INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

Employing Incremental Learning for the Detection of Multiclass New Malware Variants

Mohammad Eid Alzahrani[1]*

1 Department of Computer Science, Faculty of Computing & Information, Al-Baha University, Al-Baha, Saudi Arabia

OPEN ACCESS

Received: 12-11-2023

Accepted: 07-02-2024

Published: 27-02-2024

Citation: Alzahrani ME (2024) Employing Incremental Learning for the Detection of Multiclass New Malware Variants. Indian Journal of Science and Technology 17(10): 941-948. https://doi.org/10.17485/IJST/v17i10.2862

* **Corresponding author**.

meid@bu.edu.sa

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Background/Objectives: The study aims to achieve two main objectives. The first is to reliably identify and categorize malware variations to maintain the security of computer systems. Malware poses a continuous threat to digital information and system integrity, hence the need for effective detection tools. The second objective is to propose a new incremental learning method. This method is designed to adapt over time, continually incorporating new data, which is crucial for identifying and managing multiclass malware variants. **Methods:** This study utilised an incremental learning technique as the basis of the approach, a type of machine learning whereby a system retains previous knowledge and builds upon the information from the newly acquired data. Particularly, this method is suitable for tackling mutating character of malware dangers. The researchers used various sets of actual world malwares for evaluating the applicability of these ideas which serves as an accurate test environment. **Findings:** The findings of the research are significant. We utilizing 6 different datasets, which included 158,101 benign and malicious instances, the method demonstrated a high attack detection accuracy of 99.34%. Moreover, the study was successful in identifying a new category of malware variants and distinguishing between 15 different attack categories. These results underscore the effectiveness of the proposed incremental learning method in a real-world scenario. **Novelty:** This research is unique because of the novel use of a tailored incremental learning technique for dealing with dynamic threat environment of malwares. However, with a new threat they cannot be so well adapted using traditional machine learning methods. On the other hand, the technique put forward in this paper facilitates continuous learning that can be modified to match different types of malicious software as they develop. The ability to evolve and adapt is an important addition to current cybersecurity practices that include malware identification and classification.

Keywords: Cybersecurity; Malware Detection; Incremental learning

1 Introduction

Malware's growing prevalence has raised serious security issues for computer systems and networks throughout the globe [1]. Malware is a subset of harmful software that targets computer systems to cause disruption or harm, steal confidential data, or gain unauthorized access. Viruses, trojans, worms, and ransomware are just a few examples of malware [2,3]. Malware variant creation has grown significantly, making it difficult to identify and categorize them precisely. Malware detection and classification are essential for ensuring computer system security. Machine learning methods have traditionally been used to identify and categorize malware. Nonetheless, these methods have trouble dealing with malware's ongoing evolution [4]. Existing machine learning algorithms may have difficulty correctly detecting and classifying new malware variants as they are developed and released [5].

A machine learning approach called incremental learning has shown considerable promise for overcoming the difficulties in identifying and categorizing malware. A system may learn from new data while maintaining knowledge from past data in incremental learning. This method may assist in regularly adding fresh data to the system's knowledge base, enabling it to adapt to new malware strains and gradually increase accuracy [6]. The issue of malware detection has been addressed using a wide variety of machine learning models, such as decision trees, support vector machines, and neural networks. These methods have a high success rate for accurately identifying and categorizing malware. They struggle, however, to keep up with the new malware varieties that are continuously being created [7].

The use of incremental learning has shown significant promise for overcoming the difficulties in identifying and categorizing malware [8]. A system may continually learn from and respond to new data in incremental learning, which helps it gain accuracy over time. Many studies have suggested incremental learning-based methods for malware detection in recent years [9–11]. Incremental learning enables a model to constantly learn from and adjust to fresh data [12]. Since that new malware variants are continually being developed and disseminated, this method is especially helpful for detecting and identifying malware families. The ability to learn from fresh data without having to retrain the model entirely is one of the main benefits of incremental learning [13]. The ongoing development and dissemination of new malware variants are significant in the context of malware detection. The retraining of a model on all the new data is required by traditional machine learning approaches, which may be time-consuming and computationally costly [14]. The ability to recognize new malware families is another benefit of incremental learning. Identifying new malware families using traditional machine learning algorithms may be challenging, which depend on a predefined set of attributes to identify malware. On the other hand, incremental learning enables a model to pick up new characteristics as it comes across fresh data. The likelihood that the model will be able to identify new malware families increases [15].

The research contribution in this study focuses on the potential of incremental learning-based machine learning methods for efficient and accurate malware detection. These methods enable the system to learn from new data while retaining past knowledge, adapt to new malware strains, and recognize new malware families, which is challenging for traditional algorithms.

The paper is organized as follows: Section 2 presents the related work. Section 3 discusses the Incremental Model for detecting multiclass malware on multiple datasets. In Section 4, the proposed model, along with the algorithm, is presented. Section 5 includes the experimental setup, results, and analysis. Section 6 concludes the work.

The authors propose a framework for the automatic analysis of malware behaviour using machine learning [16]. They suggest a method for analyzing the behaviour of mobile malware by using behaviour classification and self-learning data mining. This approach can identify the malicious network behaviour of unknown or metamorphic mobile malware [17]. To ensure continuous learning, the study recommends incremental learning as a useful approach to update the learning data continuously. The study suggests a support vector machine based on an incremental learning method [18] for malware detection. A novel system is proposed by authors in the study [19] for automatically detecting malware intrusions in mobile devices to improve security. The study by authors [20] presents an efficient method for detecting malware and accurately identifying the corresponding malware family. A study [21] proposes a new approach for malware detection based on binary visualization and self-organizing incremental neural networks. The study by authors [22] proposes an effective encrypted malware traffic detection method that maintains a sufficient performance level by periodic updates using machine learning. The authors in [23] propose a new malware detection framework called HAWK for evolutionary Android applications to address the challenges posed by evolving camouflage in malware. The study [24] examines 11 continuous learning techniques for three malware tasks covering common incremental learning scenarios, including task, class, and domain incremental learning.

1.1 Incremental Model for Detection of Multiclass Malware on Multiple Datasets

In incremental learning, model parameters are modified as fresh data become available. This enables the model to adapt to shifting data distributions and improve its predictions. Online learning, which changes model parameters on a per-example

basis, is one way to incremental learning[8]. This is ideal for issues in which data enters continually, such as Malware detection. To apply online learning for the detection of multivariant malware, we must develop a model architecture that can handle various input data and predict the existence of several malware families[25]. One way is a neural network with several output nodes, where each node corresponds to a distinct malware family.

Suppose we have a dataset containing features X and labels Y, where each row represents a file and each column is a feature. We can minimize the loss function by training a neural network with L output nodes using stochastic gradient descent (SGD), as shown in Eq.1.

$$L(X,Y) = N_{i=1}L_{j=1}(X_i, Y_{ij}) \tag{1}$$

where N is the number of files in the dataset, L is the number of malware families, and $L_j(X_i, Y_{ij}$ is the loss function for the jth malware family, given the input features X_i and the corresponding label Y_{ij}. A common loss function for binary classification problems is the logistic loss function as given in Eq. 2.

$$L_j(X_i, Y_{ij}) = log(1 + exp(-y_{ij} * f_j(X_i))) \tag{2}$$

Where, $f_j(X_i)$is the output of the jth output node for the input X_i and model parameters.

The gradient of the loss function requires to be computed related to the parameters with an objective to model parameters using SGD. It is possible to determine the gradient for the logistic loss function using the chain rule expressed in Eq. 3.

$$L_j(X_i, Y_{ij}) = -Y_{ij} * (1 - (Y_{ij} * f_j(X_i))) * f_j(X_i) \tag{3}$$

where (z) is the sigmoid function, which maps any input to the range [0, 1] as given in Eq. 4

$$(Z) = 1/(1 + exp(-z)) \tag{4}$$

We can make sure that the model responds to changes in the data distribution over time and offers precise forecasts for each malware family by adjusting the weights in this manner.

Consider that there are three datasets, each with a unique distribution of files and virus families. For each dataset, we may train a different neural network, and then we can combine the predictions from each network using a weighted average to arrive at the final prediction. We may adjust the settings for each network independently for each new file X_i before updating the weights to arrive at a final conclusion, as a result, the model can adjust to variations in the distribution of the data and enhance its predictions over time.

1.2 Proposed Approach

In general, incremental learning has the potential to be an effective approach for identifying multivariant malware across many datasets. We can make certain that the model adjusts to changes in the data distribution and offers precise forecasts for each malware family by utilizing online learning to update the model parameters as new data becomes available.

The proposed algorithm for incremental learning for detecting multivariant malware on multiple datasets:

- Initialize K neural networks, each with its own set of model parameters and output nodes corresponding to the malware families we want to detect.
- For each dataset D_k, train the corresponding neural network on the files in $D_k$using standard supervised learning techniques.
- Initialize the weights $W_k$for each network to be equal, so that each network contributes equally to the final prediction.
- For each new file X_i, do the following:

a. For each network k, update the model parameters iusing online learning as given in Eq. 5.

$$kt + 1 = kt * L_j(X_i, Y_{ij}\mathrm{k}) \tag{5}$$

b. Update the weights $W_k$for each network using an incremental version of the softmax function as given in Eq. 6.

$$W_{kt+1} = softmax((w_{k1} * f_j(X_i, k+1), \ldots, W_{kK} * f_{jK}(X_i, k+1)]) \tag{6}$$

c. Compute the final prediction by taking the weighted sum of the predictions from each network as given in Eq. 7.

$$Y_i = k_{k=1}w_K * f_j(, kt + 1) \tag{7}$$

- Repeat steps 4a-4c for each new file x_i
- If the model's performance on any datasets deteriorates significantly, retrain the corresponding neural network using the new data and update the model parameters and weights as necessary. We can use standard metrics such as accuracy, precision, recall, and F1-score to evaluate the performance.
- Periodically re-evaluate the weights and update them if necessary to account for changes in the data distribution over time.

Using this technique across various datasets, we can develop a reliable and flexible model for identifying multivariant malware. The secret to its success is the use of incremental softmax to adjust the weights to changes in the data distribution over time and online learning to update the model parameters.

2 Methodology

2.1 Dataset

In this study, we used 6 datasets 2 each collected from VirusShare[26], Kaggle[27], and the Malware Capture Facility Project[28]. A large variety of known and undiscovered malware families are represented in the often-updated malware sample collection known as VirusShare. The malware analysis datasets provided by Kaggle include both examples of malicious code and clean files that may be used to train incremental learning algorithms. To develop incremental learning algorithms for network traffic analysis, the Malware Capture Facility Project offers a sizable collection of malware samples that have been extracted from real-world traffic, including known and undiscovered malware samples. The total number of instances collected from these datasets was 158101, with 126376 as benign and 31725 as malicious instances.

2.2 Training the Model

The data was cleaned, missing values were handled, and the data was structured for the pre-processing process to train an incremental learning model on several datasets. The feature sets were comparable across all datasets by selecting the most crucial features and eliminating the redundant ones. The model's performance on each dataset was closely watched during the training process to prevent overfitting or underfitting, and the learning rate was modified as necessary. In order to prevent the model from overfitting the data and becoming too complex, additional techniques, including regularization, were used. In activities like malware detection, where the data is constantly changing, and new samples need to be added over time, these approaches were successful in training the incremental learning model on a variety of datasets.

2.3 Feature Extraction and Update

We effectively used feature extraction and upgraded the neural network as part of the incremental learning process. Feature extraction helps to reduce the dimensionality of the input data. The model performs better because to this method, which also prevents overfitting. The neural network was then modified using an incremental learning technique. The present network was trained on a small batch of new data at a time instead of entirely retraining the network on the whole dataset. The network weights were updated using online learning with each new sample of data in this technique. Then, the result for the next set of data was projected using the changed weights. The present network was updated using the new data, and the outcomes were compared to what was expected. The network's weights were changed to reduce the error caused by the disparity between the two. This process was repeated for each new batch of data, enabling the neural network to retain its past understanding while responding to the new input. Jobs that deal with constantly changing data, like virus detection and natural language processing, substantially benefit from this incremental learning approach.

2.4 Feature Selection

Principal Component Analysis (PCA), a standard machine learning method for dimensionality reduction[29], was used in our research investigation. We used incremental learning for malware detection to lower the number of features required for precise predictions. When fresh data is incrementally introduced to the model during learning, PCA assists in lowering the number of features needed to represent this new data accurately. Due to the possibility that many characteristics may include ones that were redundant or unnecessary and did not improve the model's accuracy, this proved to be very helpful in identifying malware. Finding the linear combinations of initial characteristics that effectively capture the greatest variation in the data is how PCA works. Principal components are these combinations of features that maintain the most critical information about the data. They

are a smaller collection of features. We trained the model on a subset of data utilizing all available features in order to employ PCA in incremental learning for malware detection. After that, we used PCA to cut down on the number of required features. We progressively added fresh data to the model using the condensed feature set. We discovered that PCA is a powerful method for lowering the number of features needed in incremental learning for malware identification. Over time, it may improve the efficacy and accuracy of the model.

2.5 Convolutional Neural Network

A convolutional neural network (CNN) is a form of neural network that is specially built for image-processing tasks[30]. It has a number of layers, including fully connected, pooling, and convolutional layers.

The convolution operation can be represented mathematically as Eq. 8:

$$h_{ij} = \sigma\left(\sum_{m=1}^{M}\sum_{n=1}^{N} x_i + m - 1\, j + n - 1\, \omega_{m,n}\right) \tag{8}$$

where x is the input image, w is the convolutional filter, sigma is the activation function, and h is the output feature map.

Following the convolution procedure, the resulting feature map is subjected to a pooling operation to shrink it and extract the most crucial features. A typical kind of pooling procedure called ”max pooling” chooses the highest value possible in a particular area of the feature map. This has the following mathematical representation as in Eq. 9.

$$y_{ij} = max(h_{2i-1,2j-1}, h_{2i-1,2j}, h_{2i,2j-1,}\ \ h_{2i,2j} \tag{9}$$

where h is the input feature map, and y is the output of the pooling operation.

The pooling procedure results are then passed through one or more fully connected layers that correspond to the input features to the output classes. The completely linked layers have the following mathematical representation as given in Eq . 10.

$$Z_j = \sigma\left(\sum_{m=1}^{M} \omega_{mij} y_i + b_j\right) \tag{10}$$

where y is the input vector, w is the weight matrix, b is the bias vector, sigma is the activation function, and z is the output of the fully connected layer.

Without retraining the network entirely, the CNN's weights may be updated with fresh data in an incremental learning approach. Techniques like transfer learning, fine-tuning, or online learning will be used. The fundamental concept is maintaining previously acquired information while updating the current weights depending on the new input.

2.6 Performance Evaluation

Accuracy, precision, recall, and F1-score are some of the evaluation metrics often employed in malware detection using deep learning. Accuracy is calculated as (TP + TN) / (TP + TN + FP + FN), where TP stands for the number of true positives, TN for the number of true negatives, FP for the number of false positives, and FN for the number of false negatives. Accuracy is the proportion of correctly classified samples over the total number of samples. Precision may be computed as TP / (TP + FP), which expresses the fraction of genuine positive detections overall positive detections. Recall may be computed as TP / (TP + FN), where TP is the percentage of genuine positive detections over the total number of positive samples. The F1-score, which balances the significance of properly recognizing malware samples (precision) and not missing malware samples (recall), is the harmonic mean of precision and recall (recall). As 2 * ((precision * recall) / (precision + recall)), the F1-score may be computed. These assessment metrics provide several viewpoints on the model's effectiveness in utilizing deep learning to identify malware, and it is crucial to take the application-specific needs into account when choosing the evaluation metrics.

3 Results and Discussion

The experiments were carried out utilizing a computer featuring an Intel(R) Core i7-7500 central processing unit clocked at 3.0 GHz and equipped with 8 GB of RAM. The complete setting was emulated by employing the Python programming language in combination with the PyTorch deep learning platform. The dataset used in this study was obtained from diverse sources. Using different malware datasets for malware detection is crucial in the context of incremental learning, which involves continuously updating a model as new data becomes available. Multiple datasets provide a diverse range of malware samples for training and testing, which is important for incrementally updating the model and improving its accuracy over time. Different datasets represent different real-world scenarios and threats, ensuring the model can handle various types of threats and attacks in different environments. Evaluating the model's generalization capability using different datasets is also important to ensure that

it can effectively detect different types of malware and scenarios. Additionally, using multiple datasets allows for the comparison of the performance of the model with other systems that may have been trained and tested on different datasets, revealing strengths and weaknesses and identifying areas for improvement in the incremental learning process. The dataset size details are given in Table 1.

To increase the model's performance and allow it to learn from new data while keeping important information gathered from previous data, we used incremental CNNs and several other approaches in our study. Transfer learning was one of the methods we used. For our work, we used a pre-trained model developed using a large dataset. This saved us from having to restart the training from scratch, which may be time- and resource-intensive. We were able to use the information and features discovered from the substantial dataset and apply them to our goal by employing a pre-trained model. We used a learning rate of 0.01, momentum of 0.9, weight decay of 0.0005, and a batch size of 32. These choices, based on best practices and experimentation, aimed to balance convergence, efficiency, and model robustness. The We chose PCA for its effectiveness in dimensionality reduction and its ability to capture variance in high-dimensional data, making it suitable for our dataset's characteristics compared to other methods.

As a consequence, the model became more precise and effective. We also used the fine-tuning method, which includes using the new data to update just the weights of the last few layers of the preexisting model. By using this strategy, the model is guaranteed to maintain its prior knowledge while simultaneously picking up new information. In this study, we improved the model using fresh data, enabling it to retain previously learnt characteristics while picking up new ones. When the new data is comparable to the old data and the model just requires small changes to its parameters to function correctly on the new data, fine-tuning is extremely helpful. In addition, we used the strategy of introducing fresh neurons into the current network. In order to avoid having a substantial influence on the network's overall performance, this method entails adding additional neurons to the network's final layer and initializing their weights to fair values. Only the weights of the newly added neurons are changed during training; the weights of the old neurons are locked. Using this method, the model is able to preserve its prior knowledge while learning from fresh input. In our study, we expanded the network's final layer with new neurons while training merely changed the weights of the existing neurons. The experimental results obtained are depicted in Table 2.

Table 1. Dataset size

Dataset #	Number of Instances	Benign	Malicious	Malware Categories
Dataset 1	22040	19522	2518	2
Dataset 2	33250	27500	5750	3
Dataset 3	50486	31954	12532	6
Dataset 4	18609	16504	2105	4
Dataset 5	28095	21575	6520	2
Dataset 6	11621	9321	2300	2
Total	158101	126376	31725	19

Table 2. Results Obtained

Dataset #	Accuracy %	Precision %	Recall %	F1 %
Dataset 1	97.34	0.94	0.982	99.23
Dataset 2	99.01	0.91	0.975	99.11
Dataset 3	98.23	0.90	0.988	97.61
Dataset 4	98.94	0.92	0.990	99.50
Dataset 5	97.01	0.94	0.987	98.99
Dataset 6	98.36	0.93	0.912	98.75
Total (Proposed Approach)	99.34	0.96	0.972	99.21

The results indicate that a proposed approach achieved an accuracy of 99.34%, a precision of 0.96, a recall of 0.972, and an F1-score of 99.21, which is higher than the accuracy achieved while using individual datasets.

The accuracy achieved by the proposed approach is very high, which indicates that the approach is very effective at detecting malware. However, it is important to note that accuracy alone is not always the best metric to evaluate a model's performance, especially in the case of imbalanced datasets where the number of samples in each class is significantly different.

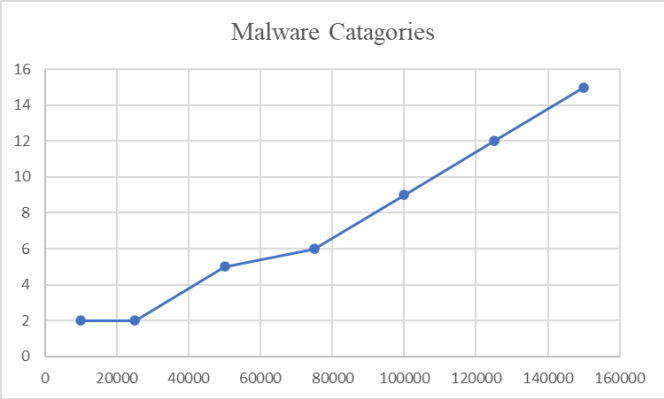

Fig 1. Malware Categories vs Instances Trained

The precision and recall achieved by the proposed approach are also high, which indicates that the approach is effective at correctly identifying malware samples while minimizing false positives. The F1-score, which is a harmonic mean of precision and recall, provides a good balance between the two metrics and is also very high, indicating that the proposed approach has good overall performance. The approach uniquely identified 15 categories of malware that were impossible to detect without using an incremental model on diverse datasets. The training curve with respect to the malware categories detected is shown in Figure 1. This indicates that the approach is able to generalize well to different types of malware and is not limited to specific types. The curve shows that as the number of malware categories detected increases, the performance of the approach improves. This indicates that the approach is able to learn and adapt to new types of malware as it is trained on more diverse datasets. Compared to results obtained in studies[4,5,17], the proposed approach performed better in terms of accuracy, presession and recall

4 Conclusion

Herein, this paper presents an advanced incremental learning approach to highly improve the accuracy and flexibility of multiclass malware variant identification and labeling. This is a new way of thinking about cybersecurity by preserving all previously learned data but introducing new threat vectors that nowadays change as we speak literally. The strength of our experiment was demonstrated by accurately locating 15 different malware categories that cannot be achieved with traditional non-interactive algorithms. It is evidently demonstrated by better performance measures like precision, recall, and F1-score. Our study rests on an innovative learning incremental architecture that is flexible and future proof to meet the escalating danger from cybercrime. Nevertheless, the study also admits some challenges mainly regarding the adaptability of the model in predicting and combating sophisticated and new malware strains, which may be improved upon in future versions. Subsequent research should involve applying current state-of-the-art deep learning algorithms to improve the model's prediction power. Additionally, further study should be undertaken to validate the theories of this work in actual field situations. This research adds to the existing body of knowledge in cybersecurity and creates room for more innovations. It emphasises on the need of continuously enhancing antimalware system as well prepares a platform towards more complex and stable cyber-safety measures.

References

1) Patil S, Vanmali A, Bansode R. Cyber Security Concerns for IoB. 2023. Available from: https://www.taylorfrancis.com/chapters/edit/10.1201/9781003305170-9/cyber-security-concerns-iob-sainath-patil-ashish-vanmali-rajesh-bansode.
2) Caramancion KM, Li Y, Dubois E, Jung ES. The Missing Case of Disinformation from the Cybersecurity Risk Continuum: A Comparative Assessment of Disinformation with Other Cyber Threats. *Data*. 2022;7(4):49–49. Available from: https://www.mdpi.com/2306-5729/7/4/49?trk=public_post_share-update_update-text.
3) Gibert D, Mateu C, Planes J. The rise of machine learning for detection and classification of malware: Research developments, trends and challenges. *Journal of Network and Computer Applications*. 2020;153:102526–102526. Available from: https://www.sciencedirect.com/science/article/pii/S1084804519303868.
4) Khan AS, Javed Y, Saqib RM, Ahmad Z, Abdullah J, Zen K, et al. Lightweight Multifactor Authentication Scheme for NextGen Cellular Networks. *IEEE Access*. 2022;10:31273–31288. Available from: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9734039.
5) Khan NA, Khan AS, Kar HA, Ahmad Z, Tarmizi S, Julaihi AA. Employing Public Key Infrastructure to Encapsulate Messages During Transport Layer Security Handshake Procedure. *2022 Applied Informatics International Conference (AiIC)*. 2022;p. 126–130. Available from: https://ieeexplore.ieee.org/

abstract/document/9914605.
6) Darem AA, Ghaleb FA, Al-Hashmi AA, Abawajy JH, Alanazi SM, Al-Rezami AY. An Adaptive Behavioral-Based Incremental Batch Learning Malware Variants Detection Model Using Concept Drift Detection and Sequential Deep Learning. *IEEE Access*. 2021;9:97180–97196. Available from: https://ieeexplore.ieee.org/abstract/document/9467300.
7) Alarfaj FK, Khan NA. Enhancing the Performance of SQL Injection Attack Detection through Probabilistic Neural Networks. *Applied Sciences*. 2023;13(7):4365–4365. Available from: https://www.mdpi.com/2076-3417/13/7/4365.
8) Habeeb RA, Nasaruddin F, Gani A, Hashem IA, Ahmed E, Imran M. Real-time big data processing for anomaly detection: A survey. *International Journal of Information Management*. 2019;45:289–307. Available from: https://www.sciencedirect.com/science/article/abs/pii/S0268401218301658.
9) Cho W, Lee H, Han S, Hwang Y, Cho SJ. Sustainability of Machine Learning-based Android Malware Detection Using API calls and Permissions. *2022 IEEE Fifth International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*. 2022;p. 18–25. Available from: https://ieeexplore.ieee.org/abstract/document/9939136.
10) Lee H, Cho SJJ, Han H, Cho WJ, Suh K. Enhancing Sustainability in Machine Learning-based Android Malware Detection using API calls. *2022 IEEE Fifth International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*. 2022;p. 131–134. Available from: https://ieeexplore.ieee.org/abstract/document/9939276.
11) Rahouti M, Ayyash M, Jagatheesaperumal SK, Oliveira D. Incremental Learning Implementations and Vision for Cyber Risk Detection in IoT. *IEEE Internet of Things Magazine*. 2021;4(3):114–119. Available from: https://ieeexplore.ieee.org/abstract/document/9548988.
12) Stocco A, Tonella P. Towards Anomaly Detectors that Learn Continuously. *2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. 2020;p. 201–208. Available from: https://ieeexplore.ieee.org/abstract/document/9307667.
13) Gu X, Zhao Y, Yang G, Li L. An Imbalance Modified Convolutional Neural Network With Incremental Learning for Chemical Fault Diagnosis. *IEEE Transactions on Industrial Informatics*. 2022;18(6):3630–3639. Available from: https://ieeexplore.ieee.org/abstract/document/9540239.
14) Kadam S, Vaidya V. Review and Analysis of Zero, One and Few Shot Learning Approaches. In: Advances in Intelligent Systems and Computing;vol. 1. Springer International Publishing. 2020;p. 100–112. Available from: https://link.springer.com/chapter/10.1007/978-3-030-16657-1_10.
15) Li J, Xue D, Wu W, Wang J. Incremental learning for malware classification in small datasets. 2020. Available from: https://www.hindawi.com/journals/scn/2020/6309243/.
16) Tayyab UEHE, Khan FB, Durad MH, Khan AB, Lee YS. A Survey of the Recent Trends in Deep Learning Based Malware Detection. *Journal of Cybersecurity and Privacy*. 2022;2(4):800–829. Available from: https://www.mdpi.com/2624-800X/2/4/41.
17) Li S, Li Y, Wu X, Otaibi SA, Tian Z. Imbalanced Malware Family Classification Using Multimodal Fusion and Weight Self-Learning. *IEEE Transactions on Intelligent Transportation Systems*. 2023;24(7):7642–7652. Available from: https://ieeexplore.ieee.org/abstract/document/9913918.
18) Chen Y, Xiong J, Xu W, Zuo J. A novel online incremental and decremental learning algorithm based on variable support vector machine. *Cluster Computing*. 2019;22(S3):7435–7445. Available from: https://link.springer.com/article/10.1007/s10586-018-1772-4.
19) Aslan O, Ozkan-Okay M, Gupta D. Intelligent Behavior-Based Malware Detection System on Cloud Computing Environment. *IEEE Access*. 2021;9:83252–83271. Available from: https://ieeexplore.ieee.org/abstract/document/9448102.
20) Zhang L, Thing VLL, Cheng Y. A scalable and extensible framework for android malware detection and family attribution. *Computers & Security*. 2019;80:120–133. Available from: https://www.sciencedirect.com/science/article/abs/pii/S016740481830419X.
21) Baptista I, Shiaeles S, Kolokotronis N. A Novel Malware Detection System Based on Machine Learning and Binary Visualization. *2019 IEEE International Conference on Communications Workshops (ICC Workshops)*. 2019;p. 1–6. Available from: https://ieeexplore.ieee.org/abstract/document/8757060.
22) Li J, Xue D, Wu W, Wang J. Incremental learning for malware classification in small datasets. 2020. Available from: https://www.hindawi.com/journals/scn/2020/6309243/.
23) Hei Y, Yang R, Peng H, Wang L, Xu X, Liu J, et al. Hawk: Rapid Android Malware Detection Through Heterogeneous Graph Attention Networks. *IEEE Transactions on Neural Networks and Learning Systems*. 2021;p. 1–15. Available from: https://ieeexplore.ieee.org/abstract/document/9524453.
24) Rahman MS, Coull S, Wright M. On the Limitations of Continual Learning for Malware Classification. *In Conference on Lifelong Learning Agents*. 2022;p. 564–582. Available from: https://proceedings.mlr.press/v199/rahman22a.html.
25) Hsieh RJJ, Chou J, Ho CHH. Unsupervised Online Anomaly Detection on Multivariate Sensing Time Series Data for Smart Manufacturing. *2019 IEEE 12th Conference on Service-Oriented Computing and Applications (SOCA)*. 2019;p. 90–97. Available from: https://ieeexplore.ieee.org/abstract/document/8953015.
26) Virusshare. . Available from: https://virusshare.com/.
27) Kaggle. . Available from: https://www.kaggle.com/.
28) Malware Capture Facility Project. *https://mcfpweeblycom/*.
29) Hasan BM, Abdulazeez AM. A review of principal component analysis algorithm for dimensionality reduction. 2021. Available from: https://publisher.uthm.edu.my/ojs/index.php/jscdm/article/view/8032.
30) Heinrich K, Zschech P, Janiesch C, Bonin M. Process data properties matter: Introducing gated convolutional neural networks (GCNN) and key-value-predict attention networks (KVP) for next event prediction with deep learning. *Decision Support Systems*. 2021;143:113494–113494. Available from: https://www.sciencedirect.com/science/article/pii/S016792362100004X.