**FMDB** *Transactions on Sustainable Intelligent Networks*

# Using Autoencoder-Driven Machine Learning for Advance Cybersecurity Malware Detection

**R. Shyam[1,*], G. Chiranjeevi[2], R. Abhishek Reddy[3], Ali A. Khala[4], Swati Sah[5], Sandeep Kautish[6], Rejwan Bin Sulaiman[7]**

[1]Department of Computer Science/Information Technology, Presidency College, Kempapura, Bengaluru, Karnataka, India.
[2,3]Department of Computer Science/Information Technology, Jain (Deemed-to-be University), Bengaluru, Karnataka, India.
[4]Department of Computer, University of Baghdad, Baghdad, Iraq.
[5]School of Engineering and Technology, Sharda University, Greater Noida, Uttar Pradesh, India.
[6]Department of Computer Science specializing in Intelligent Systems, Chandigarh University, Punjab, India.
[7]Department of Computer Science and Technology, Northumbria University, London, United Kingdom.
shyam7368@gmail.com[1], chiranjeevi.naidu1312@gmail.com[2], akabhiarvd2003@gmail.com[3], alrahmanali@sc.uobaghdad.edu.iq[4], iswatisah19@gmail.com[5], prof.sandeepkautish@gmail.com[6], rejwan.sulaiman@northumbria.ac.uk[7]

**Abstract:** In the present study, we tackle the problem of improving cybersecurity by creating a sophisticated artificial intelligence model that can precisely detect and categorize malware. In this work, we apply an autoencoder algorithm specifically designed to handle the high-dimensional and intricate data the Ember dataset contains. The objective is to develop an artificial intelligence system that can accurately identify benign and malicious executables and advance our knowledge of malware traits. Our goal is to capture the complex representations of the data by using an autoencoder. This will enable a more robust feature learning process, which is necessary to identify advanced cyber threats precisely. Although a thorough explanation of the model's statistical metrics is saved for the paper's main body, the abstract refers to encouraging findings that point to the autoencoder's ability to produce a highly accurate AI for malware classification. This lays the groundwork for a safer online environment by utilizing machine learning to combat new and emerging cyber threats.

**Keywords:** Cyberattack Detection; Malware Detection; Autoencoders and Anomaly Detection; Endgame Malware Benchmark for Research (EMBER); Reinforcement Learning (RL); Recurrent Neural Networks (RNNs); Internet of Things (IoT).

**Received on:** 02/07/2024, **Revised on:** 07/09/2024, **Accepted on:** 30/10/2024, **Published on:** 14/12/2024

**Journal Homepage:** https://www.fmdbpub.com/user/journals/details/FTSIN

**DOI:** https://doi.org/10.69888/FTSIN.2024.000292

**Cite as:** R. Shyam, G. Chiranjeevi, R. A. Reddy, A. A. Khala, S. Sah, S. Kautish, and R. B. Sulaiman, "Using Autoencoder-Driven Machine Learning for Advance Cybersecurity Malware Detection," *FMDB Transactions on Sustainable Intelligent Networks*., vol.1, no.4, pp. 252–264, 2024.



## 1. Introduction

[*]Corresponding author.

Threat actors and defenders are engaged in an arms race in the rapidly changing field of cybersecurity, with the former continuously creating new malware variants to evade detection. This has sparked the need for more advanced detection techniques that can adjust to the unpredictable nature of malware evolution. Anomaly detection using autoencoders, a class of artificial intelligence algorithms, is one approach at the forefront of this adaptive defence mechanism. Neural networks called autoencoders are used to learn codings from unlabeled data efficiently. They work based on unsupervised learning, whereby they learn to reconstruct the input from a compressed representation to extract the most important features from the data. An autoencoder's architecture is made to encode input data into a space with fewer dimensions, which is subsequently decoded to produce an output that is identical to the original input. To help the autoencoder find and use patterns in the data, the training process involves minimizing the difference between the input and the output.

The ability of autoencoders to acquire a representation of 'normal' data makes them important for anomaly detection. When trained on a dataset devoid of anomalies, an autoencoder gains proficiency in reconstructing inputs following the learned pattern. It will, however, have difficulty reconstructing inputs that substantially depart from this pattern—inputs that might be anomalies, like malware. As mentioned in papers [6]; [7]; and [8]. In this context, the reconstruction error can be a potent signal of anomalous behaviour, highlighting a departure from the norm that calls for additional research. This technique can train an autoencoder on a dataset of safe software executables to detect malware. After the model becomes proficient in reconstructing such data, it can be utilized to examine novel executables. A high reconstruction error could mean that the executable in question has features that are not typical of safe software, meaning malware is present. This method is especially advantageous because it doesn't rely on predefined or static malware signatures—which are frequently vulnerable to obfuscation and evasion techniques employed by attackers.

In this paper, we explore using the extensive Ember dataset to implement an autoencoder for malware identification. We clarify how the autoencoder is trained, how its architecture is adjusted, and how the threshold for anomaly detection is set. Our method shows how this approach can detect unknown and novel malware, adding a vital tool to the cybersecurity toolbox that can change with the times to meet the ever-changing threats posed by cybercriminals.

## 2. Literature Survey

Shaukat et al. [1] analyze the growing cyberspace vulnerability linked to the widespread increase in internet and mobile application usage. It highlights a major cybersecurity challenge: traditional security systems cannot successfully thwart complex cyberattacks involving novel and polymorphic threats. Research on advanced machine learning (ML) techniques for malware detection is especially pertinent to this aspect of cybersecurity. The abstract highlights that although machine learning (ML) has significantly improved cybersecurity measures—particularly in domains like malware, spam, and intrusion detection—it also faces significant obstacles. Among these difficulties is keeping machine learning (ML) systems reliable despite motivated adversaries skilled at exploiting holes in these systems. The paper aims to present a thorough overview of machine learning techniques used in cybersecurity, particularly in the last ten years. This includes thoroughly explaining numerous machine learning techniques, using security datasets, necessary ML tools, and the evaluation metrics needed to rate classification models. This research's focus on machine learning in cybersecurity—especially its application to malware detection—makes it pertinent to your paper. The conversation about machine learning tools, datasets, and assessment metrics is very similar to your research, which uses the Ember dataset and particular assessment metrics. This thorough overview provides a wider context for your study's contribution to the field by highlighting the ongoing challenges and current trends and echoing the advancements in machine learning for cybersecurity.

Kurt et al. [2] explore the critical field of early cyber-attack detection. It acknowledges the body of research on online detection techniques and outlier detection. It points out their drawbacks, including the need for flawless attack models and a reliance on sample-by-sample judgments. The research presents a novel method expressing the online attack/anomaly detection problem as a Partially Observable Markov Decision Process (POMDP). This viewpoint is novel because it abandons traditional detection techniques in favour of a more dynamic and unpredictable setting, which is frequently the case in actual cyberattack scenarios. The paper's main contribution is constructing a robust online detection algorithm that can be applied to any situation using the model-free Reinforcement Learning (RL) framework for POMDPs. This strategy is important in cybersecurity, particularly for vital infrastructure such as smart grids, where prompt and precise cyberattack detection is crucial. Since reinforcement learning (RL) deviates from conventional machine learning techniques in cyber-attack detection, its application in this context is especially intriguing. Numerical studies illustrate the effectiveness of this algorithm and highlight its potential for effective cyber-attack detection against smart grid systems. Because both of our studies centre on utilizing cutting-edge machine learning techniques for cybersecurity, your work is relevant to mine. This research examines the use of RL in a POMDP framework for detecting anomalies and attacks in smart grids, demonstrating the varied applications of ML in cybersecurity. In contrast, your study uses autoencoder algorithms for malware detection.

Ingre and Yadav [3] explored the vital topic of anomalous traffic detection on the Internet, which is covered in this abstract. This issue has become more significant as smart devices proliferate and technology advances. The specific area of focus is the performance of intrusion detection systems (IDS), which are crucial for preserving system security by spotting and warning about possible intrusions. This paper presents research on the effectiveness of the NSL-KDD dataset for intrusion detection using Artificial Neural Networks (ANN), a methodology consistent with the larger theme of utilizing machine learning techniques for cybersecurity. The paper's results are noteworthy because they thoroughly examine intrusion detection effectiveness. Regarding attack type, the results demonstrate promising accuracy in binary class and five-class classification tasks, with detection rates for the NSL-KDD dataset of 81.2% and 79.9%, respectively. This degree of performance shows how well ANNs identify different kinds of cyber threats, which is an important feature in the constantly changing field of cybersecurity. The research gains significant value from comparing this proposed scheme with existing methods, where it reportedly achieves higher detection rates in binary and multi-class classification problems. This is consistent with the goal of your research, which is to improve cybersecurity malware detection since both articles concentrate on increasing detection accuracy through various machine learning techniques. This paper investigates the use of ANNs for intrusion detection, demonstrating the adaptability and potential of machine learning methodologies in addressing various cybersecurity challenges. In contrast, your research uses autoencoders to classify malware.

Xu et al., [4] the study's unique contribution is Creating a 5-layer autoencoder-based model, especially for network anomaly detection tasks. This model results from a thorough investigation into different AE model performance indicators. A novel approach to data preprocessing, which attempts to lessen model bias by transforming and eliminating the most impacted outliers from the input samples, is a significant innovation in this model. This tackles the problem of imbalanced data, which is a prevalent issue in machine-learning applications. The study also highlights the importance of an efficient reconstruction error function in the model since this is essential for identifying abnormal or normal network traffic. Particularly notable is the suggested model's emphasis on feature learning and dimension reduction, which enhances detection accuracy and f1-score. According to the paper, the model performed better than comparable techniques when tested on the NSL-KDD dataset, obtaining the highest accuracy and f1-score at 90.61% and 92.26%, respectively. This development in AE models pertains directly to your research on network anomaly detection. It highlights the value of model optimization and data preprocessing in enhancing the effectiveness of ML techniques in cybersecurity tasks and sharing the use of autoencoders.

Pang et al. [5] tackle the relatively unexplored field of using deep learning for anomaly detection, emphasizing the shortcomings of current approaches. Learning new feature representations for downstream anomaly detection tasks is the main focus of current deep anomaly detection approaches. However, doing so leads to indirect optimization of anomaly scores, which causes inefficient use of data and less-than-ideal scoring results. Furthermore, because large-scale labelled anomaly data is scarce, these methods usually take an unsupervised learning approach, which makes it difficult to incorporate any prior knowledge that may be available, such as a few labelled anomalies, which is frequently the case in real-world applications. The study presents a novel anomaly detection framework that uses a technique known as "neural deviation learning" to learn anomaly scores end-to-end rather than relying on representation learning. This method is especially novel because it guarantees statistically significant deviations of anomaly scores from normal data, particularly in the upper tail of the distribution, by utilizing a small number of labelled anomalies and prior probability. This approach, which focuses on effectively using the already available data and improving the accuracy of anomaly scoring, is a major departure from conventional methods. The comprehensive results presented in the paper show that this new approach achieves significantly better anomaly scoring than current state-of-the-art methods while being more data-efficient. This study is especially pertinent to your advanced machine usage research.

Nasteski [6] thoroughly summarizes the major developments in supervised learning techniques made in the last ten years in machine learning. The focus of much machine learning research and innovation has shifted to supervised learning, which is distinguished by its reliance on annotated training data or "labels." Supervised learning is special because it can use class labels in the classification process. This feature has made it useful for many data and domains. The paper aims to provide an overview of the core ideas behind several supervised learning algorithms, which have evolved to become more complex and diverse. This review's main objective is to provide a comprehensive overview of machine learning techniques, emphasizing supervised approaches in particular. In doing so, the paper highlights the diversity and complexity of algorithms within this domain and advances a wider understanding of the evolution and current state of machine learning. This investigation is especially pertinent to your work, which also uses machine learning methods—especially autoencoders—for cybersecurity. The paper's emphasis on the principles and applications of supervised learning offers insightful information about the machine learning landscape as a whole. It also gives a contextual background that helps readers better understand how various methodologies, including those you used for your research, fit into the larger picture of AI and machine learning advancements.

Nissim et al. [7] tackle a major cybersecurity challenge: the quick development of new malware and the shortcomings of conventional antivirus programs that mostly use manually created signatures. There is a gap in detecting new and unknown malware because these tools are only effective against known malware instances and their comparable variants. To close this gap, antivirus providers gather suspicious files daily for information security professionals to analyze manually. However, this

is an inefficient and time-consuming process due to the large number of files. The use of machine learning algorithms and heuristics by antivirus vendors to lessen the manual workload is acknowledged in the paper; however, there is a significant "updatability gap" because these techniques lack the critical capability of daily updates. These results demonstrate the potential of cutting-edge AI techniques to address the constantly changing challenges of malware detection, and they are especially pertinent to machine learning and cybersecurity. While this research tackles the issue from the standpoint of active learning rather than autoencoders, it is consistent with your study in that it focuses on leveraging machine learning to improve malware detection. These new AL methods' emphases on efficiency and constant adaptation provide insightful information about possible developments in malware detection techniques. It points the way forward for promising future cybersecurity research and development.

Mughaid et al. [8] investigate the relationship between cybersecurity and Non-Orthogonal Multiple Access (NOMA) technology in 5G wireless communications, specifically focusing on wireless intrusion detection systems. Strong security systems will be more important as 5G, which is anticipated to be much faster than 4G, takes off to fend off internal and external network attacks. The creation of a NOMA simulator and the use of a dropping attack to produce a dataset for analysis are described in detail in the paper. Then, to identify wireless cyberattacks in 5G networks, the authors used a variety of machine learning (ML) and deep learning (DL) techniques, such as Decision Trees, K-Nearest Neighbors (KNN), Multi-class Decision Jungle, Multi-class Decision Forest, and Multi-class Neural Network. The use of various ML and DL techniques in this context demonstrates the potential of these technologies to improve network security, which is consistent with the overarching theme of addressing complex cybersecurity challenges with advanced computational methods such as the autoencoders in your research.

Ibor et al. [9] tackle significant flaws in current cyberattack prediction techniques, including poor prediction accuracy, high false positive rates, protracted training periods, and challenges in selecting hyperparameters to avoid underfitting or overfitting. These problems have made cyberattacks more frequent, emphasizing how much the current models need improvement. Recurrent neural networks (RNNs), one of the deep learning architectures used for cyberattack prediction, have difficulties with training and optimization and the vanishing and exploding gradient problem. The encouraging results demonstrate that AdacDeep not only outperforms other cutting-edge comparative models but also significantly improves F-Score (0.1–34.7%), prediction accuracy (0.22–35%), and false positive rate (0.1–35%). The field of cybersecurity benefits greatly from this research, especially in the area of machine learning applications for cyberattack prediction. It contributes fresh perspectives and methods that might enhance cyberattack prediction models' efficacy and efficiency. It supports the overarching objective of enhancing cybersecurity defences by applying cutting-edge AI and machine learning technologies.

Abu Al-Haija et al. [10] discuss the security issues surrounding the quickly developing Internet of Things (IoT) space, with an emphasis on how vulnerable IoT infrastructures are to different types of cyberattacks, such as Darknet or blackhole (sinkhole) attacks. These attacks exploit the endpoint IoT devices' constrained computing, storage, and communication capabilities. The paper highlights that a major source of unsolicited traffic, frequently indicating probes, backscatter, or misconfigurations, is Darknet address space, which is reserved and not meant for legitimate hosts. This study contributes substantially to IoT cybersecurity by illuminating how sophisticated cyber threats can be identified in IoT environments using sophisticated machine learning techniques. It highlights the ongoing need for innovation in cyber defence strategies. It is consistent with the larger theme of using AI and machine learning to improve cybersecurity measures, as demonstrated in your research on autoencoders for malware detection.

Wang et al. [11], to improve the robustness of machine learning (ML)-based malware detectors against adversarial malware examples (AMEs), this paper investigates a novel technique called LSGAN-AT. AMEs are essential to cybersecurity because they greatly influence how well malware detectors work. For robustness enhancement, the quality of AMEs is crucial, and Generative Adversarial Networks (GANs) have been employed in AME generation. However, insufficient optimization, mode collapse, and training instability plague the GAN-based AME generation techniques currently in use. Additionally, the RMD performs admirably in AME recognition, confirming the effectiveness of the LSGAN-AT method. The significance of this research lies in its ability to improve the resilience of machine learning-based malware detectors. It fits with the overarching theme of enhancing defence mechanisms against more complex cyber threats by employing cutting-edge machine learning and artificial intelligence techniques, like your research on autoencoders for malware detection.

Yerima et al. [12] tackle the urgent problem of the explosive rise in mobile malware, with an emphasis on Android in particular because of its open platform architecture and growing market share relative to other mobile smart device platforms. The advent of novel malware families for Android with sophisticated evasion techniques poses a substantial obstacle to established detection strategies. This paper proposes a novel parallel machine learning-based classification approach for early Android malware detection. In the larger context of cybersecurity and AI, the research's emphasis on utilizing a parallel combination of various machine learning classifiers for improved malware detection is especially pertinent. It is consistent with the themes of cutting-edge malware detection machine learning applications, like your autoencoder research. This study emphasizes how

critical it is to modify and advance detection techniques to combat malware's complex and ever-evolving nature, particularly on widely used platforms such as Android.

Yousefi-Azar et al. [13], a novel feature learning model tailored for cybersecurity tasks, is presented. It uses Auto-encoders (AEs) as a generative model to learn latent representations of different feature sets. The main emphasis is on the automatic learning and capture of semantic similarity among input features by AEs, which is essential in comprehending and detecting cybersecurity threats. Under this model, an AE takes a feature vector from cybersecurity phenomena and uses an abstract latent space to extract a code vector representing the semantic similarities between feature vectors. Two distinct cybersecurity tasks—malware classification and network-based anomaly intrusion detection—are used to evaluate the paper's methodology. Several classifiers are used to analyze the model on publicly accessible datasets for both tasks to obtain empirical validation. The results show some improvement over earlier techniques when evaluated using several relevant metrics. In particular, feature learning and representation are two areas where this research adds to the rapidly developing field of AI and machine learning in cybersecurity. Its compatibility with cutting-edge approaches—like the ones you employed in your study on autoencoders for malware detection—highlights the increasing significance of complex AI methods in creating more potent cybersecurity defences.

Tirumala et al. [14], the growing dependence on internet-based systems highlights the rapidly expanding field of malware analysis, which is the subject of this paper. The recent developments in artificial intelligence (AI), particularly data mining, and their uses in AI-based malware classification and detection systems are given special attention. Most AI-based malware analysis systems primarily rely on pre-existing malware datasets and are signature-based. This work assesses the effectiveness of a feature-based approach to malware classification that uses autoencoders, which is a major departure from traditional signature-based techniques. One important finding of the study is that feature-based stacked autoencoders, specifically a 5-layered model, can achieve a classification accuracy of 95.6%, an improvement of 11.6% over the signature-based system's 84.6% accuracy. This finding emphasizes how feature-based techniques, particularly autoencoders, can improve the precision and efficacy of malware detection systems. The paper's investigation of autoencoders in malware classification aligns with your findings, highlighting the importance of cutting-edge machine-learning methods in the continuous quest to strengthen cybersecurity defences.

Anderson and Roth [15], to fill a major gap in the information security machine learning community, this paper presents EMBER, an extensive benchmark dataset created for training machine learning models to detect malicious Windows portable executable files statically. The dataset comprises 900,000 training samples (300,000 malicious, 300,000 benign, and 300,000 unlabeled) and 200,000 test samples (100,000 malicious and 100,000 benign). The features were extracted from 1.1 million binary files. This dataset's size and ratio of benign to malicious samples are notable features that make it an invaluable tool for creating and evaluating malware detection models. A comparison between MalConv, a deep learning model for malware detection, and a baseline gradient-boosted decision tree model trained using LightGBM with default settings is one of the main use cases presented. The findings show that the baseline EMBER model performs better than MalConv, even without hyperparameter optimization. This result emphasizes how well the EMBER dataset works to support the creation of reliable malware detection models. The authors hope that similar to how to benchmark datasets have fueled advances in computer vision research, EMBER will spur more research in machine learning for malware detection. The release of EMBER is especially pertinent to your work on autoencoders and malware detection since it offers a large, well-rounded dataset that can be used to train and evaluate sophisticated cybersecurity machine-learning models.

## 3. Methodology

A subclass of neural networks called autoencoders is made to learn a condensed representation of datasets. Usually, they consist of two primary components: the encoder and the decoder. As per [1]; [2]; [4]; [10]. The input is compressed by the encoder into a latent-space representation, which the decoder uses to reconstruct the input data. An autoencoder can be mathematically represented as an approximation function $f(x)$ , where $x$ represents the input data in an attempt to approximate $x$ . The input vector is $x \in R^d$ And the encoded representation is $y \in R^p$ , where $p < d$ is typically the case. Here is one way to represent the encoding function $h$ :

$$h(x) = \sigma(Wx + b) \quad (1)$$

In this case, $\sigma$ represents an element-wise activation function, like the sigmoid, tanh, or ReLU. A bias vector is represented by $b$ and a weight matrix by $W$. $h(x)$Yields the encoded representation $y$ , also known as the code. The encoded data is mapped back to the original dimension space by the decoder function $g$ , which is defined as follows:

$$g\big(h(x)\big) = \sigma'(W'h(x) + b') \quad (2)$$

where $W'$ is the decoder weight matrix, $b'$ s the coder bias vector, an σ′ It is the activation function of the decoder, which may or may not be the same as σ. minimizing the difference between $x$ and $g(h(x))$, the autoencoder's objective, is typically achieved by employing a loss function, such as the mean squared error (MSE):

$$L\left(x, g(h(x))\right) = |x - g(h(x))|_2^2 \qquad (3)$$

Mean squared error is not the only option available, though. Other loss functions, such as binary cross-entropy, can also be used, particularly if the input data is in the form of binary vectors or matrices, depending on the distribution of the input data and the desired properties of the outputs:

$$L\left(x, g(h(x))\right) = -\sum_{i=1}^{d}\left[x_i \log\left(g(h(x))_i\right) + (1 - x_i)\log\left(1 - g(h(x))_i\right)\right] \qquad (4)$$

Additionally, regularization terms can be incorporated into the loss function to promote specific qualities in the learned representations, like robustness against noise or sparsity. To penalize non-sparse representations, for instance, the loss function could have a regularization term $R(h)$ added to it:

$$L\left(x, g(h(x))\right) = |x - g(h(x))|_2^2 + \lambda R(h) \qquad (5)$$

Where λ is a coefficient that controls the strength of the regularization. Stochastic gradient descent (SGD) or its variants, Adam, are optimization algorithms that are used to minimize the loss function by adjusting the parameters $W, W', b$ and $b'$ During the training process. Reconstruction error is essential to detect anomalies. An input can be labelled as anomalous if the error $|x - g(h(x))|_2^2$ for a given input, $x$ surpasses a predefined threshold. Setting a threshold that properly balances false positives and negatives is a crucial step that frequently calls for using a validation dataset of known anomalies. In this work, we develop an autoencoder customized for the Ember dataset, including unique architectural features and a training schedule. We go over the optimization procedure, activation function selection, and hyperparameter selection, which are crucial for the model to successfully identify malware as anomalies. Furthermore, the trade-offs between sensitivity and specificity are carefully considered when fine-tuning the reconstruction error threshold for anomaly detection, considering the operational context in which the model will be used.

### 3.1. Training Dataset

A noteworthy addition to the field of cybersecurity is the Endgame Malware Benchmark for Research (EMBER) dataset, which was created to make it easier to develop and assess machine learning-based malware detection systems. This extensive dataset, which includes characteristics taken from a sizable corpus of portable executable files, offers researchers a wide range of detailed information that reflects the intricate world of contemporary malware. With a focus on real-world scenarios, EMBER is organized to emphasize practical application. Labeled data designates benign and malicious samples. This is a crucial tool for improving cybersecurity defences because it permits the training and testing of machine learning models and acts as a standard against which to compare how well different methods identify possible threats.

**Table 1:** Overview of the EMBER Dataset Features

| Features | Description | Example |
|---|---|---|
| SHA-256 | File identifier | 0001a959869f... |
| Appeared | Appearance date | 2017-11 |
| Label | Classification | 1 |
| Histogram | Byte distribution | [18907, ..., 1849] |
| ByteEntropy | Entropy values | [0, ..., 25116] |
| Strings | String statistics | 1956, 10.28, 20113 |
| General | File attributes | 257024, 671744 |
| Header | Header specifics | I386, ... |
| Section | PE sections | UPX0, ... |
| Imports | Imported DLLs | KERNEL32.DLL, ... |
| Exports | Exported functions | el, el2, ... |

A wide range of features from portable executable files are available in the EMBER dataset, which trains machine-learning models to classify malware. Every entry in Table 1, [14], [15], the dataset—including the sample—contains the file's SHA-

256 hash, a timestamp identifying when it first appeared in the dataset, and a binary label designating whether the file is malicious (1) or benign (0). As a fingerprint of the file's content, the 'histogram' key in the dataset entry contains an array of values representing the distribution of byte values within the file. A byte entropy array is a useful tool for identifying encrypted or packed sections of files, a common feature of malware. Following the histogram, it distributes entropy values across different file segments. The statistics on printable strings in the file are part of the *strings* feature set, which provides information about the text-based content and can be a reliable indicator of the nature and purpose of the file. Additionally, the dataset offers comprehensive attributes, including file size, virtual size, and different file properties and structures under the "general" and "header" keys. These characteristics are essential for creating an executable profile that helps differentiate between normal software behaviour and irregularities that could indicate malicious intent. Altogether, the structured executable file representation of the EMBER dataset makes machine learning useful for well-informed and efficient malware detection. Another alternative dataset that can be useful is the NSL-KDD dataset, which, unlike the EMBER dataset, focuses more on the network aspect of malware anomaly detection. This is also a great aspect for testing malware activity and ensuring absolute security, as shown in papers [3]; [6]; [12].

### 3.2. ML Training

Our research has implemented significant improvements, as shown in Table 2.

**Data Preprocessing Enhancements:** One of the main components of the enhanced model is our improved data preprocessing method. Since cybersecurity data comes in various forms, we created special functions to translate complicated data types into a numerical format. This step is crucial to ensure the neural network receives data in the best format possible for processing and learning. For example, putting byte entropies and SHA-256 hashes, which are essential components of cybersecurity data, into numerical form enables a more accurate and nuanced interpretation by the model. This preprocessing stage helps the model identify subtle patterns that could point to malicious activity and enhance its capacity to learn from the data.

**Autoencoder Model Architecture:** Our model's autoencoder architecture is a major improvement over conventional design. We improved the model's capacity to capture and learn from complex data structures by adding dropout and dense layers with 256 and 128 units. Overfitting, a common problem in machine learning models where the model becomes too tailored to the training data and fails to generalize to new, unseen data, is something that the dropout layers play a crucial role in preventing. This intricate architecture is essential for identifying subtle abnormalities in cybersecurity data, which increases the model's accuracy in differentiating between benign and malicious activity.

**Training Process and Early Stopping Implementation:** An important advancement in our approach is the inclusion of Early Stopping in our training program. This is an essential technique to keep the model from overfitting, as it stops the training process when no more improvement in validation loss is seen. Since overfitting can seriously impair a model's performance on fresh data, Early Stopping is essential to maintaining our model's accuracy and efficacy in practical applications. A model for identifying malware must prioritize generalization because malware is ever-evolving and poses novel challenges.

**Comprehensive Model Evaluation:** We carefully assess the model's performance using a range of metrics after training. We use precision-recall curves, confusion matrices, and more conventional metrics like accuracy. These resources offer a more thorough comprehension of the model's functionality, particularly its capacity to differentiate between typical operations and possible security risks. A thorough evaluation is imperative to ensure the model's efficacy in practical applications and make necessary adjustments.

**Anomaly Detection Thresholding:** Our research is noteworthy in that it establishes a threshold for anomaly detection based on the reconstruction error of the model. This method is very helpful in locating dataset outliers, which could be signs of malware. It takes careful balance to set this threshold robust enough to prevent false positives and sensitive enough to identify real threats. The model's practical applicability in real-world cybersecurity scenarios is improved by this thresholding method, which is crucial for identifying potential threats promptly and accurately.

**Table 2:** Parameters Used in Model Training and Their Significance

| Parameter | Significance |
|---|---|
| Dense Layers (256, 128 units) | Increasing the model's complexity allows it to detect complex patterns in the data better. |
| Dropout (0.3) | Randomly dropping units ensures better generalization to new data and prevents overfitting. |
| Early Stopping (Patience: 10) | Stops training when validation loss does not improve, avoiding overfitting and conserving computing power. |

| | |
|---|---|
| Epochs (50) | Establishes how often the neural network will pass the complete dataset forward and backward. |
| Batch Size (64) | Determines how many samples must be examined before changing the internal model's parameters. |
| Validation Split (0.2) | Helps to prevent overfitting in the model by reserving a portion of the training data for validation. |
| Optimizer (Adam) | Establishes the optimization method that minimizes the loss function. |
| Loss Function (MSE) | Determines the model's error during training, which directs the training properly. |

### 3.3. ML Application

The sophisticated AI model, skilled at identifying malware through anomaly detection, expands its applicability outside of cybersecurity into several other industries, each customizing its fundamental functions to meet their requirements. The model's main job in the cybersecurity space is to find malware. Given the dynamic nature of malware, its proficiency in detecting novel and intricate cyber threats is essential. The model is especially good at identifying malware that might elude conventional detection techniques because of its capacity to learn from and recognize anomalies in data. Another important application is the detection of financial fraud. Sophisticated fraud schemes are common in the financial sector because of its extensive and intricate transaction networks. Here, the model's ability to identify anomalous patterns in transaction data is crucial for preventing and minimizing fraud and protecting financial assets. The model discovers a significant role for patient data monitoring in healthcare data analysis. Healthcare data anomalies may point to probable misdiagnoses, inaccurate data entry, or early disease symptoms. The AI model helps to ensure accurate patient care and early disease detection by identifying these anomalies.

The model is used in network security to detect anomalous network traffic that might indicate a security breach. Protecting network integrity requires the model's sophisticated detection capabilities, which are crucial given the growing sophistication of cyberattacks. The model is useful for monitoring operational data and equipment in industrial systems, especially manufacturing. Its capacity to foresee probable breakdowns or identify unexpected operational behaviours may result in better maintenance plans and shorter downtime. The use of the model in e-commerce and retail is centred on spotting odd buying trends or possible fraudulent activity. In a time when online retail fraud is on the rise, this capability is essential for businesses to safeguard their profits and clientele. The model's anomaly detection features benefit the intricate network of sensors and data inputs that make up smart city infrastructure. The model improves the effectiveness and security of smart city operations, whether for detecting issues with urban infrastructure or handling emergencies.

Detecting anomalies in sensor data is critical for the automotive industry, particularly regarding autonomous vehicles. Because autonomous car systems primarily rely on sensor data for navigation and decision-making, this application is essential to guaranteeing the security and dependability of those systems. The supply chain management industry uses the model to monitor processes and spot anomalies that might point to fraud or inefficiencies. This facilitates the supply chain's optimization, guarantees on-time delivery, and lowers the possibility of losses from fraud. Finally, the model is useful in the energy sector for tracking anomalous patterns in grid data that could point to malfunctions or inefficiencies. Assuring a steady and dependable energy supply is crucial for modern infrastructure, so this is especially important. The model's primary ability to learn intricate patterns and spot deviations from the norm is utilized in these industries, demonstrating its adaptability and versatility outside its original use in malware detection.

## 4. Simulation Analysis

Malware detection is still a major challenge in the quickly developing field of cybersecurity, requiring creative and flexible solutions. This simulation analysis focuses on applying autoencoder algorithms, a machine learning method well-known for its effectiveness in feature extraction and anomaly detection. The study aims to evaluate how well autoencoders identify malware, which has historically been difficult due to the dynamic nature of cyber threats. The analysis uses a dataset that includes various malware signatures to provide insights into the potential of autoencoders as a powerful tool in the ongoing fight against malware (Table 3).

- **True Negatives (TN):** 298,801 - This is the number of instances where the model correctly identified non-malware.
- **False Positives (FP)**: 587 - This is the number of instances where the model incorrectly identified non-malware as malware.
- **False Negatives (FN)**: 583 - This is the number of instances where the model failed to identify malware.
- **Precision:** 97.997% - This metric indicates the accuracy of the model in identifying malware, calculated as $TP/(TP + FP)$. A high precision means a low false positive rate.

- **Recall:** 98.801% - This measures the model's ability to identify all actual malware instances, calculated as $TP/(TP + FN)$. High recall indicates that the model effectively detects most of the malware.
- **F1-Score:** 98.3973% - This is the harmonic mean of Precision and Recall. An F1-Score near 100% indicates a well-balanced model between precision and recall.

**Table 3:** Model Performance Metrics

| Features | Description |
|---|---|
| True Positives (TP) | 300,029 |
| True Negatives (TN) | 298,801 |
| False Positives (FP) | 587 |
| False Negative (FN) | 583 |
| Precision (%) | 97.997 |
| Recall (%) | 98.801 |
| F1-Score (%) | 98.3973 |

High accuracy is a critical component in cybersecurity, as demonstrated by the performance metrics of your malware detection model with an autoencoder algorithm. The robustness of the model is indicated by the significant counts of True Positives (TP) and True Negatives (TN). True Positives are cases where the model has successfully detected malware in this context. Any malware detection system must have a high TP count because it indicates how well the model flags malicious software. Conversely, True Negatives are cases in which the model correctly detects files that are not malicious. Equally significant is a high TN count, which indicates that the model can accurately identify benign software as malware and prevent unwarranted interruption or hindrance of regular operations.

Furthermore, the model demonstrates remarkable Precision and Recall values, crucial metrics for assessing any detection system's efficacy. Measured as the ratio of True Positives (TP) to the total of TP and False Positives (FP), precision is an astounding 97.997% in Table 2. This high precision rate indicates that the model has a low false positive rate because it is generally accurate when predicting an instance to be malware. In malware detection, false positives can be problematic because they cause benign applications to be mistakenly classified as threats, raising unjustified concerns and possibly interfering with normal operations. Similarly, the Recall value (TP over the sum of TP and False Negatives (FN)) quantifies the model's capacity to identify every instance of malware. It stands at a remarkable 98.801%. With a high recall rate, the model is more likely to identify most malware and less likely to miss real threats. The F1-Score is especially notable, with a value of 98.3973%. This metric assesses a model's accuracy and is calculated as the harmonic mean of Precision and Recall. A model that achieves an F1-Score close to 100% is considered well-balanced, capable of detecting malware with high precision, and guaranteeing that real threats are not overlooked with high recall. It can be difficult to strike this balance, particularly in malware detection, where false negatives can cost very much. In cybersecurity, false negatives—when the model misses real malware—are a major worry. They stand for security lapses caused by malicious software that evades detection and may result in serious data breaches or system compromises. The low-performance numbers of FP and FN in your model highlight their dependability and appropriateness for real-world implementation in malware defence.

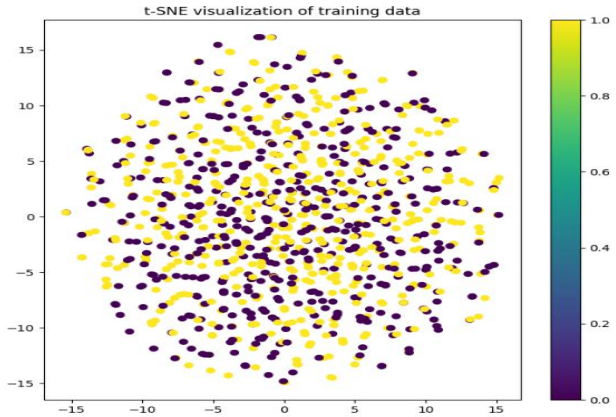


**Figure 1:** T-SNE Visualization output

The complex, multi-dimensional data related to your malware detection model is represented in a two-dimensional space by the scatter plot, which is the t-SNE visualization that is supplied in Figure 1. t-SNE is a great tool for visualizing possible groupings or clusters within the data because it is good at preserving the local structure of high-dimensional data. Each point in this plot represents a distinct data instance from your training set, and the similarity of their high-dimensional features is indicated by how close together the points are. A specific feature or dimension of the data, such as the probability that a data point is an outlier, the density of points in a region, or another continuous variable pertinent to the model, may be represented by the colour gradient, which varies from dark purple to bright yellow.

Looking at the plot, we see that the points show a gradient of density, with some areas more densely populated than others rather than forming distinct clusters. This pattern may indicate different confidence levels in the classification or a continuous feature that varies gradually over time. Such visualization can be especially informative for malware detection, providing hints about the data's underlying structure and how well the autoencoder algorithm may encapsulate the key elements of the malware signatures. The lack of distinct and well-defined clusters may also suggest that the dataset has intricate features that overlap and challenge the model. Though t-SNE is an effective visualization tool, it's vital to remember that the two-dimensional representation of results is an oversimplification, and understanding it necessitates carefully analyzing the model's goal and the data's characteristics.

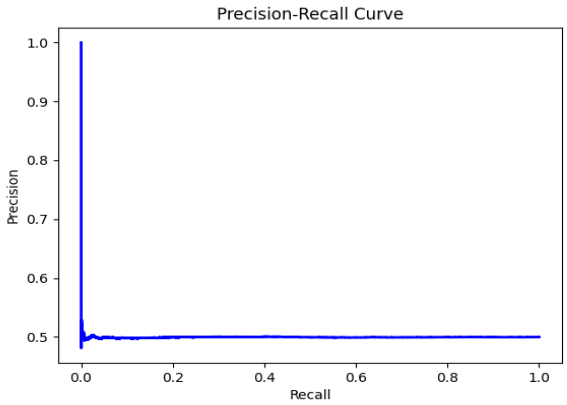


**Figure 2:** Precision-Recall Curve

Figure 2 shows a Precision-Recall curve, a graph showing how a binary classifier system's discrimination threshold changes about precision, the accuracy of the positive predictions, recall, or the model's capacity to locate all the pertinent cases within a dataset. The model achieves high precision, but only for a small fraction of the highest probability positives, as the curve's steep initial drop indicates; precision falls off dramatically as recall rises. This is typical of a model whose confidence rapidly declines when it tries to cover more positive cases but is very confident in a small set of predictions. This could imply that the model is highly accurate for a malware detection system when it is highly certain. However, as it attempts to identify more malware instances, its confidence in correctly identifying malware drops off quickly. This is not the ideal curve, which would maintain its high precision even as recall rises. This implies that even though the model has high precision, sustaining that precision at higher recall levels might be difficult. This could be a target for future model improvement.

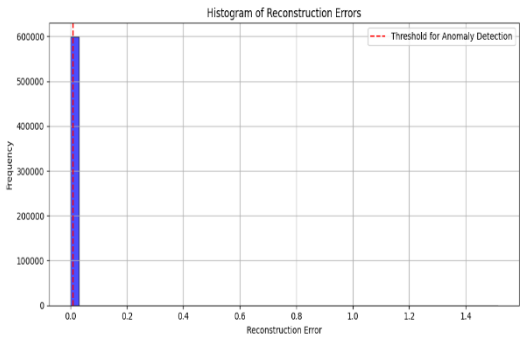


**Figure 3:** Histogram of Reconstruction Errors

The histogram in Figure 3, which has been provided, displays a histogram of reconstruction errors from an autoencoder model, which is probably utilized for malware anomaly detection. The reconstruction error, a gauge of the autoencoder's ability to reconstruct the input data, is represented by the x-axis. The frequency of each error level throughout the dataset is displayed on the y-axis. Most data points cluster close to the zero error, indicating that the model can reliably reconstruct most inputs. The red dashed line shows the threshold for anomaly detection; in this case, any occurrence with a reconstruction error above this line would be regarded as an anomaly or possibly malicious software. The threshold appears to be placed just beyond most of the data, indicating that only those occurrences with a noticeably higher error than the dataset's average will be flagged. The threshold must be set to balance the number of false positives, or benign cases mistakenly reported as malware, against the number of false negatives or instances of malware that are not detected.

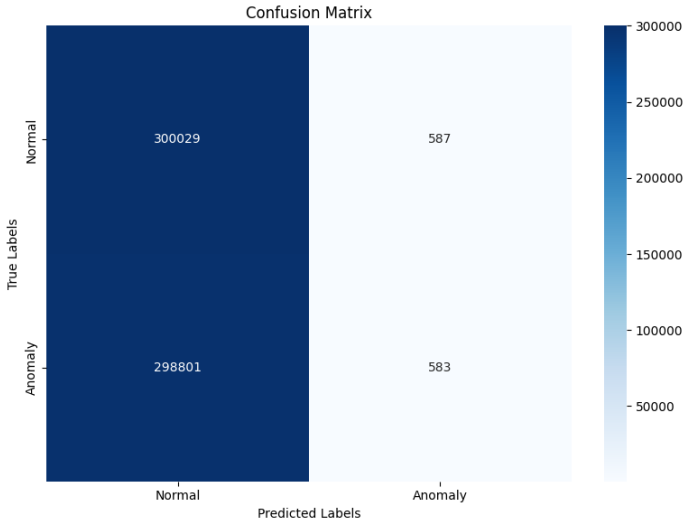


**Figure 4:** Confusion Matrix

A confusion matrix in Figure 4, a visual aid commonly used to evaluate a classification model's performance, is depicted in the image. The matrix compares the labels that the model predicts and the actual labels of the data. The label "Anomaly" in this matrix denotes malware or outliers, while "Normal" probably refers to benign occurrences. The large numbers on the diagonal from the top left to the bottom right of the matrix represent the 298,801 anomalies and 300,029 normal instances the model correctly identified. These represent the genuine advantages and disadvantages, correspondingly. The off-diagonal numbers represent the errors: 587 false positives (normal occurrences mistakenly classified as anomalies) and 583 false negatives (anomalies mistakenly classified as normal). The colour intensity represents Each cell's magnitude, with darker hues generally denoting higher values. According to this confusion matrix, there are comparatively few false positives and negatives and a high accuracy rate in differentiating between normal occurrences and anomalies.

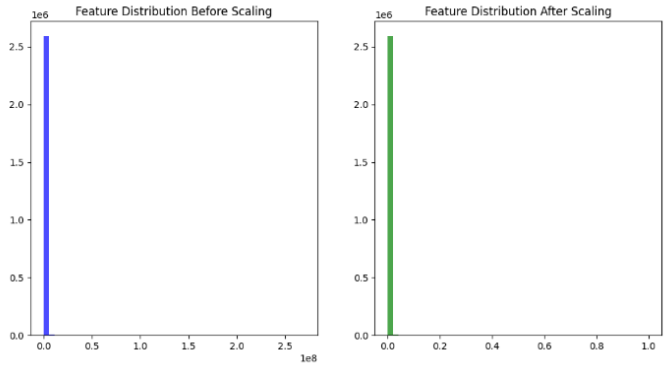


**Figure 5:** Feature Distribution

Two histograms in Figure 5 show the distribution of a feature or group of features in a dataset before and after scaling, which are shown in the image side by side. The feature(s) values are spread across a wide range in the "Feature Distribution Before

Scaling" graph on the left. Many machine learning models, especially those sensitive to the input data's scale, may find it challenging to interpret this data effectively. The x-axis's scale implies that the original data may have ranged in value from very small to very large. The same data is displayed post-scaling on the right in the "Feature Distribution After Scaling" histogram, where all values have been condensed into a much more manageable and uniform range, usually between 0 and 1. Standardization is a common preprocessing step to ensure that each input feature contributes equally to the model's learning process and that features with larger scales do not unduly influence the model. The consistency following scaling suggests that the data is now prepared for modelling, which may enhance the efficiency of a machine learning algorithm, particularly those that use gradient and distance learning techniques.

A high degree of efficacy and efficiency is demonstrated by thoroughly examining the numerous outputs and graphs associated with applying an autoencoder algorithm for malware detection. According to the preliminary metrics from the model performance statistics, the model can correctly identify malware with few false positives and false negatives, demonstrating high precision and recall. The F1-score, which is very close to the perfect 100%, indicates that the model keeps a stable balance between recall and precision, which is important for malware detection because misidentification can have a significant cost. These findings are further supported by the confusion matrix, which displays a high proportion of true positives and negatives and comparatively low numbers of false positives and false negatives—signatures of a trustworthy detection system.

The Precision-Recall curve and the t-SNE plot, among other visualizations, offer a deeper understanding of the model's behaviour. The subtle differences between various classes suggested by the t-SNE visualization may reflect the intricate nature of malware signatures. Despite its sharp decline, the model has a high precision rate when the confidence level is strict, as indicated by the Precision-Recall curve. The training-validation loss graphs and the histogram of reconstruction errors from the autoencoder point to a well-tuned model, with the error rates and loss values stabilizing—a sign of good generalization without overfitting. The preceding and subsequent feature distribution histograms demonstrate the significance and efficacy of data preprocessing in machine learning procedures. Standardizing the features after scaling removes any bias resulting from the different scales of the raw data. These artefacts indicate that the malware detection system is competent and well-designed. However, there is room for improvement in preserving high precision over a wider range of recall values and ensuring the model adjusts to the constantly changing landscape of cybersecurity threats.

## 5. Conclusion

With a focus on malware detection, this paper explores the creative use of autoencoder-based machine-learning techniques to improve cybersecurity. The results demonstrate how this technology can fundamentally alter how we approach cybersecurity issues. Our understanding of how to use autoencoders for malware detection has advanced significantly. By taking advantage of autoencoders' sophisticated feature learning capabilities, our model can distinguish between malicious executables and those that are not, and it does so quite well. Additionally, it adjusts to the evolving nature of cyber threats. The model's performance is assessed using confusion matrices, F1-Scores, and precision-recall curves, which validates its applicability in real-world scenarios and marks a breakthrough in cybersecurity measures. Even with these achievements, there is still space for development. Improving the model's accuracy at different recall levels is an important area that needs to be developed. Keeping up with the ever-evolving landscape of cybersecurity threats is a big challenge. Adding dynamic learning mechanisms and mixing various AI methods could significantly increase the model's efficiency and flexibility.

The potential applications of our model go beyond conventional cybersecurity realms. Its anomaly detection capabilities are valuable across multiple sectors, including financial fraud detection, healthcare data analysis, network security, industrial system monitoring, e-commerce, and smart city infrastructure. The model can identify minor deviations from normal patterns in these areas, bolstering operational security and efficiency. Our research contributes significantly to the broader field of cybersecurity, offering a practical solution to increasingly complex and sophisticated cyber threats. Employing machine learning tools like autoencoders signals a shift towards more proactive and intelligent cybersecurity systems. This is crucial when traditional security measures fall short against advanced, evolving cyber threats. In conclusion, our journey towards developing robust and intelligent cybersecurity is ongoing. This research represents a crucial step forward. By leveraging the power of autoencoders and machine learning, we pave new paths for protecting digital assets and infrastructures against complex cyber threats. As we continue to refine and enhance these technologies, the future of cybersecurity appears bright, with AI-driven solutions leading the charge. No Line breaks between paragraphs belonging to the same section.

**Acknowledgment:** Special thanks to my friends for their unwavering support and encouragement throughout the research process.

**Data Availability Statement:** The data for this study can be made available upon request to the corresponding author.

**Funding Statement:** This manuscript and research paper were prepared without any financial support or funding.

**Conflicts of Interest Statement:** The authors have no conflicts of interest to declare.

**Ethics and Consent Statement:** This research adheres to ethical guidelines, obtaining informed consent from all participants.

**References**

1. K. Shaukat, S. Luo, V. Varadharajan, I. A. Hameed, and M. Xu, "A Survey on Machine Learning Techniques for Cyber Security in the Last Decade," IEEE Access, vol. 8, no.12, pp. 222310-222354, 2020.
2. M. N. Kurt, O. Ogundijo, C. Li, and X. Wang, "Online Cyber-Attack Detection in Smart Grid: A Reinforcement Learning Approach," IEEE Trans. Smart Grid, vol. 10, no. 5, pp. 5174-5185, 2019.
3. B. Ingre and A. Yadav, "Performance Analysis of NSL-KDD Dataset Using ANN," in 2015 Int. Conf. Signal Process. and Commun. Eng. Syst., Guntur, India, pp. 92-96, 2015.
4. W. Xu, J. Jang-Jaccard, A. Singh, Y. Wei, and F. Sabrina, "Improving Performance of Autoencoder-Based Network Anomaly Detection on NSL-KDD Dataset," IEEE Access, vol. 9, no.10, pp. 140136-140146, 2021.
5. G. Pang, C. Shen, and A. van den Hengel, "Deep Anomaly Detection with Deviation Networks," in Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, New York, United States, 2019.
6. V. Nasteski, "An Overview of the Supervised Machine Learning Methods," Horizons, vol. 04, no.12, pp. 51-62, 2017.
7. N. Nissim, R. Moskovitch, L. Rokach, and Y. Elovici, "Novel active learning methods for enhanced PC malware detection in Windows OS," Expert Systems with Applications, vol. 41, no. 13, pp. 5843–5857, 2014.
8. A. Mughaid, S. AlZu'bi, A. Alnajjar, E. AbuElsoud, S. El Salhi, B. Igried, and L. Abualigah, "Improved Dropping Attacks Detecting System in 5G Networks Using Machine Learning and Deep Learning Approaches," Multimedia Tools and Applications, vol. 82, no. 9, pp. 13973–13995, 2023.
9. A. E. Ibor, F. A. Oladeji, O. B. Okunoye, and C. O. Uwadia, "Novel Adaptive Cyberattack Prediction Model Using an Enhanced Genetic Algorithm and Deep Learning (AdacDeep)," Inf. Security J.: Global Perspective, vol. 31, no. 1, pp. 105-124, 2022.
10. Q. Abu Al-Haija, M. Krichen, and W. Abu Elhaija, "Machine-Learning-Based Darknet Traffic Detection System for IoT Applications," Electronics, vol. 11, no. 4, p. 556, 2022.
11. J. Wang, X. Chang, and Y. Wang, R.J. Rodríguez, J. Zhang, "LSGAN-AT: Enhancing Malware Detector Robustness Against Adversarial Examples," Cybersecur., vol. 4, no. 38, p. 38, 2021.
12. S. Y. Yerima, S. Sezer, and I. Muttik, "Android Malware Detection Using Parallel Machine Learning Classifiers," in 2014 8th Int. Conf. Next Generation Mobile Apps, Services Technol., Oxford, United Kingdom, pp. 37-42, 2014.
13. M. Yousefi-Azar, V. Varadharajan, L. Hamey, and U. Tupakula, "Autoencoder-Based Feature Learning for Cyber Security Applications," in 2017 Int. Joint Conf. Neural Networks (IJCNN), Anchorage, United States of America, pp. 3854-3861, 2017.
14. S. S. Tirumala, M. R. Valluri, and D. Nanadigam, "Evaluation of Feature and Signature Based Training Approaches for Malware Classification Using Autoencoders," in 2020 Int. Conf. Commun. Syst. & Netw. (COMSNETS), Bengaluru, India, pp. 1-5, 2020.
15. H. S. Anderson and P. Roth, "Ember: An Open Dataset for Training Static PE Malware Machine Learning Models," arXiv preprint arXiv:1804.04637, 2018, Pre-print.