[16] Divekar, Abhishek & Parekh, Meet & Savla, Vaibhav & Mishra, Rudra & Shirole, Mahesh. (2018). Benchmarking datasets for Anomaly-based Network Intrusion Detection: KDD CUP 99 alternatives.

[17] Soltani, Mahdi & Jafari Siavoshani, Mahdi & Jahangir, Amir. (2022). A Content-Based Deep Intrusion Detection System. International Journal of Information Security. 21. 1-16. 10.1007/s10207-021-00567-2.

[18] Leevy, Joffrey & Khoshgoftaar, Taghi. (2020). A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 Big Data. Journal of Big Data. 7. 10.1186/s40537-020-00382-x.

[19] Dhaliwal, Sukhpreet & Nahid, Abdullah & Abbas, Robert. (2018). Effective Intrusion Detection System Using XGBoost.

[20] Kim, Jiyeon & Kim, Jiwon & Kim, Hyunjung & Shim, Minsun & Choi, Eunjung. (2020). CNN-Based Network Intrusion Detection against Denial-of-Service Attacks. Electronics. 9. 916. 10.3390/electronics9060916.

[21] Said Ouiazzane, Malika Addou and Fatimazahra Barramou, "A Multiagent and Machine Learning based Hybrid NIDS for Known and Unknown Cyber-attacks" International Journal of Advanced Computer Science and Applications(IJACSA), 12(8), 2021.

[22] Haripriya C, Prabhudev Jagadeesh M. P. "A Review of Benchmark Datasets and its Impact on Network Intrusion Detection Techniques," 2022 Fourth International Conference on Cognitive Computing and Information Processing (CCIP), Bengaluru, India, 2022, pp. 1-6, doi: 10.1109/CCIP57447.2022.10058660.

[23] Haripriya C, Prabhudev Jagadeesh M. P (2022). An Efficient Autoencoder Based Deep Learning Technique to Detect Network Intrusions. International Transaction Journal of Engineering, Management, & Applied Sciences & Technologies, 13(7), 13A7P, 1-9. http://TUENGR.COM/V13/13A7P.pdf DOI: 10.14456/ITJEMAST.2022.142

[24] https://learn.microsoft.com/en-us/dotnet/standard/parallel-programming/data-parallelism-task-parallel (Acccessed on 02.01.2023)

[25] A Realistic Cyber Defense Dataset (CSE-CIC-IDS2018) was accessed on 03.01.2023 from https://registry.opendata.aws/cse-cic-ids2018.

[26] Alzughaibi, S.; El Khediri, S. A Cloud Intrusion Detection Systems Based on DNN Using Backpropagation and PSO on the CSE-CIC-IDS2018 Dataset. Appl. Sci. 2023, 13, 2276. https://doi.org/10.3390/app13042276

[27] Farhan, Rawaa & Maolood, Abeer & Hassan, Nidaa. (2020). Performance analysis of flow-based attacks detection on CSE-CIC-IDS2018 dataset using deep learning Deep learning Flow-based intrusion detection Internet of thing (IOT). Indonesian Journal of Electrical Engineering and Computer Science. 20. 1413-1418. 10.11591/ijeecs.v20.i3.pp1413-1418.