

A Clustering Approach for Intrusion Detection with Big Data Processing on Parallel Computing Platform

B. REIS, S.B. KAYA, and O.K. SAHINGOZ*


Abstract— In recent years there is a growing number of attacks in the computer networks. Therefore, the use of a prevention mechanism is an inevitable need for security admins. Although firewalls are preferred as the first layer of protection, it is not sufficient for preventing lots of the attacks, especially from the insider attacks. Intrusion Detection Systems (IDSs) have emerged as an effective solution to these types of attacks. For increasing the efficiency of the IDS system, a dynamic solution, which can adapt itself and can detect new types of intrusions with a dynamic structure by the use of learning algorithms is mostly preferred. In previous years, some machine learning approaches are implemented in lots of IDSs. In the current position of artificial intelligence, most of the learning systems are transferred with the use of Deep Learning approaches due to its flexibility and the use of Big Data with high accuracy. In this paper, we propose a clustered approach to detect the intrusions in a network. Firstly, the system is trained with Deep Neural Network on a Big Data set by accelerating its performance with the use of CUDA architecture. Experimental results show that the proposed system has a very good accuracy rate and low runtime duration with the use of this parallel computation architecture. Additionally, the proposed system needs a relatively small duration for training the system

Index Terms— Deep Neural Network, Big Data, CUDA, GPU, Parallel Computation.


I. INTRODUCTION

DUE to the recent improvements in networking technologies, the global network, the Internet, increase its popularity and used in most of the real work activities. At the

SAMI BERK KAYA is with Istanbul Kultur University Computer Engineering Department, Istanbul, Turkey, (e-mail: saamiberk@gmail.com).

 <https://orcid.org/0000-0001-6887-9777>

BUMINHAN REIS, is with Istanbul Kultur University Computer Engineering Department, Istanbul, Turkey, (e-mail: buminhanreis@gmail.com).

 <https://orcid.org/0000-0002-7221-277X>

OZGUR KORAY SAHINGOZ, is with Istanbul Kultur University Computer Engineering Department, Istanbul, Turkey, (e-mail: o.sahingo@iku.edu.tr).

 <https://orcid.org/0000-0002-1588-8220>

* Corresponding Author

Manuscript received May 10, 2019; accepted June 17, 2019.
DOI: [10.17694/bajece.563167](https://doi.org/10.17694/bajece.563167)

same time, the number of hacking activities grows and directly affect the performance of the network [1]. The hackers/attackers aim to access the network in an unauthorized way and try to change some data in the network or sniff the communication between other networks, disable some network services during a specific time. Therefore, protecting networks against different types of attacks becomes an important research area not only for the network security managers but also for many scholars

Traditionally use of some antivirus programs, network authentication tools or firewalls are preferred for protecting the systems from these attacks [2]. Although they decrease the number of attacks, they are not enough for whole protection. In this point, intrusion detection and prevention systems emerged as an alternative solution for security threats. An intrusion detection mainly analyzes the whole networks activities, especially from the data communication view, and tries to catch the unauthorized and abnormal ones. In case of catching this type of intrusion, the intrusion detection system needs to create an alert message for the network security admins to get necessary precautions. However, the intrusion prevention system can automatically execute some services to prevent attacks such as blocking the IP addresses from which the intrusions occur.

This type of intrusion detection and prevention system firstly needs to identify intrusions, and this can be done with two different ways as signature-based detection and anomaly-based detections. In the former one system maintains and intrusion database which stores the patterns of the known attacks. If an incoming request/message match with the database, it is automatically labeled as an intrusion. Although this method is preferred in most of the commercial IDS systems, and it has very low false alarm rates. However, due to the static structure of the signature-based approach, systems can miss most of the unknown attacks, such as zero-day attacks. Additionally, intruders can evaluate their attack types according to created signatures. Therefore, signature-based systems are fragile in this manner.

On the other side, anomaly-based systems react to anomalous behaviors in the networks. According to the previous history of the monitored systems, if there are some anomalous messaging, they are labeled as intrusions. These systems can easily detect zero-day attacks due to its dynamic structure. However, for defining the normal behavior of the

system, huge training time is needed with a great data set which is not much suitable for traditional machine learning techniques [3].

In recent years, deep learning approaches are highly preferred in the processing of Big Data. These techniques result in better accuracy rates and low false positive rates. Therefore, in this paper, it is aimed to use a deep neural network approach for training the intrusion detection system. To increase the accuracy rate of the system a clustered approach is preferred and to decrease the training time, parallel execution is preferred. The details of the algorithmic design of the frameworks and obtained test results are presented in the ongoing sections.

The rest of the paper is organized as follows. In Section 2 the related works are surveyed. The proposed system's details with the clustering approach and the Experimental result of the system are presented in Section 3 and Section 4 respectively. Finally, conclusions and future works are drawn.

II. RELATED WORKS

In previous studies, different types of intrusion detection systems have been implemented. Each of them uses different approaches to increase the accuracy rate of the system. The trend of these works shows that's, current systems mainly focus on the use of machine learning approaches for setting up dynamic detection system. In some studies, Java based WEKA libraries, which are especially useful for different artificial intelligence problem solutions [4], are preferred, some works are focused on the use of Tensorflow libraries with Python programming language.

IDS can be applicable in many different application areas. In one of the current researches[5] authors focused on the security operations of smart grids. In this manner they mainly analyzed the 37 different cases in the intrusion detection and prevention systems concepts in smart grid environments

One of the recent works is presented by Farahnakian and Heikkonen in 2018, and they prefer the used of deep learning approach for IDSs with the use of Deep Auto-Encoder model. As a dataset, they use the KDDCUP'99 and use an unsupervised learning algorithm to avoid overfitting [6].

In [7], Spaffard et. Al. uses autonomous agents for intrusion detection and shortened the system as AAFID. AAFID has a distributed architecture; agents work as autonomous entities to monitor the message traffic in the host side. Due to its distributed structure, there is not a central authority to generate an alarm, and agents do not communicate with each other directly.

Sagha et al. proposed a real-time learning methodology for their IDS system with the use of reinforcement learning mechanism [8]. Their learning engine uses a fuzzy modeling technique and temporal difference learning which is a prediction method which estimates the status of the messages according to previous knowledge. They also used a Reward-Penalty Plane (RPP)structure to show the efficiency of the actions in a specific state.

An alternating decision tree approach is used in [9] for the

classification of intrusions. They used NSL-KDD dataset for testing their system and about 98 % accuracy rate is reached in the experiments. They planned to use some evolutionary algorithms to increase the efficiency of the system. The system also used a clustering approach. They clustered the dataset into four different clusters as DOS, Probe, R2L, and U2L. After that, they make a partition about each cluster into training and test sets.

At the same time, some hybrid detection mechanisms are also preferred in the literature. Desai mainly uses some signature-based matching algorithms, and for increasing efficiency, a fuzz genetic algorithm is adapted to the system [10]. The signature matching algorithm is used for identifying the inner attacks. However, the fuzzy genetic algorithm is preferred for detection of the external attacks. Due to the structure of the system, it can be executed either in offline or online environments.

A neural network-based approach was proposed in [11] with the use of single hidden layer with different number of neurons. In this paper, authors analyzed KDDCup99 Dataset and they claimed to reach acceptable accuracy rates in a shorter time depending on the attack types. In the same neural network-based approach, the effect of the training functions was examined in [12]. They showed that the training function can highly change the accuracy rate of the systems and "trainlm" function results the best between the compared functions.

III. PROPOSED SYSTEM

This section gives details about the proposed system. We initially discuss the NSL-KDD data set and the related stages of pre-processing and feature selection techniques. Later, we introduce the K-Means algorithm for partitioning the data set into clusters of train and test set pairs based on similarity. Finally, with the addition of the K-Means algorithm, we present a hybrid IDS with the ensemble of a GPU-accelerated Deep Neural Network (DNN) classifier. of the proposed system's framework is shown in Fig.1.

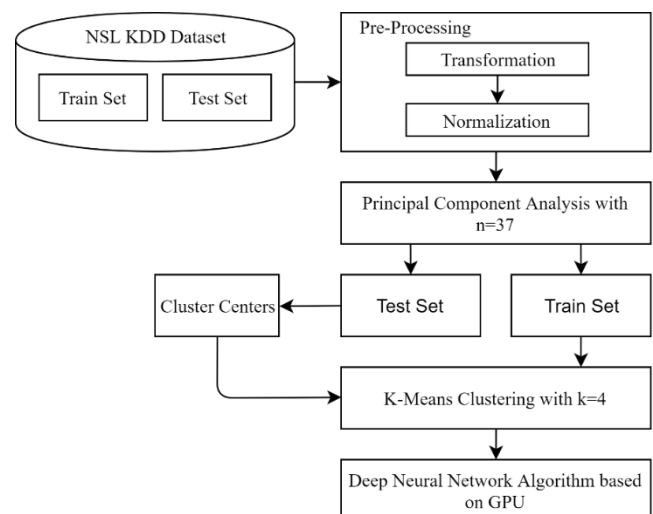


Fig.1. The Framework of the Proposed System

A. Dataset

The NSL-KDD dataset is a successor of DARPA’s KDD’99 dataset which was obtained from the previous 1998 DARPA intrusion detection evaluation program. The environment created in this program was aimed to acquire raw TCP/IP data for a Local Area Network (LAN). Every TCP/IP connection was represented with 41 various features [13]. Although it has been noted that this newer version of the KDD data set has some flaws as pointed out in [14], it is also a good candidate as a benchmark dataset in the Intrusion Detection field. The NSL-KDD data set has a total of 42 features, which 41 of them being the attributes that define the characteristics of the records.

In NSL-KDD data set, the total number of different attack types is 39. All the attacks can be grouped into four main categories as DoS, Probe, R2L, and U2R. While the NSL-KDD data set contains a total of 39 different attack types, the number of unique attack types contained in the train set and test set differentiate from each other. The attack types contained in the train set which are called as “known attacks” have a total number of twenty-two attack types. On the other hand, the test attacks have a total number of thirty-seven attacks. The additional attacks in the test set which are not present in the train set are called “novel attacks.” Having unknown attack types in the test set shows if the system can adapt to different scenarios when the testing is done. The attack types contained in the train and test sets are shown in Table 1 with the novel attacks being represented as bold.

TABLE I
ATTACK TYPES CONTAINED IN THE TRAIN AND TEST SETS OF NSL-KDD

Attack Groups	Attack Types
DOS	worm, processtable, mailbomb, pod, apache2, smurf, neptune, land, back, udpstorm, teardrop,
Probe	saint, mscan, portsweep, nmap, ipsweep, satan,
R2L	named, sendmail, httptunnel, snmpgue , ss, snmpgetattack, xsnoop, xlock, warezmaster, multi hop, guess_password, fip_write, phf, imap,
U2R	ps, xterm, sqlattack, perl, rootkit, buffer_overflow, loadmodule,

One of the advantages of the NSL-KDD over to its successor KDD’99 data set is that the amount of data contained in the train and test sets are reasonable so that every classifier can be evaluated on equal terms by using the same data sets. The number of data contained in the NSL-KDD train set is 125,973 and 22,544 for the test set. The Fig.2 shows the total number of data contained in NSL-KDD data set with the distribution to 5 classes. A sample record from the NSL-KDD data set is shown in Table 2.

B. Data Cleaning

After analyzing the training set, we encountered faulty data in the binary type attribute ‘su_attempted’ where 59 of the

samples are defined with the value of 2. We corrected the values of these samples as 0. Also, the attribute “num_outbound_cmds” has the same value across all the data set which is 0. Since the objective of the system is to find patterns in the data by examining the dissimilarities, we removed this feature from the data set. Thus, the total number of attributes present in the NSL-KDD data is reduced from 41 to 40.

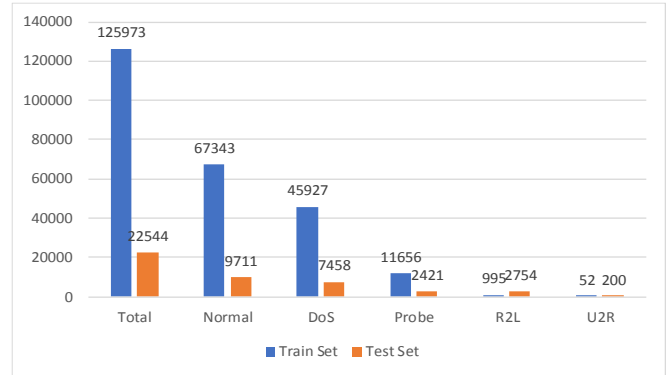


Fig.2. The number of data contained in the NSL-KDD data set

TABLE II
SAMPLE RECORD OF NSL-KDD DATA SET

0, udp, domain_u, SF, 79, 2766, 0, 0, 0, 3, 0, 1, 0, 0, 0, 4, 0, 0, 0, 0, 0, 4, 4, 0.00, 0.00, 0.00, 0.00, 1.00, 0.00, 0.00, 255, 255, 0.89,0.11,0.06,0.03, 0.00, 0.00, 0.00, 0.00, normal, 21
--

C. Data Preprocessing

In this stage, we described the methods that we used for converting the categorical data into numerical representations so that the neural network could have a better understanding of these data, thereby it can have a wider perspective on detecting the attacks. In this stage, the categorical data present in the NSL-KDD data set, which are known to be the three features in the nominal category as “protocol_type”, “service” and “flag”, converted into numerical representations using One-Hot Encoding method.

After transforming the categorical features, the number of attributes is increased from 40 to 121. To get into account this increased in the dimensions of the data set, a feature selection algorithm needs to be considered. This feature selection algorithm is detailed in the ongoing subsection.

D. Feature Selection

Even though there are 121 features representing the data in the data set, not all of them might necessarily assure the best performance for the IDS. Using unnecessary features means increased computational costs and error rates for the system. For this reason, considering the feature correlations, a set of selected features can have the same or rather better results compared to using all of them. Therefore, we used a feature selection algorithm called as Principal Component Analysis (PCA) which is a technique that has the aim of transforming several variables that are correlated with each other to several

variables that are uncorrelated and called as Principal Components (PCs).

The aim of the PCA is to reduce the dimensions of the initial variables in the data set while keeping the variance as same as the initial state [15]. To find the best column vector sample to work with, we experimented every possible vector sample using the original NSL-KDD train set with the DNN classifier and found the column vector samples with the highest accuracy score to be 37. Fig.3 shows the error rate results obtained from an interval sample of the experiments.

E. Data Normalization

Scaling or normalization is a concept of transferring the data into forms and ranges so that the modeling can be done more appropriately. By this definition, we used a normalization technique called Min-Max Normalization.

In this technique, the original data is transformed linearly by finding the maximum and minimum values in every feature set. In [16], it is emphasized that when scaling is done for the training data set; the same process is also should be done on the test set using the numerical characteristics of the train data. Also, it is shown that the scaling method is improved the accuracy of the experimented classifiers.

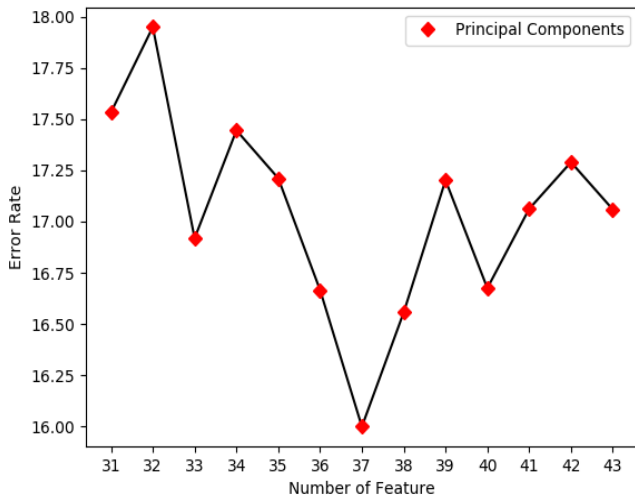


Fig.3. Best Correlated Features Chosen by the PCA

F. DNN Classifier

Using the TensorFlow framework, we built a GPU-accelerated DNN structure for classification of the data in NSL-KDD data set. The structure of the DNN classifier is composed of two hidden layers containing 64 neurons in the first layer and 32 neurons in the second layer. The parameters for the classifier’s initiation specified as 1,000 for the batch size and 500 for the number of epochs.

A pair of sigmoid functions are used as activation functions for both hidden layers. In Fig.4 [17], a general structure of the neural networks is shown.

G. K-Means Algorithm

The K-Means algorithm [18] which is one of the most commonly used clustering algorithms, finds the similarities

between the data points using a distance formula such as Euclidean, Manhattan or Minkowski distance [19].

In the proposed work, we used the Euclidean distance-based K-Means algorithm. The formula for the distance calculation is defined in the Equation (1) as:

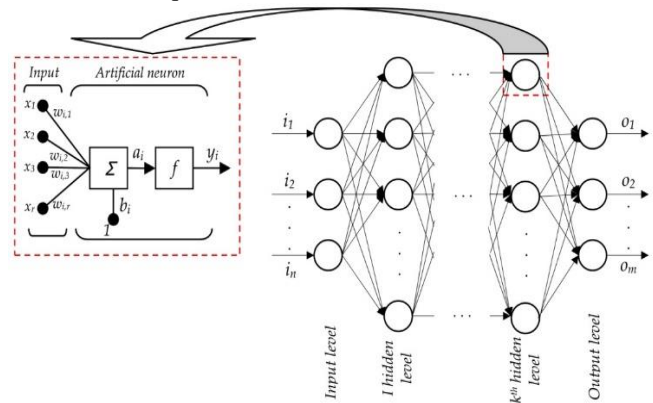


Fig.4. General Structure of Neural Networks

$$Dist_{xy} = \sqrt{\sum_{k=1}^m (X_{ik} - X_{jk})^2} \tag{1}$$

Using this formula, the steps for finding the clusters using the K-Means algorithm can be defined as:

- Let $X = \{x_1, x_2, x_3, \dots, x_n\}$ be the instances and $V = \{v_1, v_2, v_3, \dots, v_c\}$ be the set of centers.
- 1. Choose a cluster center ‘c’ randomly.
- 2. The distance between each instance is found, and the cluster center is recalculated.
- 3. Assign the data points to the cluster centers with minimum distances.
- 4. Cluster center is updated with using the Equation (2):

$$V_i = \left(\frac{1}{C_i}\right) \sum_1^{C_i} x_i \tag{2}$$

- where C_i represents the number of data in i^{th} the cluster.
- 5. Recalculate each instance distance to the center of the cluster.
- 6. Unless the state of the clusters does not change, repeat the steps 3-5.

H. Hybrid IDS Approach Using K-Means with DNN

Clustering is one of the well-known types of techniques for finding relations in the data and putting them into similar groups. The partitioning of the data into clusters happens such that the similarity of a cluster is greater than that among other clusters [20]. In this approach, a hybrid IDS structure has been built using the K-Means algorithm and a GPU-Accelerated DNN. The addition of the K-Means algorithm helps to cluster the NSL-KDD data set into groups of data based on similarity.

The implementation of the K-Means algorithm is based on the study made in [21]. In our approach, the training data set is partitioned into clusters using the K-Means algorithm. After the partitioning, the cluster centers of the training set are used for partitioning the test set into clusters. Using the cluster centers of the training set to partition the test set provides a good data distribution. This plays an important role in finding the best clusters containing pairs of the train and test sets with similar data. A visual representation of this process is shown in Fig.5.

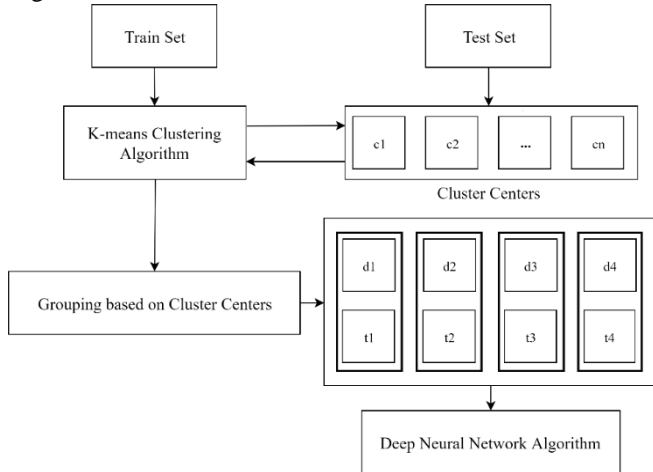


Fig.5. Visual Representation of the K-Means Algorithm Implementation

IV. EXPERIMENTAL RESULTS

The test environment contains a notebook with the features of Intel Core x64-based processor i5-4200H CPU @ 2.80 GHz, 12 GB Random Memory Access (RAM), 64-bit Windows 8.1 Pro Operating System. The NVIDIA GTX 1050Ti Graphics Processing Unit (GPU) is used for experimenting and building the GPU-Accelerated model. The specifications are shown in Table 3.

TABLE III
GPU SPECIFICATIONS

Property	Value
Processor	NVIDIA GeForce GTX 1050
NVIDIA Cuda Cores	768
Base Clock	1290 MHz
Boost Clock	1392MHz
Graphics Performance	high-6747
Memory Speed	7 Gbps
Memory Config	4GB GDDR5
Memory Interface	GDDR5
Memory Interface Width	128-bit
Memory Bandwidth	112(GB/sec)

The performance measures used for the evaluation of the neural network are the following:

- False positives (FP): defines the attacks that are detected incorrectly.
- False negatives (FN): instances of attacks detected as normal behavior.
- True positives (TP): correctly classified instances as normal behavior.
- True negatives (TN): correctly classified instances as attacks.
- The accuracy or True positive rate (TPR): the percentage of the ratio of correct predictions overall predictions:

$$\frac{TP + TN}{TP + TN + FP + FN} \times 100 \tag{3}$$

False alarm rate is defined as:

$$\frac{FP}{TN} \tag{4}$$

Detection rate is defined as:

$$\frac{TN - FN}{TN} \tag{5}$$

Precision (P) is defined as:

$$\frac{TP}{TP + FN} \tag{6}$$

Recall (R) is defined as the ratio of correctly predicted normal behaviors over all the correctly predicted instances:

$$\frac{TP}{TP + FP} \tag{7}$$

F-measure is defined as the consideration of both the precision and recall computing the score:

$$2 \times \frac{P \times R}{P + R} \tag{8}$$

According to these performance measures, Firstly, we tested our system without clustering approach. The prediction results obtained by the neural network trained on the original NSL-KDD data set is shown in Table 4. As can be seen form this table, system return acceptable results if there exist sufficient data as in Normal, DoS and Probe classification. However, in the other case the prediction rate is very small.

TABLE IV
PREDICTION RESULTS OF THE NEURAL NETWORK ON THE TEST DATA

Data Label	Test Data	Correct Predictions	Incorrect Predictions	Prediction Rate %
Normal	9711	9424	287	94.04
DoS	7458	7060	398	94.66
Probe	2421	2084	337	86.08
R2L	2754	815	1939	9.5
U2R	200	19	181	29.59

Addition to these prediction values, the confusion matrix values have great importance. Therefore, these values are listed in Table 5. Although, total accuracy rates are in acceptable level, in the R2L and U2R type packets, false values are too large.

TABLE V
NUMBER OF TP, TN, FP, AND FN FOR THE OPTIMAL NEURAL NETWORK RESULTS

True Positives	True Negatives	False Positives	False Negatives
9424	9978	287	285

If we get the sum of both these tables, the performance metrics can be summarized as in Table 6.

TABLE VI
PERFORMANCE MEASURES FOR THE OPTIMAL NEURAL NETWORK RESULTS

Accuracy	False Alarm	Detection Rate	Precision	Recall	F-measure
86.0628	0.02876	0.71387	0.76749	0.9705	0.85712

After analyzing the Table 4, the prediction rates of the attacks from the attack class of R2L and U2R took our attention as the neural network could not give good results predicting them. Finding the performance measure results showed that the performance of the neural network is indeed not as good as expected.

Therefore, a cluster-based solution is proposed in this paper as the flow is showed in Fig. 5 with data distribution of the 4 clusters. Additionally, the analysis of the NSL-KDD data set showed that the number of data contained in the train and test sets based on the attack classes R2L and U2R are not appropriate for the learning process. For this reason, we used a shuffled dataset in our proposed approach. According to the proposed system's approach, the results are shown in Table 7.

TABLE VII
PREDICTION RESULTS OF THE PROPOSED SYSTEM ON THE TEST DATA

Data Label	Test Data	Correct Predictions	Incorrect Predictions	Prediction Rate(%)
Normal	15,376	15,233	143	99.07
DoS	10,779	10,742	37	99.66
Probe	2,751	2,741	10	99.64
R2L	754	696	58	92.31
U2R	44	36	8	81.82

To understand the detailed performance of the proposed system, the confusion matrixes of clusters are listed in Table 8.

TABLE VIII
THE NUMBER OF TP, TN, FP, AND FN FOR CLUSTERS

Cluster #	True Positives	True Negatives	False Positives	False Negatives
1	29	7518	3	1
2	619	3395	8	5
3	10975	918	35	64
4	3610	2388	97	39

Table 7 shows the data distribution of the 4 clusters. Analyzing the clusters shows that the distributions are generally biased towards either more to the attack types or to the normal data. This proves that the K-Means algorithm does cluster the data that have similar identities. Another representation of the distributions of the clusters is shown in Fig.6.

As can be seen from this Figure, clustering based approach mainly collects the data group according to its similarities with the clusters and to make classifications. Each cluster does not correspond to a specific type of attack. Cluster1 and Cluster4 mainly stores lots of Normal type requests, other types distributed to the related clusters according to their similarities. According to these results, the performance measures are found as shown in Table 9.

After calculating the performance measures, an average of 99.15 % accuracy scores is found which is shown in Table 9. In the field of IDSs, the false alarm rate is an important factor to determine the effectiveness of the system. In our results, the average false alarm rate is found to be as 0.02. The average precision and recall rates obtained by our system are 0.97 and 0.99, respectively. From these results, the average F-measure is found as 0.98.

TABLE IX
PERFORMANCE MEASURES FOR THE PROPOSED SYSTEM'S RESULTS

Cluster #	Accuracy %	False Alarm	Precision	Recall	F-measure
1	99.947	0.0003	0.9063	0.967	0.9355
2	99.677	0.0023	0.9872	0.992	0.9896
3	99.175	0.0381	0.9968	0.994	0.9942
4	97.783	0.0406	0.9738	0.989	0.9815
Avg	99.145	0.0203	0.9660	0.986	0.9752

Finally, the performance of the proposed system is compared according to the CPU and GPU performances. The used IDE platform, Tensorflow, enables the use of GPU power. Therefore, the performance comparison of both CPU and GPU is depicted in the Table 10. The computing power and related parameters of the used GPU/CUDA platform is depicted in Table 3. Currently there are betted computing power GPUs, in the use of the it is expected to increase the efficiency of the system.



Fig.6. Data Distribution Across the 4 Clusters

TABLE X
TRAINING AND TESTING TIMES BASED ON CPU

Cluster #	GPU Time		CPU Time	
	Train	Test	Train	Test
1	31.659	0.0156	64.702	0.0158
2	21.016	0.0015	35.828	0.0015
3	59.438	0.0157	99.079	0.0159
4	37.879	0.0013	53.192	0.0014

According to these data, when comparing the GPU-Accelerated DNN times against CPU based DNN, an improvement can be seen. However, the performance improvement is also related with the design of the proposed Deep Learning approach. Therefore, if the level/complexity of the system increased, the performance improvement also enhanced.

V. CONCLUSION AND FUTURE WORK

In recent years, there is an increased number of computers and computing devices which are connected to the Internet and they are online with anytime and anywhere concept. As a result, there is a growing importance of the cybersecurity of not only single device but also networks. To prevent the anonymous attacks from both the outsiders and the insiders, firewalls and antiviruses cannot enable enough prevention. Therefore, some additional attack prevention and detection mechanisms are needed to be used.

In this paper, it is aimed to implement an Intrusion Detection System, which uses a deep neural network technology with Big Data for training the system. Although some previous works

prefer different static architecture such as rule based systems, a learning-based IDS system is aimed to develop. To diminish one of the important deficiencies of the use of the huge size of data, a parallel execution platform, as NVidia CUDA platform, is preferred. Additionally, to increase the accuracy rate of the system a clustered approach is preferred. The experimental results showed that the proposed system decreases the training time of the system with the use of CUDA platform and has a very good accuracy rate in the testing phase.

In the future work, it is aimed to use a modern updated dataset, with a bigger size of data. Additionally, there are new deep learning models that can be applied to Intrusion Detection Systems [22]. They can be also implemented for getting better accuracy rates.

REFERENCES

- [1] A. Borkar, A. Donode, and A. Kumari, "A survey on Intrusion Detection System (IDS) and Internal Intrusion Detection and protection system (IIDPS)," 2017 International Conference on Invenitive Computing and Informatics (ICICI), Coimbatore, 2017, pp. 949-953.
- [2] S. Han, M. Xie, H. Chen and Y. Ling, "Intrusion Detection in Cyber-Physical Systems: Techniques and Challenges," in IEEE Systems Journal, vol. 8, no. 4, pp. 1052-1062, Dec. 2014. doi: 10.1109/JSYST.2013.2257594
- [3] L. Haripriya and M. A. Jabbar, "Role of Machine Learning in Intrusion Detection System: Review," 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, 2018, pp. 925-929.
- [4] E. Aydemir, "Weka ile Yapay Zekâ", Seekin, p:216, July 2018
- [5] P. I. Radoglou-Grammatikis and P. G. Sarigiannidis, "Securing the Smart Grid: A Comprehensive Compilation of Intrusion Detection and Prevention Systems," in IEEE Access, vol. 7, pp. 46595-46620, 2019. doi: 10.1109/ACCESS.2019.2909807
- [6] F. Farahnakian and J. Heikkonen, "A deep auto-encoder based approach for intrusion detection system," in Advanced Communication

- Technology (ICACT), 2018 20th International Conference on. IEEE, 2018, pp. 178–183.
- [7] E.H. Spafford, D. Zamboni, "Intrusion Detection Using Autonomous Agents. Computer Networks", The International Journal of Computer and Telecommunications Networking 34 (4), 547–570 (2000)
- [8] H. Sagha, S. B. Shouraki, H. Khasteh and M. Dehghani, "Real-Time IDS Using Reinforcement Learning," 2008 Second International Symposium on Intelligent Information Technology Application, Shanghai, 2008, pp. 593-597. doi: 10.1109/IITA.2008.512
- [9] M. A. Jabbar and S. Samreen, "Intelligent network intrusion detection using alternating decision trees," 2016 International Conference on Circuits, Controls, Communications and Computing (I4C), Bangalore, 2016, pp. 1-6. doi: 10.1109/CIMCA.2016.8053265
- [10] A. S. Desai and D. P. Gaikwad, "Real time hybrid intrusion detection system using signature matching algorithm and fuzzy-GA," 2016 IEEE International Conference on Advances in Electronics, Communication and Computer Technology (ICAECCT), Pune, 2016, pp. 291-294.
- [11] O. Can and O. K. Sahingoz, "An intrusion detection system based on neural network," 2015 23rd Signal Processing and Communications Applications Conference (SIU), Malatya, 2015, pp. 2302-2305. doi: 10.1109/SIU.2015.7130338
- [12] G. Karatas and O. K. Sahingoz, "Neural network based intrusion detection systems with different training functions," 2018 6th International Symposium on Digital Forensic and Security (ISDFS), Antalya, 2018, pp. 1-6. doi: 10.1109/ISDFS.2018.8355327
- [13] C.F. Tsai, Y.F. Hsu, C.-Y. Lin, W.-Y. Lin, "Intrusion detection by machine learning: A review", Expert Systems with Applications, 2009, vol. 36, no. 10, pp. 11994-120000.
- [14] W.C. Lin, S.W. Ke, C.F. Tsai, "CANN: An Intrusion Detection System Based on Combining Cluster Centers and Nearest Neighbors", Knowledge-Based Systems, 2015, vol. 78, pp. 13-21.
- [15] Z. Elkhadir, K. Chougali and M. Benattou, "Intrusion Detection System Using PCA and Kernel PCA Methods", In: El Oualkadi A., Choubani F., El Moussati A. (eds) Proceedings of the Mediterranean Conference on Information & Communication Technologies 2015. Lecture Notes in Electrical Engineering, vol. 381, Springer.
- [16] N. Paulauskas and J. Auskalis, "Analysis of Data Pre-Processing Influence on Intrusion Detection Using NSL-KDD Dataset", 2017 Open Conference of Electrical, Electronic and Information Sciences (eStream), 2017, Vilnius, pp. 1-5.
- [17] D. Tanikić and V. Despotovic, "Artificial Intelligence Techniques for Modelling of Temperature in the Metal Cutting Process", 2012, Metallurgy, Yogiraj Pardhi, IntechOpen, DOI 10.5772/47850
- [18] T. Velmurugan and T. Santhanam, "Performance Evaluation of K-Means and Fuzzy C-Means Clustering Algorithms for Statistical Distributions of Input Data Points" European Journal of Scientific Research, vol. 46, no. 3, pp. 320-330, 2010.
- [19] A. Singh, A. Yadav & A. Rana, "K-means with Three Different Distance Metrics", International Journal of Computer Applications, 2013, vol. 67, no. 10, pp. 13-17.
- [20] A.K. Jain, "Data clustering: 50 years beyond K-means", Pattern Recognition Letters, 2010, vol. 31, no. 8, pp. 651-666.
- [21] D. Y. Mahmood, M. A. Hussein, "Feature Based Unsupervised Intrusion Detection", International Journal of Computer and Information Engineering, 2014, vol. 8, no. 9, pp. 1665-1669.
- [22] G. Karatas, O. Demir and O. K. Sahingoz, "Deep Learning in Intrusion Detection Systems," 2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT), Ankara, Turkey, 2018, pp. 113-116. doi: 10.1109/IBIGDELFT.2018.8625278

BIOGRAPHIES



BUMINHAN REIS received the B.S. degree in Computer Engineering from Istanbul Kultur University in 2018. He has been studying foreign languages (Chinese) from that time. His research interests are machine learning, computer security and software engineering.



SAMI BERK KAYA received the B.S. degree in Computer Engineering from Istanbul Kultur University. He worked as software developer and analyst and currently he has been working in BEEBOT software company since 2018. His research interests are big data processing, machine learning, computer security and software engineering.



OZGUR KORAY SAHINGOZ received the B.S. degree from the Computer Engineering Department of Bogaziçi University in 1993 and M.S. and PhD degrees in Computer Engineering Department of Istanbul Technical University in 1998 and 2006 respectively. He is currently working as an Associate Professor in the Computer Engineering Department of Istanbul Kultur University. He is the author of more than 95 papers and he is still working in two research projects. His research interests include artificial intelligence, machine/deep learning, data science, software engineering and UAV Networking.