NON-NEGATIVE MATRIX FACTORIZATION BASED INTRUSION DETECTION SYSTEM FOR IOT TRAFFIC

Abderezak Touzene, Ahmed Al Farsi and Nasser Al Zeidi

Department of Computer Science, College of Science, Sultan Qaboos University, Oman

ABSTRACT

With the emergence of smart devices and the Internet of Things (IoT), millions of users connected to the network produce massive network traffic datasets. These vast datasets of network traffic (Big Data) are challenging to store, deal with and analyse to detect normal or cyber-attack traffic. In this paper we developed an Intrusion Detection System (NMF-IDS) based on Non-Negative Matrix Factorization dimension reduction technique to handle the large traffic datasets and efficiently analyses them in order to detect with a good precision the normal and attack traffic. The experiments we conducted on the proposed IDS-NMF give better results than the traditional ML-based intrusion detection systems, we have got an excellent detection accuracy of 98%.

KEYWORDS

Intrusion Detection Systems, Machine Learning, Dimensionality Reduction, IoT traffic.

1. INTRODUCTION

Based on reports published by cybersecurity institutions in several countries worldwide, network cyber-attacks have increased exponentially in recent decades. These days, as we witness the era of the Fourth Industrial Revolution, 4IR and its emerging technologies like the Internet of Things, Quantum Computing, and Artificial Intelligence. Millions of users have become connected to the Internet, and hundreds of millions of devices connected to the network produce millions of large network traffic records datasets. Those massive datasets from network traffic contain millions of cyber-attacks, so it is crucial to analyse this data quickly in real-time to detect attacks.

This paper focus on Nonnegative Matrix Factorization as a dimension reduced technique to efficiently analyses large IoT network traffic. Nonnegative Matrix Factorization (NMF) is an approximation numerical method aiming at decomposing data matrix A into its simpler factors lower-rank matrices H and W. NMF is an unsupervised Machine Learning technique widely used in data mining, dimension reduction, clustering, factor analysis, text mining, computer vision, and bioinformatics, image recognition and recommendation systems to name a few. In contrast to Singular Value Decomposition (SVD) and Principal Component Analysis (PCA), Nonnegative Matrix Factorization (NMF) requires that A, W, and H be nonnegative. For many real-world data, non-negativity is inherent, and the interpretation of factors has a natural interpretation which could be one of the advantages of NMF compared to PCA and SVD.

Formally, Nonnegative Matrix Factorization problem is to find two low-rank factors matrix $W_{m \times k}$ and $H_{k \times n}$ for a given nonnegative matrix $A_{m \times n}$, such that A ≈ WH. Most of the available optimization techniques include Hierarchical Alternative Least squares HALS, Multiplicative Updates (MU), Stochastic Gradient Descent, and Block Principal Pivoting (ALNS-BPP), which are based on alternating optimizing W and H while keeping one of them fixed.

1.1. Intrusion Detection System Background

This section discusses the background of Intrusion detection systems, including their definition and diverse types. Moreover, it discusses several papers on machine learning IDS algorithms.

1.1.1. Intrusion Detection System (IDS)

An intrusion Detection System is defined as a hardware device or software that observes systems for malicious network traffic or policy violations. The purpose of IDS is to detect various types of malicious network traffic or malicious computer use that a firewall cannot recognize. This is critical to achieving high protection against actions threatening computer systems' availability, integrity, or confidentiality [1].

1.1.2. Types of Introduction Detection Systems (IDS)

There are many classifications for intrusion detection systems. However, this classification has been used extensively in previous studies based on the data collection method:

1. Network intrusion detection system, which observes and analyses data traffic to detect if there is an attack or malicious behaviour (NIDS).
2. The Host-based intrusion detection system monitors and analyses data from log files (HIDS), and based on the detection technique, it can be categorized into three main categories: Specification-based IDS, Anomaly-based IDS, and signature-based IDS.

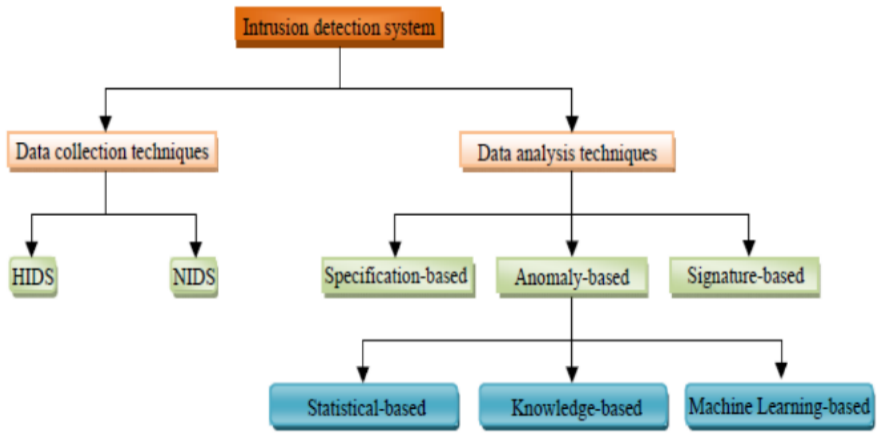

Figure 1: IDS Classification

- **Signature-based (Misuse) Intrusion Detection Systems:** Signature-based intrusion detection analyzes network traffic, searching for occasions or combinations similar to a predefined event pattern that describes a known attack.
  Advantages: Signature-based Intrusion Detection Systems effectively detect intrusions without too many false alarms.

Disadvantages: Signature-based-IDS can only identify attacks it knows in advance, requiring constant updating signatures. Signature-based-IDS detectors sometimes are trained very well in detecting specific types of attacks that prevent them from seeing some new kinds of attacks.

- **Anomaly-based Introduction Detection Systems:** Anomaly-based intrusion detection systems identify abnormal behavior on a network. Commonly attacks differ from regular legitimate network traffic; the detector can detect them by detecting these changes and differences. Anomaly detectors are trained well on normal network traffic from historical data collected. So, they can see abnormal behaviors easily.

  Advantages: Anomaly-based Intrusion Detection systems can detect abnormal behavior, so they can detect an attack without any knowledge about it

  Disadvantages: Because of the variations in users and network behaviors, Anomaly-based IDS may fire many false alarms. Anomaly detection approaches must be trained in huge datasets of normal behavior activities.

1.2. Motivation

In this paper we aim at overcoming some of the limitations existing in traditional machine learning-based Intrusion Detection Systems (IDS), such as a considerable amount of training and testing on Big data datasets and to detect type of attack in real-time.

In this study we analyse the performance of ML-based IDS using KDD and CIC datasets by applying NMF, which will help reduce the dataset's dimensions into lower-rank matrices that can be used for analysing and testing any new network traffic in real time.

The rest of the paper is structured of the remaining as follows: In section 2, will start with a brief background on Machine Learning based IDS and NMF and introduce the related work on which the proposed solution will be built. In section 3 will discuss the design of the proposed NMF based IDS including the, and detection phase. Section 4 is dedicated to the experimental work, it describes the implementation environment and the datasets used, and the performance evaluation of our IDS. Finally, section 5 will summarizes the paper and highlights some limitations along with the future works.

2. BACKGROUND AND RELATED WORK

This section will discuss the background of Machine Learning IDS and background of NMF.

2.1. Machine learning-based IDS

Many recent Anomaly Intrusion Detection Systems (AIDS) is based on Machine Learning methods. There are a lot of ML algorithms and methods used for ML-based IDS, such as neural networks, nearest neighbour, decision trees, and clustering methods, applied to discover the meaningful features from IDS datasets [1] [2].

2.1.1. Supervised learning in Intrusion Detection System

Supervised ML-based IDS techniques that can identify attacks based on labeled training datasets. A supervised ML technique can be divided into training and testing. Training phase, important features are specified and processed in datasets, then we train the model from these datasets. There are many applications of supervised machine learning-based IDS. Li et al. used an SVM classifier with an RBF kernel to classify the KDD 1999 dataset into predefined classes. From a total of 41 attributes, a subset of features was carefully chosen by using the feature selection approach [3]. K-Nearest Neighbours (KNN) classifier: The k-Nearest Neighbour (k-NN) method is a typical non-parametric classifier used in machine learning [4]. These methods aim to name an unlabelled data sample to the class of its k nearest neighbours. Point X denotes a sample of unlabelled data that needs to be classified.

2.1.2. Unsupervised ML-based Intrusion Detection System

Unsupervised ML can be defined as an ML technique that obtains information of interest from input data sets without class labels. The input data points are usually treated as a set of random variables. A standard density model is then generated for the data set. In supervised learning, output labels are presented and used to train the machine to obtain the desired results for an unseen data point. By contrast, in unsupervised learning, no label is provided. Instead, the data is automatically grouped into different categories through the learning process [5].

2.2. Nonnegative Matrix Factorization (NMF)

Non-negative matrix factorization is an algorithm that takes a nonnegative input matrix $A \in \mathbb{R}^{m\times n}$ and decomposes it into lower rank matrices W and H based in low rank parameter K. NMF is an unsupervised Machine Learning technique commonly used in clustering, dimensionality reduction, factor analysis, data/text mining, computer vision, bioinformatics, image recognition and recommendation systems to name a few. In contrast to Principal Component Analysis (PCA) and Singular Value Decomposition (SVD), NMF requires that A, W, and H be nonnegative. For many real-world data, non-negativity is inherent, and the interpretation of factors has a natural interpretation which could be one of the advantages of NMF compared to PCA and SVD [10] [11].

2.2.1. Foundations of Nonnegative Matrix Factorization

NMF takes a nonnegative input matrix $A \in \mathbb{R}^{m\times n}$ m is number of rows which represent number of features and n is number of column which represent number of samples, and low rank parameter K which is positive integer < $\{\mathrm{m}, \mathrm{n}\}$, NMF algorithms aims to find two low rank matrices $W \in \mathbb{R}^{m\times k}$ and $H \in \mathbb{R}^{k\times n}$ such that $A \approx WH$.

NMF aims to minimize the following cost function:

$$\min_{W,H} \|A - WH\|_F^2 \; \overset{such\ that}{W \geq 0 \; and \; H \geq 0}. \tag{1}$$

$$W \leftarrow \|A - \bar{W}H\|_F^2 \tag{2}$$

$$H \leftarrow \|A - W\bar{H}\|_F^2 \tag{3}$$

Most of the available optimization techniques include Hierarchical Alternative Least squares HALS, Multiplicative Updates (MU), Stochastic Gradient Descent, and Block Principal Pivoting (ALNS-BPP), which are based on alternating optimizing W and H while keeping one of them fixed.

2.3. Related Work

X. Guan in [6] presented an efficient and fast anomalous intrusion detection model that includes many data from different sources. A new method based on non-negative matrix factorization (NMF) is discussed to characterize program and user behaviors in a computer system. A large amount of high-dimensional data was collected in their experiments. NMF was used and reduced the vectors to a smaller vector length after that, any simple classifier can be implemented in low-dimension data instead of the entire dataset. After getting low dimension features the model can differentiate between normal traffic and abnormal traffic easily by using a threshold, so any user behavior on that threshold will be considered an attack.

Limitations: Although the implemented NMF-based IDS gives good accuracy, the datasets were nonstandard. Moreover, the threshold technique used in the testing phase could not be applied to multi-class network attacks.

In this work we implement our NMF-IDS and test it using recent standard datasets specialized in IoT traffic such as **KDD99** and **CIC-IDS2017.** We also developed a new technique to detect multi-class attacks.

3. NMF BASED INTRUSION DETECTION SYSTEM

NMF-IDS system consist of three major phases. In phase 1 the network dataset file is converted into a two dimensional matrix $A^{m\times n}$. In phase 2, the matrix $A^{m\times n}$ will be factorized into two low-rank matrices $W^{m\times k}$ and $H^{k\times n}$. Phase 3 consists of the detection phase.

3.1. NMF Factorization Phase

Lee and swing [7] proposed a multiplicative updates algorithm (MU) to solve the NMF factorization problem as follows:

The matrices W and H are updated using the following formulas:

$$w_{ij} \leftarrow w_{ij} \frac{(AH^T)_{ij}}{(WHH^T)_{ij}} \tag{4}$$

$$h_{ij} \leftarrow h_{ij} \frac{(W^T A)_{ij}}{(W^T WH)_{ij}} \tag{5}$$

H^T means the matrix transpose of the matrix H. MU algorithm can be divided into individual smaller sub problems of matrix dot product. In step 1 we update W based on AH^T, HH^T and WHH^T, then in step 2 we update H based on W^TA, W^TW and W^TWH. See algorithm 1 below.

Algorithm (1)

```
• $[W, H] = NMF(A, k)$
• $A^{m\times n}$ is the input matrix,
  $k$ is rank of approximation
• (1): initialize random matrix $H$
• (2): **while** stopping criteria are not satisfied **do**
    ○ /*compute W given H*/
    ○ (3): $computes\ W \leftarrow W_i \frac{A\ H^T}{WHH^T}$
    ○ /*compute H given W*/
    ○ (4): $computes\ H \leftarrow H^i \frac{W^T A}{W^T WH}$
    ○ (5): **end while**
```

The while loop at algorithm 1 will stop if the stopping criteria are satisfied. Either it reaches the maximum number of iterations specified by the user, or it reaches convergence based on the Frobenius norm function $\min_{W,H} ||A - WH||_F^2 \text{ such that } W \geq 0 \text{ and } H \geq 0$.

3.2. NMF Detection Phase

After MU algorithm reach to convergence or it reach the maximum number of iterations specified by the user, we obtain the factor matrices *W* and *H*, that can be used to represent every sample from *A* as weighted linear combination of columns of W, every column of w called bases where the corresponding h_{ij} called the weights or encoding.

$$\begin{bmatrix} a_1 \\ . \\ . \\ . \\ . \end{bmatrix} = \begin{bmatrix} w_1 & w_2 & . & w_k \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \end{bmatrix} * \begin{bmatrix} h_{11} \\ h_{12} \\ . \\ . \\ h_{1k} \end{bmatrix} \tag{6}$$

$$a_1 = W * h_1$$

$$\begin{bmatrix} a_1 \\ . \\ . \\ . \\ . \end{bmatrix} = \begin{bmatrix} w_1 \\ . \\ . \\ . \\ . \end{bmatrix} h_{11} \begin{bmatrix} w_2 \\ . \\ . \\ . \\ . \end{bmatrix} h_{12} + \ldots + \begin{bmatrix} w_k \\ . \\ . \\ . \\ . \end{bmatrix} h_{1k}$$

Now if we have a new sample let's call it i and we want to check if it matches one or more of the samples from the training set, we compute the encoding of it using W:

$$i = W * h' \tag{7}$$

Based on the upper formula we can find h' by:

$$h' = (W^T W)^{-1} W^T \times i \tag{8}$$

Now we check the similarity of this encoding with every encoding that existed in H. The closest match (sample class) is that sample whose encoding is the closest to the new sample (multi-class detection).

We can determine the matching score between the encodings using the following formula**:**

$$s_1 = \frac{h'.\, h_1}{|h'|\,|h_1|} \quad s_2 = \frac{h'.\, h_2}{|h'|\,|h_2|} \quad . \; . \; . \quad s_n = \frac{h'.\, h_n}{|h'|\,|h_n|}. \tag{9}$$

Where s_i with the maximum value is the closest encoding to the new input sample i.

4. Implementation of NMF-IDS and Experimental Results

4.1. NMF-IDS Methodology

In this study we will test our NMF-IDS on two known datasets (KDD, and CIC) after a pre-processing phase, training phase using NMF factorization eq. 4, 5, and detection testing eq. 7, 8, 9.

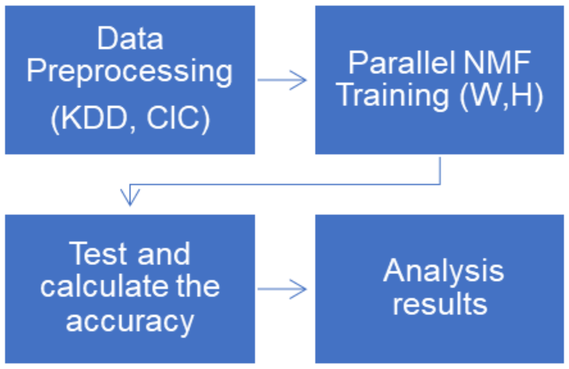
Data Preprocessing (KDD, CIC)
Parallel NMF Training (W,H)
Test and calculate the accuracy
Analysis results

Figure 2: NMF-IDS Methodology

4.2. Datasets

This section will discuss the datasets used in this study in detail and how we pre-process them. To study the performance of the proposed solution in this study, we applied it and analysed its efficiency and accuracy on two different datasets, namely KDD and CIC.

KDD99: KDD dataset is a dataset used in an international competition held at the University of California [8], where the goal of that competition was to build an intrusion detection system that can differentiate between a normal good connection, or a bad connection called an intrusion or attack. KDD dataset is about 5 million connection records that was generated from 7 days' network traffic. It contains 41 features, and it is labelled by either Normal or specific type of attack. KDD contains attacks that can be categorized to Denial of service (DoS), Remote to local (R2L), and User to root (U2R).

CIC-IDS2017: The second dataset used in this study is CICIDS2017, created by I.Sharafaldin [9] from the Canadian Institute for Cybersecurity. It's a benchmark dataset consisting of **2830743 samples** and **78 features**. CIC dataset contains more recent attacks. For example, Brute Force FTP, DoS, infiltration, DDoS.

4.3. Datasets Pre-processing

This section explains the pre-processing methods for KDD and CIC datasets before applying NMF to them to get the best results.

Label Encoding: To apply NMF to any dataset, we must ensure that all elements within the dataset are non-negative numbers. KDD dataset contains some features with text values namely, service, Protocol_type, and flag, so they need to be converted to numeric values using label_encoder from sklearn library of Python.

Normalization: Some features from the datasets contain large numbers. For example, src_bytes and dst_bytes from KDD have large values that can reach thousands. Also, in CIC dataset Flow_Duration contains values reach more than 1 million and Destination_Port can reach thousands, those great values may affect the model's performance as it will be biased to those great values. Therefore, normalization is applied to ensure that all the dataset's values are in the same range. In this study, we apply min-max normalization to make sure that all the values are ranged between 0 and 1 only.

Train/Test Split: We divided KDD and CIC datasets into several training data sizes to apply the proposed parallel NMF on it.

- Training datasets sizes (30K)
- Testing dataset size 3K

The original shape of the datasets was $samples \times features$ to reduce the number of features and to get correct results out from NMF we will transpose the input matrix so it will be on the following shape $features \times samples$.

4.4. Experiments and Results

To conduct our experiments, the computer node is Lenovo Think-System SR630 which is based on CentOS 7 Linux operating system, Dual 14-cores CPUs, 197 GB RAM, and 480TB + 12GB (SSD) storage.

4.4.1. Experiment (1) Find Best Rank *K* for KDD Dataset

To make the most of NMF Algorithm, we must choose the rank hyper-parameter K carefully to ensure that it gives us good results and simultaneously reduces the dimensions of the dataset. Although we keep the most amount of the features whenever we raise the value of K, but in terms of detection accuracy and performance, it may give worse results and slower training.

Different runs of NMF with different values of *K*, as shown in Figure 3. After 1000 iterations of NMF updates, it gave varied results. By analyzing the results of the experiments, $K = 10$ was selected as it gives an accuracy rate reaching up to 98% in 1000 iterations.

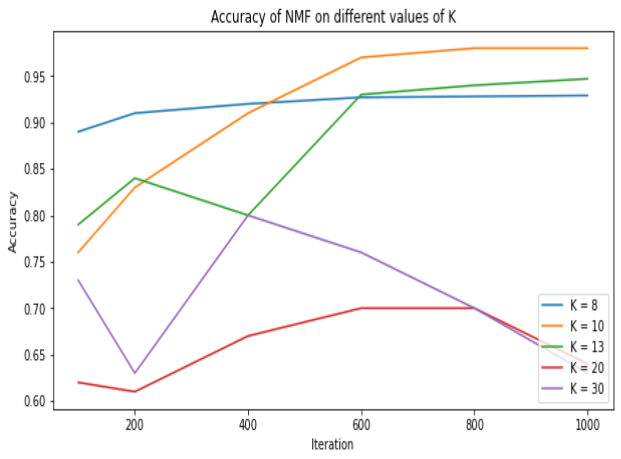

Figure 3: NMF-IDS Best Rank selection for KDD

4.3.2. Experiment (2) Training & Testing on 30K samples KDD

Using $K = 10$ we implemented NMF on 30000 samples of 42 features from KDD dataset. The results are shown in table (1)

Table 1. 30K samples KDD dataset Results

Iterations	**Training Time(s)**	**Accuracy (%)**
100	3.9	76
200	7.7	83
400	15.0	91
600	23	97
800	31	98

As shown in the Table 1, increasing the iterations gives better results the accuracy rate is increasing. But we get this at an additional cost of training time (factorization). As it is clear, NMF finished 100 iterations in approximately 4 seconds, with a detection accuracy of 76%, compared to 800 iterations in approximately 31 seconds, but with a detection accuracy of 98%.

4.3.3. Experiment (3) Find Best Rank *K* for CIC Dataset

In this experiment we run Different runs of NMF with different values of K for 1000 iteration, as shown in the Figure 4. After 1000 iterations of NMF updates, it gave varied results. By analyzing the results of the experiments, $K = 13$ was selected as it gives an accuracy rate reaching up to 90% in 1000 iterations.

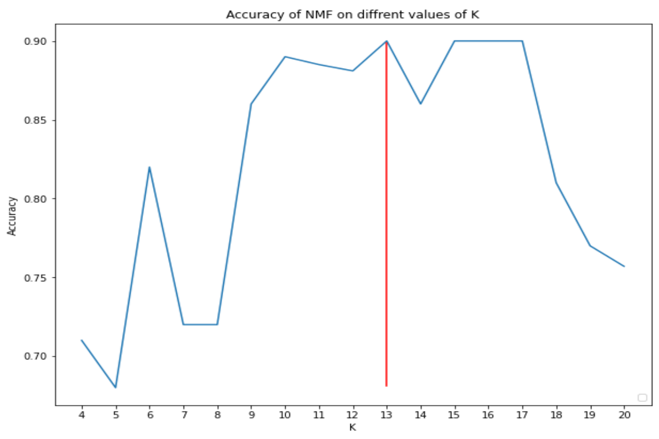

Figure 4: NMF-IDS Best Rank selection for CIC

4.3.4. Experiment (4) Training on 30K samples CIC

Using $K = 13$ based in experiment 4, we implemented parallel NMF on 30000 samples of CIC. Using different sizes of processors, we got the following results shown in Table 2.

Table 2. 30K samples CIC Dataset Results

Iterations	**Training Time(s)**	**Accuracy (%)**
100	20.5	66
200	41.0	74
400	82.1	84
600	124.0	87
800	164.0	90

As shown in the Table 2, increasing the iterations gives better results the accuracy rate is increasing but with an additional cost for training. NMF-IDS reaches a detection accuracy of 90% in 164 seconds.

5. CONCLUSION

Millions of users and hundreds of millions of devices connected to the network produce millions of network traffic records. In this paper we proposed an Intrusion detection system based on dimensionality reduction using Non-Negative Matrix Factorization (NMF) to be able to analyze efficiently large IoT traffic datasets. The algorithm used to compute NMF factors (W and H) is based Alternating Update algorithm using Multiplicative Update (MU). Unlike the previous work, our implementation can detect multi-class of network attacks. Experiential results show a detection precision of 98% for KDD datasets and 90% precision for CIC dataset. As a limitation to this work, we noticed that for large datasets the computational cost of the NMF factorization is large using a single computer. Another limitation is the slow convergence rate for MU method in term of number of iterations. In our future work we will extent NMF-IDS using High Performance Computers (HPC) to handle in real-time larger datasets. We will also consider using faster convergence alternative update Hierarchical Alternating Least Squares (HALS).

REFERENCES

[1] L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "IoT Security Techniques Based on Machine Learning: How Do IoT Devices Use AI to Enhance Security", IEEE Signal Process. Mag., vol. 35, no. 5, pp. 41–49, 2018.

[2] N. Kshetri and J. Voas, "Hacking Power," no. December, pp. 91–95, 2017.

[3] Y. Li, J. Xia, S. Zhang, J. Yan, X. Ai, and K. Dai, "An efficient intrusion detection system based on support vector machines and gradually feature removal method," Expert Syst. Appl., vol. 39, no. 1, pp. 424–430, 2012.

[4] W. C. Lin, S. W. Ke, and C. F. Tsai, "CANN: An intrusion detection system based on combining cluster centers and nearest neighbors," Knowledge-Based Syst., vol. 78, no. 1, pp. 13–21, 2015.

[5] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," Cybersecurity, vol. 2, no. 1, 2019.

[6] X. Guan, W. Wang, and X. Zhang, "Fast intrusion detection based on a non-negative matrix factorization model," J. Netw. Comput. Appl., vol. 32, no. 1, pp. 31–44, 2009.

[7] D. D. Lee and H. S. Seung, "Algorithms for Non-Negative Matrix Factorization," in Advances in Neural Information Processing Systems, no. 1, 2000, pp. 556–562.

[8] S. Stolfo, "KDD-99 Dataset," online, 1999. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (accessed Dec. 24, 2022).

[9] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," ICISSP 2018 - Proc. 4th Int. Conf. Inf. Syst. Secur. Priv., vol. 2018-Janua, no. Cic, pp. 108–116, 2018.

[10] R. Hedjam, A. Abdesselam, and F. Melgani, "NMF with feature relationship preservation penalty term for clustering problems," Pattern Recognit., vol. 112, 2021.

[11] T. Masuda, T. Migita and N. Takahashi, "An Algorithm for Randomized Nonnegative Matrix Factorization and Its Global Convergence," 2021 IEEE Symposium Series on Computational Intelligence (SSCI), Orlando, FL, USA, 2021, pp. 1-7.