engineering proceedings

MDPI

Proceeding Paper

Enhancing Insider Malware Detection Accuracy with Machine Learning Algorithms †

Md. Humayun Kabir *, Arif Hasnat, Ahmed Jaser Mahdi, Mohammad Nadib Hasan, Jaber Ahmed Chowdhury and Istiak Mohammad Fahim

Department of Computer and Communication Engineering, International Islamic University Chittagong, Kumira Chattogram 4318, Bangladesh; arifhasnat83@gmail.com (A.H.); jasermahdi@gmail.com (A.J.M.); nadibhasan.gtu.in@gmail.com (M.N.H.); jaberahmediiuc@gmail.com (J.A.C.); istiakfahim94@gmail.com (I.M.F.)

* Correspondence: mdhkrrabby@gmail.com; Tel.: +880-151-528-6984

† Presented at the 10th International Electronic Conference on Sensors and Applications (ECSA-10), 15–30 November 2023; Available online: https://ecsa-10.sciforum.net/.

Abstract: One of the biggest cybersecurity challenges in recent years has been the risk that insiders pose. Internet consumers are susceptible to exploitation due to the exponential growth of network usage. Malware attacks are a major concern in the digital world. The potential occurrence of this threat necessitates specialized detection techniques and equipment, including the capacity to facilitate the precise and rapid detection of an insider threat. In this research, we propose a machine learning algorithm using a neural network to enhance malware detection accuracy in response to insider threats. A feature extraction, anomaly detection, and classification workflow are also proposed. We use the CERT4.2 dataset and preprocess the data by encoding text strings and differentiating threat and non-threat records. Our developed machine learning model incorporates numerous dense layers, ReLU activation functions, and dropout layers for regularization. The model attempts to detect and classify internal threats in the dataset with precision. We employed random forest, naive Bayes, KNN, SVM, decision tree, logical regression, and the gradient boosting algorithm to compare our proposed model with other classification techniques. Based on the results of the experiments, the proposed method functions properly and can detect malware more effectively and with 100% accuracy.

Keywords: cybersecurity; insider threat; malware detection; machine learning

Citation: Kabir, M.H.; Hasnat, A.; Mahdi, A.J.; Hasan, M.N.; Chowdhury, J.A.; Fahim, I.M. Enhancing Insider Malware Detection Accuracy with Machine Learning Algorithms. *Eng. Proc.* **2023**, *58*, 104. https://doi.org/10.3390/ecsa-10-16234

Academic Editor: Francisco Falcone

Published: 15 November 2023

1. Introduction

Information is one of today's most precious but vulnerable resources. Most organizations and people in the modern world manage and safeguard their digital content via computer networks and information technologies. As a consequence, there is a rise in cybersecurity threats such as identity theft, hacking, and malware attacks. Malware is used in many of these attacks [1]. Malware, also known as malicious software, consists of programming (code, scripting, dynamic content, and other applications) intended to obstruct or prevent operation, gather data that might be used for exploitation or compromise privacy, obtain illegal access to system resources, or engage in other malicious acts [2]. Infrastructure is always in danger of serious harm from sophisticated malware that is always changing [3]. Threats nowadays are conducted utilizing modern technologies which makes them difficult to identify. Therefore, it is obvious that the need for an intelligent system that can recognize new malware by analyzing the structure of system operations produced by malware as well as a suitable mechanism to recognize infected files is essential. Traditional methods of identifying insider malware frequently fail to precisely identify malicious activities carried out by authorized users. The growing environment of cybersecurity necessitates sophisticated techniques that are capable of recognizing unusual user behavior and differentiating it from authorized actions taken by users. In previous related

works, researchers used several techniques such as DCNN [4,5], static malware-analysis [6], CNN [7], autoencoder network with a grey-scale image representation [8], Visual-AT [9], etc., to identify malware.

This study aims to investigate and evaluate various machine learning algorithms to enhance the precision of insider malware detection. By employing multiple machine learning techniques, anomaly detection, and datasets, including system logs and user behavior data, we intend to develop a robust and scalable solution capable of identifying malicious insider activities. In the present research, we propose a neural network-based machine learning algorithm to improve malware detection accuracy. First, we normalized the CERT 4.2 dataset. Then we designed a neural network model with dense layers and optimizers to identify malware. The effectiveness of our proposed method was then evaluated by comparing it to methods from recent research.

To summarize, we contribute the following in this paper:

1. We propose a reliable and effective machine learning-based method for enhancing the accuracy of infiltrator malware identification.
2. We developed a neural network model with dense layers and optimizers to detect malware.
3. For the purpose of evaluating the efficacy of our proposed method, we contrasted our suggested method with methods used in recent research.

The remaining sections of the paper are organized as follows. In Section 2, a summary of current insider threat detection methodologies is provided. In Section 3, we present our suggested malware detection method. Finally, Section 4 is allocated to the analysis of the results obtained from the proposed approach.

2. Research Background

As described previously in this paper, malware is one of the most significant security risks on the Internet right now. Researchers have developed numerous methods for detecting malware. A hierarchical paradigm was suggested by [3] to find security risks or events in real time. This study examined the challenges faced by basic, statistical analysis, conventional machine learning, and deep learning methods. The authors of [4] provided a framework to quickly identify a user's good and bad behavior. In their ensemble learning-based system for detecting insider threats, they recommend over bootstrap sampling, which reduces overfitting caused by sample imbalance. The authors used employee resource access patterns from a benchmark dataset in [5]. They transformed them into 1-D feature vector grayscale images and used DCNN to identify malicious insiders by seeing odd patterns. The idea of determining an executable's maliciousness using a brief overview of behavioral data is presented in [6]. They used static malware analysis. The authors of [7] used a convolutional neural network to identify and classify images by automatically extracting the characteristics of the malware images. The authors of [8] proposed identifying malware using a deep learning autoencoder network and a grey-scale image of the program. Niket et al. [9] used trained deep learning image recognition models to classify malware binaries as images; their results showed that that DL models generalize data better than k-NN. The authors introduced Visual-AT, the first general machine learning (ML)-based visualization technique, to identify malware and its variants [9]. It uses two ML algorithms and transformed picture data to identify and analyze difficult-to-identify malware and variants using AT. The authors of [10] suggested an image-based insider threat detector through geometric transformation (IGT), which turns unsupervised anomaly identification into supervised image classification. Many malware detection models were examined neural network malware detection is more efficient and accurate than other methods, according to the findings.

3. Methodology

This paper explores neural network classification techniques for insider threat detection in order to generate new cybersecurity solutions. The CERT4.2 dataset is simulated

event log records that represent activities in an organization's computer system. Our work focuses on this dataset in order to gain insights into computer behaviors and enhance security measurements. Here, we develop an innovative machine learning model to protect against potential threats.

To be inclusive in the neural network model we develop, we extract and process the information in a way that organizes it with the variables we need as "features". We ensure that all text strings are encoded into integers. The distinction between insider threat and non-threat rows (true positives and true negatives) must also be made. We used the pre-processing techniques to extract the records of true negatives from three complementary CSV files. We first added all the relevant records and then chose a few non-threat true-negative recordings from the R1 dataset. We now had a combined CSV file containing our threats and non-threat baseline. We added a new column representing a binary representation of true or false, 1 or 0, to distinguish between true and false occurrences.

We also manually converted the date and time provided field into Unix epoch time as part of the dataset preparation stage; a manual conversion method was used to determine the Unix epoch time from the supplied date and time values. This demonstration was done to show how the date field might be converted into a big integer, creating a new column for Unix epoch time. A new column was generated in the original spreadsheet by referencing the "scratch" sheet in order to get the intended result. The formula used in this context is expressed as follows:

$$NC = (C2 - DATE(1970, 1, 1)) * 86400$$

Spreadsheet software, such as Microsoft Excel and Google Sheets, uses the formula to estimate the total number of seconds that have elapsed since 1 January 1970, with respect to a specified date. This approach is in accordance with established norms for managing timestamps in the world of computing. Following up, we encoded the vector column and feature set column mapping; in our last manual pre-processing tasks, we format the CSV into "categorize" by label encoding the data. We mapped the discrete set of vectors for the record of interest in Excel (the value for HTTP is 0, the value for email is 1, and the value for the device is 2). In conclusion, we manually performed label encoding in Excel for a finite set of vectors for the vector column and feature set column map. From the CMU CERT4.2 dataset, the user data usually falls under two categories: malicious and legitimate users. The following illustrates the feature set applied in the proposed method.

Figure 1 shows the suggested insider threat machine learning model feature set. Several features improve the model's ability to detect hostile behaviors in an organization.

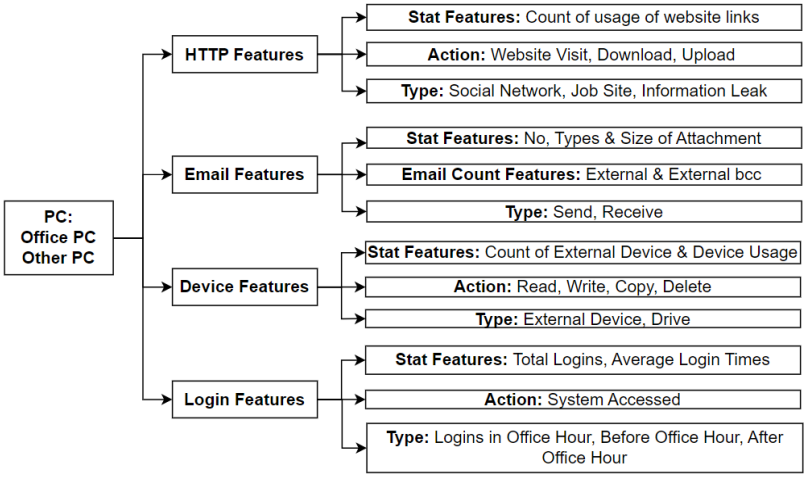

Figure 1. Feature set of the proposed method.

Our technique required splitting the preprocessed CMU CERT4.2 dataset into training and testing subsets. A two-pronged approach is needed for training, testing, and assessment. Machine learning begins with training models on the training set, a carefully selected collection of data to introduce the model. This phase teaches the model to identify data patterns, correlations, and features that signal malware presence or absence. After training, the models are tested on a separate dataset reserved for assessment. This testing set was separated from the training set to expose the models to new conditions.

4. Model Evaluation

We built a neural network classification model for detecting and classifying insider threat from the dataset. In our model, the input layer receives the dataset's features, and each feature corresponds to a node in the input layer. Then, in the hidden layer section, we add a dense layer to the model. First, we have selected the dense layer as 32. To avoid the vanishing gradients issue and inject non-linearity into our model, we employed the rectified linear unit (ReLU) activation function. Figure 2 presents a concise overview of a sequential Keras model. The model consists of three dense layers, each with 32, 16, and 8 units correspondingly. This is followed by a dropout layer and a final dense layer with a single unit. The model has a cumulative count of 109,089 parameters that are eligible for training.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 32)                108416

 dropout (Dropout)           (None, 32)                0

 dense_1 (Dense)             (None, 16)                528

 dropout_1 (Dropout)         (None, 16)                0

 dense_2 (Dense)             (None, 8)                 136

 dropout_2 (Dropout)         (None, 8)                 0

 dense_3 (Dense)             (None, 1)                 9

=================================================================
Total params: 109,089
Trainable params: 109,089
Non-trainable params: 0
```

Figure 2. Summary of our proposed model.

In order to address the problem of overfitting, a technique known as dropout is used during the training phase. This technique involves randomly setting a portion of the neuron outputs in the hidden layer to zero. The dropout parameter, which is set to 0.5 in this case, signifies that 50% of the inputs will be randomly assigned a value of zero during training. Subsequently, two more dense layers were included in the model, consisting of 16 and 8 neurons, using the rectified linear unit (ReLU) activation function. The ultimate dense layer comprises a solitary neuron, serving the purpose of discerning between malicious and benign samples in a binary classification test. To do this, the sigmoid activation function is used. In brief, the model consists of many densely connected layers that use rectified linear unit (ReLU) activation functions, which are alternated with dropout layers to enforce regularization. The activation function used in the output layer for binary classification is sigmoid.

Figure 3 describes the sequential operations included in the machine learning model suggested for detecting insider threats. The following procedural stages delineate the systematic methodology by which the model evaluates data and formulates educated choices about possible insider threats.

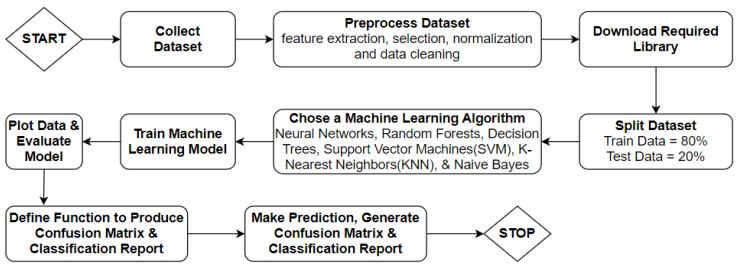

Figure 3. Procedures of the proposed model.

5. Results and Discussion

Table 1 displays the outcomes of applying our proposed neural network model, random forest, naive Bayes, gradient boosting, KNN, SVM, decision tree, and the logical regression algorithm to the CERT4.2 dataset.

$$\text{Accuracy(A)} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \quad \text{Precesion(P)} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall(R)} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad \text{F1 Score(F1)} = \frac{2 \times \text{Recall} \times \text{Precesion}}{\text{Recall} + \text{Precesion}}$$

Table 1. The results of classification techniques.

Classifier	Training (75%)				Training (80%)			
	A	**P**	**R**	**F1**	**A**	**P**	**R**	**F1**
Our Proposed Method (neural network using Adam optimizer)	100	100	100	100	100	100	100	100
Neural Network (using Adagrad optimizer)	90	91	90	89	99	99	99	99
Random Forest	99.64	99.63	99.65	99.64	99.73	99.74	99.75	99.74
Naive Bayes	61.42	61.42	100	76.10	61.66	61.71	100	76.87
Gradient Boosting	99.01	99.11	99.74	99.99	99.2	99.19	99.79	99.17
KNN	99.45	99.17	99.17	99.10	99.59	99.29	99.29	99.40
SVM	99	99	99	99	99.10	99	98	99
Decision Tree	99.08	99.06	98.67	99.01	99.15	99.09	98.97	99.19
Logistic Regression	61	61	94.5	76	61	61	96.18	76

As per Table 1, the performance of the proposed model is superior to that of the other machine learning algorithms. Compared to the other algorithms, the proposed neural network model provides significantly improved precision and recall while maintaining an accuracy level of 100%. The AUC score of 1.00 that our proposed method achieved is excellent. The categorization outputs derived from several machine learning algorithms and methodologies were used to detect insider threats.

The visual representation depicts the disparities in performance seen across different methods, hence facilitating the identification of the most productive way for properly discerning between normal actions and actions of malintent.

Figure 4 visually represents the confusion matrix related to the suggested machine learning approach for detecting insider threats.

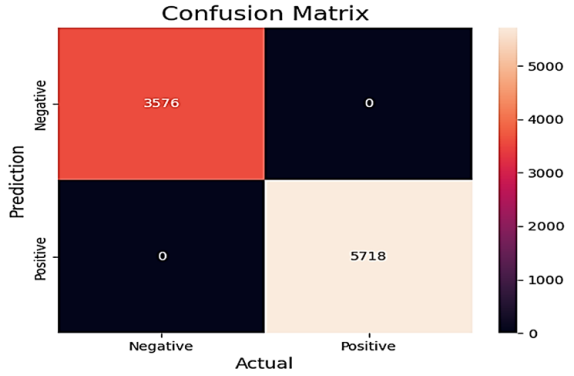

Figure 4. Confusion matrix of our proposed model.

Figure 5 illustrates the fluctuating patterns of accuracy and loss during the training phase of our proposed machine learning model designed to detect insider threats. The

neural network model that we suggested has shown remarkable performance across various critical criteria, demonstrating its durability and dependability in the context of identifying insider threats.

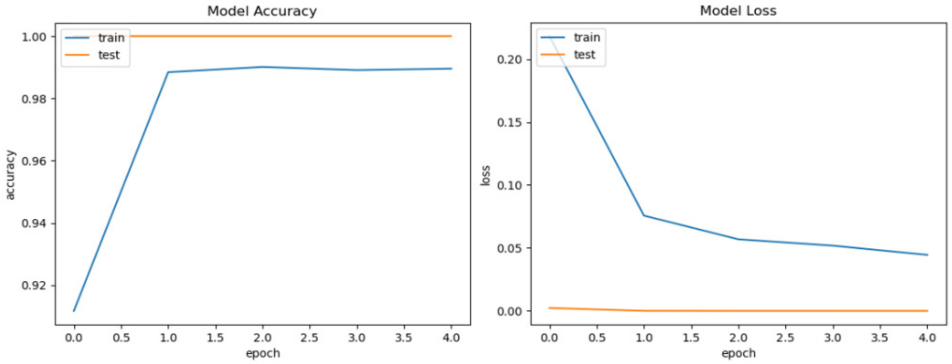

Figure 5. Our proposed model's accuracy and loss.

Number of training samples: 14,868 for test-size 20%. Table 2 depicts the training hyperparameters associated with the proposed methodology. The model is trained using the Adam optimizer, which is a commonly used choice for developing deep learning models. The batch size is specified as 32, suggesting that the model is updated after processing every 32 training instances. The learning rate used in this study is purposely fixed at 0.001 to address the potential issue of overfitting the training data. The use of a momentum coefficient of 0.9 enhances the expeditious convergence of the model towards an ideal solution. The model undergoes training for a standard duration of five epochs, which aligns with the conventional practice in the training of neural networks.

Table 2. The training hyperparameters for the suggested approach.

Hyperparameter	Values
Optimizer	Adam
Batch Size	32
Rate of Learning	0.001
Momentum	0.9
Epoch	5
Training Steps	2325
Activation Function	Sigmoid, ReLu

The performance assessment of existing approaches using CERT 4.2 is shown in Table 3. Our model for detecting internal threats outperforms all others. Our model has a validation accuracy of 100% and a training accuracy of 99%, versus 94% for the DCNN model [5], and it performs better than the 99.2% accuracy of ensemble learning (Bootstrap) [4] with a validation accuracy of 100%. Our model's 100% validation accuracy outperforms the 90% random forest model [6]. Our method detects and classifies insider threats considerably more precisely.

Table 3. Performance evaluation concerning current methods using CERT 4.2.

Model	Accuracy
DCNN [5]	94%
Ensemble Learning (Bootstrap) [4]	99.2%
Random Forest [6]	90%
Our Proposed Method	100%

6. Conclusions

In conclusion, this paper proposed a neural network-based machine learning algorithm to enhance the detection accuracy of infiltrator malware. Using the CERT4.2 dataset, the research effectively demonstrated the efficacy of the proposed method. Other classification techniques such as random forest, naive Bayes, KNN, SVM, decision tree, logical regression, and gradient boosting were outperformed by the proposed algorithm's 100% accuracy in detecting insider threats. A neural network-based approach, multiple dense layers, ReLU activation functions, dropout layers, feature extraction, anomaly detection, and classification workflow contribute to the algorithm's high accuracy and efficacy. The algorithm obtains a greater comprehension of internal malware patterns by encapsulating text sequences and preprocessing the data.

Author Contributions: Conceptualization: M.H.K.; methodology: M.H.K., A.H. and A.J.M.; software: A.H., A.J.M. and I.M.F.; formal analysis: M.H.K., M.N.H., A.H. and A.J.M.; writing—original draft preparation: A.H., A.J.M., J.A.C. and I.M.F.; writing—review and editing: M.H.K., A.H., A.J.M., J.A.C. and M.N.H.; supervision: M.H.K. and M.N.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Baset, M. Machine Learning for Malware Detection. Ph.D. Dissertation, Heriot Watt University, Edinburgh, UK, 2016.
2. Virus and Malware Removal—Wickenburg Computers—Fixing Your Computer Problems. (n.d.). Available online: https://www.wickenburgcomputers.com/services/virus-and-malware-removal (accessed on 1 October 2023).
3. Tayyab, U.-e.-H.; Khan, F.B.; Durad, M.H.; Khan, A.; Lee, Y.S. A Survey of the Recent Trends in Deep Learning Based Malware Detection. *J. Cybersecur. Priv.* **2022**, *2*, 800–829. [CrossRef]
4. Zhang, C.; Wang, S.; Zhan, D.; Yu, T.; Wang, T.; Yin, M. Detecting Insider Threat from Behavioral Logs Based on Ensemble and Self-Supervised Learning. *Secur. Commun. Netw.* **2021**, *2021*, 4148441. [CrossRef]
5. Gayathri, R.G.; Sajjanhar, A.; Xiang, Y. Image-Based Feature Representation for Insider Threat Classification. *Appl. Sci.* **2020**, *10*, 4945. [CrossRef]
6. Noever, D. Classifier Suites for Insider Threat Detection. *arXiv* **2019**, arXiv:1901.10948.
7. Rhode, M.; Burnap, P.; Jones, K. Early-stage malware prediction using recurrent neural networks. *Comput. Secur.* **2018**, *77*, 578–594. [CrossRef]
8. Cui, Z.; Xue, F.; Cai, X.; Cao, Y.; Wang, G.; Chen, J. Detection of Malicious Code Variants Based on Deep Learning. *IEEE Trans. Ind. Inform.* **2018**, *14*, 3187–3196. [CrossRef]
9. Xing, X.; Jin, X.; Elahi, H.; Jiang, H.; Wang, G. A Malware Detection Approach Using Autoencoder in Deep Learning. *IEEE Access* **2022**, *10*, 25696–25706. [CrossRef]
10. Liu, X.; Lin, Y.; Li, H.; Zhang, J. A Novel Method for Malware Detection on ML-based Visualization Technique. *Comput. Secur.* **2019**, *89*, 101682. [CrossRef]