Bulletin of Electrical Engineering and Informatics
Vol. 13, No. 3, June 2024, pp. 2000~2008
ISSN: 2302-9285, DOI: 10.11591/eei.v13i3.6802

Kernel rootkit detection multi class on deep learning techniques

Suresh Kumar Srinivasan, SudalaiMuthu Thalavaipillai
Department of Computer Science and Engineering, Hindustan Institute of Technology and Science, Chennai, India

Article Info

Article history:

Received May 25, 2023
Revised Sep 17, 2023
Accepted Sep 28, 2023

Keywords:

Artificial intelligence
Cloud instance
Data set
Deep learning
Google Cloud Platform
Rootkit

ABSTRACT

The harmful code application known as a rootkit is designed to be loaded and run directly from the operating system's (OSs') Kernel. Rootkits deployed in the Kernel, called Kernel-mode rootkits, can alter the OS. The intention behind these Kernel changes is to conceal the hack. Detecting a Kernel rootkit in a target machine is found to be quite challenging. Numerous techniques can be employed to modify the Kernel of a system. Kernel rootkits also create hidden access for attacks, enabling unauthorized entry to be gained by attackers on the machine. The ultimate consequence is that essential computer data can be modified, personal information can be gathered, and hackers can observe behavior. Synthetic neural networks support artificial intelligence, a branch of deep learning that models the human brain and operates on large datasets. This study proposed the Kernel rootkit detection multi-class deep learning techniques (KRDMCDLT). Deep learning algorithms are utilized to recognize the Kernel rootkit from a batch of data by selecting essential properties for learning tracking models. Thus, by identifying the OS malware, trojan assaults can be stopped before they can access infected data. This Kernel rootkit detection was tested in a Google Cloud Platform (GCP) computing system.

Corresponding Author:

Suresh Kumar Srinivasan
Department of Computer Science and Engineering, Hindustan Institute of Technology and Science
1 Rajiv Gandhi Salai (OMR) Padur, Chennai-603103, Tamil Nadu, India
Email: sureshkumarphd2018@gmail.com

1. INTRODUCTION

The complicated adaptable and scalable internet-based computing system allows computer hardware and software, data storage, and computing speed to be made portable whenever needed, as well as an alternative plan for the organization. Cloud technology can provide a wide range of services more easily online. A rising number of organisations are switching from domestic to public cloud providers, which suggests that sustaining the integrity and scalability of web services is becoming more difficult [1]. The importance of the challenges is emphasized by many valuable features to both customers and organizations, including reduced expenses, improved productivity, rapidity, reliability, efficiency, and security. The application that is intended to harm and destroy systems and devices is known as intrusive malware, or malware [2]. Some common types of malwares include malicious software, spyware, viruses infections, and rootkits. Malicious software from the backdoor virus group can be used to gain unintentional entry to an operating system (OS) or communication network. Rootkits, which are difficult to detect and can be hidden on an individual's computer, enable hackers to remotely access the system, acquire administrative rights, and extract data [3]. Different methods, such as static rootkit deduction, dynamic rootkit deduction, signature rootkit deduction, heuristic-based rootkit deduction, and machine learning-based rootkit deduction, are used to detect rootkits. However, all these methods have limitations. They cannot detect mutation rootkits, and the accuracy is very low even when a deduction is made. The proposed solution is the Kernel rootkit detection

multi class deep learning techniques (KRDMCDLT) to detect the Kernel rootkit. This model incorporates four deep learning algorithms: multilayer perceptron (MLP) [4], radial basis function networks (RBFN), restricted Boltzmann machine (RBM), and convolutional neural network (CNN) which can detect the rootkit in parallel. These four algorithms are embedded in the KRDMCDLT model.

Individuals can frequently exceed the precision of deep learning [5] networking technologies. Neural network models are trained with CNN, MLP, RBM, and RBFN in conjunction with extensive tagged data collection. The cloud implementation of this system depends on machine learning methods to scan for malicious programmes. The cloud environment being used is Google Cloud Platform (GCP), the biggest and most well-known internet technology used for implementation globally. GCP is designed to facilitate the efficient and secure hosting of apps for vendors and app developers, irrespective of whether they are SaaS. The rootkit malware family of malicious software can be utilized to gain unauthorized access to a communication or operating system (OS). The benefits of GCP, including accessibility, security, adaptability, and reliability are appreciated.

A rootkit is a form of harmful software designed to enter a computer system without authorization and avoid being discovered by the security measures in place. Deep learning models, data mining, and machine learning-based rootkit detection are some of the techniques used for rootkit detection. Other techniques include static rootkit detection, dynamic malware identification, based upon signatures rootkit detection, heuristic-based rootkit detection, and backdoor detection based on rootkits. In the majority of rootkit investigation studies, artificial intelligence, machine learning, and data analysis methods are frequently used.

2. METHOD

The file is downloaded and transferred to the GCP cloud on a client machine. The KRDMCDLT are available within the cloud. An approach for detecting Kernel rootkits is proposed in this study, which involves acquiring four different datasets and individually analyzing each of them using various deep learning algorithms. Self-organizing maps, neural networks with convolution, deep relief networks, limited Boltzmann machines, and axial basis function networks are a few of the artificial intelligence techniques used in the current study. The malicious URL dataset (dataset 4), classification of malware dataset (dataset 3), dynamic APICall sequence dataset (dataset 2), and malware dataset (dataset 1) were collected from the www.kaggle.com.

2.1. Sustainability structure for tests

The framework can be tested in the GCP web services [6] cloud environment by setting up and operating a virtual machine. GCP notebook instances can be employed for building, preparing, and drawing conclusions from the model. Manual download options and file system storage are available for GCP Sagemaker Notebook instances. Additionally, hosting a GCP Notebook application on the sample is allowed by the Jupyter experimental setup, facilitating the training and assessment of the deep learning model's outcomes. The design of the model, which aids in the identification of rootkits and benign entities, is presented in Figure 1.

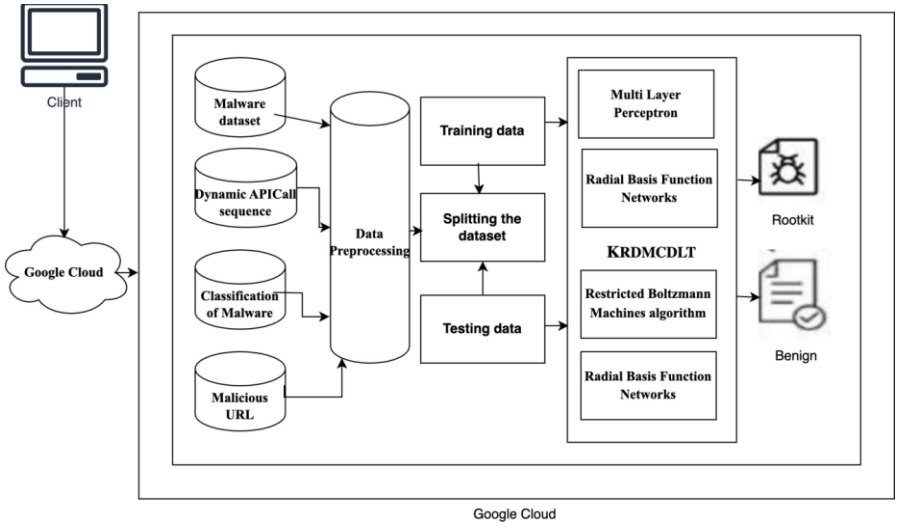

Figure 1. Architecture diagram of Kernel rootkit detection multi class on deep learning techniques

2.2. Data preprocessing

The four datasets—rapid application programminng interface (two), group of rootkits (three), malware [7] dataset analysis (one), malicious URL dataset (four), and—that have been gathered are used for comparison. There are roughly 50,000 clean file and 50,000 malicious files in dataset 1. The categorization feature maps the malware [8] as 1, whereas the safe file is mapped as 0. The dataset includes a number of characteristics, including millisecond, state, staticprio, previous, and policy, among others. "maj felt," "shared vm," truncate count," "now,""exec vm," and "hash" have been eliminated because their characteristic values are zero. This model's chosen target data is "classification." Dataset 2 includes 1,090 good application programming interface (API) call sequences and about 41,897 malicious API call sequences, including examples like t1, t2, t3, t4, t5, and so on. The 'hash' characteristic has been eliminated from the data pertaining to features for the 102 examples in dataset 2. The data collection chosen in this model is 'malware.'

There are two different comma separated value (CSV) file types in dataset 3 [9]. The malicious clean files (5184) are in a sample collection known as the securely attached integrated-5184 dataset. It has 69 characteristics, of which 15 are derived characteristics and 54 are raw. 'Base of data,' 'Image base,' 'Section alignment,' 'File alignment,' and 'Base of code,' are some of its attributes. The element that needs data's identify 'class' has been eliminated from the dataset's 69 instances. The clamp Raw-5184 testing dataset has 5184 clean and malicious code samples. There are 55 unprocessed aspects in it, such as "machine," "number sections," "creation year," "pointer to symbol table," and "a number of symbols,". The term "class" serves as the model's chosen data source.

Dataset 4 [10] is composed of 651 instances, incorporating a total of 191 web addresses. Among these addresses, 428,103 are categorized as secure or safe, 96,457 as defacement web addresses, 94,111 as faked email addresses, and 32,520 as dangerous. "Uniform Resources Locator" and "type" are this dataset's two main attributes. 'Uniform Resources Locator,' 'domain,' 'category,' and 'type' have been omitted from the dataset. The model's data source for evaluating harm is the word "category".

2.3. Splitting the dataset

In machine learning [11], splitting a dataset is a typical practice to assess a model's performance and avoid overfitting. The dataset typically consists of three main components: training, validation, and test sets. This study's data collection is divided into a data set to train and a test set of data. One of the most important steps in data pre-processing is dividing the collected data into training dataset and testing datasets. The models' dependability and efficiency might be improved by this method. The largest component of the first dataset, known as the training data, has been used to train or fit the deep learning model, with a setting of 80%. The model has been evaluated using test data with a 20% threshold.

2.4. Artificial intelligence algorithm

2.4.1. Multi layer perceptron

In machine learning [12], the practice of splitting a dataset to assess a model's performance and prevent overfitting is commonly employed. Typically, a dataset comprises three primary components: training, validation, and test sets. A critical step in data pre-processing, the data collection for this study was split into a dataset used for training and a testing dataset. Utilizing this method may lead to improved reliability and efficiency of the models. The more significant portion of the first dataset, the training data, was used for training or fitting the deep learning model, accounting for 80%. The model's evaluation was performed using test data, with a threshold of 20%.

2.4.2. Radial basis function networks

A radial basis function (RBF) network [13] is employed, which uses a special neural network with a feed-forward algorithm to address problems involving function averaging. RBF networks are characterized by having a multi-layer structure, global computation, and a quicker learning process when compared to other artificial intelligence models. Circular basis functions have three layers: the input data, concealing components, and result. Received and communicated to the external nodes, where the calculation is carried out, is information compiled from the data being entered nodes. For activities requiring prediction, like as analysis or grouping, the outcome section is used. The overall outcome for each neuron is calculated using a set of characteristics that can be taught to specify the distance between each neuron's input and the RBF layer's centre point. It is necessary to define irregular correlations between the characteristics of the input variable and the chosen variable in order to accurately anticipate the output. The scale vector computing class from the the support vector machine module of sci-kit-learn is initially initialised using the parameters gamma='scale' and kernel='rbf' in order to use the RBM network analyzer. The RBM network is then built on the basis of the original samples using the selection methodology, and the groups of identities are predicted

on the basis of the testing data using the model-generation technique. Finally, using the grade activity for positive effects, the performance of the RBM network on the assessment data is assessed.

2.4.3. Restricted Boltzmann machine algorithm

RBMs [14], a random computational neural network, are a subset of the more prominent family of Boltzmann machines. The viewed layer and the invisible layer are the two layers that make up an RBM. Weighted connections connect these two layers, but due to the "restricted" nature of these connections, there is no interaction within either layer. The hidden layer is created by removing feature data from the dataset, creating an obscured layer. The secret layer cell computes a weighted sum of the input data from the visible layer. It adds the outcome after processing the visual layer of data neurons enter with an invalid value. Repeat the procedure until the final results are identical to the initial data. The capacity of RBMs to develop meaningful models for information provided, which may be used for tasks like feature extraction, clustering, and generation, is one of their main advantages. RBMs [15] are self-supervised models that can be trained from the data being entered without the need for data tags. The sklearn—neural_network module's Bernoulli RBM is used to start an RBM model and include RBM into it. After the training and test data set have been preprocessed using the transformation process, the RBM models are fitted to the sample set. Finally, a classification algorithm trained on the updated information is applied to the test data to make predictions. The model's performance is subsequently assessed.

2.4.4. Convolution neural networks

CNN are a specific type of deep neural network commonly utilized for visual data processing, such as image recognition. Within deep learning, various neural network architectures exist, including fully connected, pooling, and convolutional layers. As the network's depth increases, the receptive field also expands. Small Kernel matrices, which have been trained to recognize features in the input data, constitute the quickly learned variables. The integrated layer replaces the network output at specific points, reducing representation density and the number of calculations and weights required. Calculating the fully connected layer is straightforward because every cell in that layer was connected to every other neuron in the preceding and subsequent layers, using techniques such as offset factors and standard matrix multiplication. The fully attached layer creates a symbolic mapping from the data entering and the results. A CNN [16] extracts pertinent features from incoming visual data and executes complex image-processing tasks through its hidden layers. To efficiently carry out these tasks, the artificial neural network must possess a secret component that learns structured models of the incoming visual data. The level of abstraction gradually increases with the addition of more layers.

The most important advantage of CNNs is their ability to automatically detect crucial traits without human assistance. The CNN [17] model's architecture must first be defined. There is just one deep level, one optimum pooling level, and one convolutional level. The model is constructed using the Adam method, a digital efficiency function for loss calculation, and a measure of accuracy. The model is built using the training dataset and repeated ten more times with a batch size limitation of 32. Predictions are generated using the model's prediction approach and are approximated to the nearest number for further evaluation with experimental data. Efficiency is then calculated by determining the average similarity between the expected and actual labels, segmented by elements.

3. RESULTS AND DISCUSSION

The mentioned four algorithms are used to turn the data into models [18]. Subsequently, the dataset is divided. The file is downloaded and uploaded to Google Cloud for comparison with the model. Following the comparison with the model, the task of detecting whether the file is a rootkit or benign is assigned to the model. The four algorithms and four datasets are executed sequentially. Each algorithm and dataset are tested with sixty-four sample rootkits. The rootkit samples, as mentioned in Table 1, were collected from virustotal.com. Table 2 displays the confusion matrix for multi-class Kernel rootkit detection using deep learning techniques.

These rootkits were downloaded onto the client machine. The specific file was then transferred to Google Cloud for verification using the KRDMCDLT to determine whether it was a rootkit or benign [19]. When this verification was conducted on the optimal launch system in windows 2022 against 64 rootkits [20], every test attempting to differentiate the cloud was unsuccessful. The rootkit [21], which targeted a brand-new boot system, was included in the tests. This system yielded no false positives, achieving a 100 percent rate of certified negatives and zero false positives. The ideal windows instance system is expected to remain flawless. Of the 64 rootkits tested, 42 were included in the validation set, and the remaining 16 out of the 58 rootkits needed to be distinguished, leaving only 6. The findings showed a 23.63% number of false negatives and a 76.36% positive detection accuracy. An incorrect rootkit installation setup was responsible

for the false negatives. Nevertheless, attackers continue to refine their rootkit installation techniques, resulting in a lower false negative rate.

Table 1. Input field

S.no	Malware name	Malware size	Offensive
1	Virus. BAT.Qwerty. b	676 kb	ucrtbase.dl
2	Virus.Boot.Catman	28 kb	Scvhost.exe
3	Virus.Unix.Sillysh.b	6.84 kb	Aubot.exe
4	9ba7332fdca46ed72bd788def5498140	793 kb	User32.dll
5	38c7bd26550daa3b4527f4eeefe8a0dd	81.5 kb	Svchost.exe
6	D0617FEDF0EA31D7D5FB55BD334D85D6	8 kb	Svchost.exe
7	f0f927ee20a62d0b0a1b37d68d1406ea	78 b	Svchost.exe
8	$%&%_2169.vir@	22.86kb	Taskhost.exe
9	Backdoor.Win32.Haxdoor.gs	1.26 mb	Taskhost.exe
10	bakuryu	121 kb	Scvhost.exe
11	shell.jpg	89.1 kb	Svchost.exe
12	f6e671d8630df5d8045ff4243da94f74	24 kb	Ucrbase.dl
13	afe8df184dccf6db48cf27916d0d0da6	48 kb	ucrtbase.dll
14	6eddd98e0463acaa3aa0eeab26b1d3c9	1 kb	Ucrbase.dll
15	80da4801d2b70d7044e9d660a05c676	5.03 kb	Svchost.exe
16	4356aded80ee30d1f85321ecc28694b3	140 b	Taskhost.exe
17	e08de794d84c472b1fd9a862bd729556	107 b	System32.dll
18	Rootkit.Win32.Agent.agk	512 b	Ucrbase.dll
19	Rootkit.Win32.Agent.azt	512 b	Ucrbase.dll

Table 2. Confusion matrix of Kernel rootkit detection multi-class on deep learning techniques

Actual	Predicted(-)	Predicted(+)
-	6	0
+	16	42

An aspect of the aforementioned deep learning models that can be observed is their capability to handle vast datasets, manage complex and nonlinear relationships, effortlessly extract high-level features, and adapt to novel and evolving threats. These methods, regarded as the most effective, are employed for evaluating the rootkit detection models. Methodologies [22] were used to evaluate the model's efficiency on the GCP web service. The corresponding graphical representation provide a description of results evaluation. Malicious information visualisation research in Table 3 while the virus dataset evaluation procedure is shown in Figure 2.

Table 3. Malicious information visualization research

Algorithms	Accuracy (%)	Precision	Recall	F1 score
MLP	99.96	.9	.9	.9
RBFN	99.82	.9	.9	.9
RBM	59.30	.6	.62	.6
CNN	50.33	.5	1	0.67

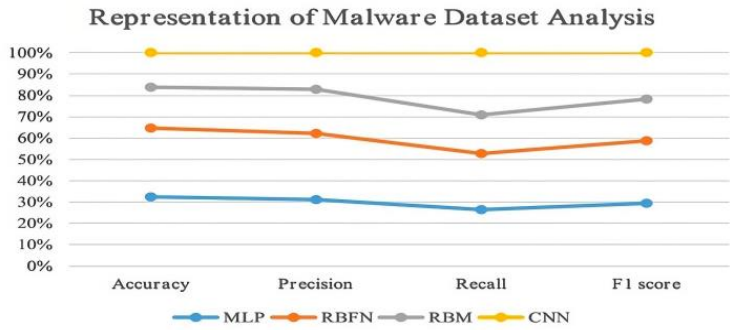

Figure 2. Malware dataset analysis

The most effective method is the multi-layer perceptron algorithm, as it can analyze vast amounts of initial data, address complex nonlinear challenges, and produce predictions on the malware dataset with an

accuracy of 99%. A summarized sequence of changing API calls illustratedis presented in Table 4. A graphical depiction of a sequence of changing API calls illustrated is displayed in Figure 3.

Table 4. Sequence of changing API calls illustrated

Algorithms	Accuracy (%)	Precision	Recall	F1 score
MLP	98	1	.98	0.99
RBFN	99	1	0.96	0.99
RBM	97	1	0.99	0.99
CNN	98	1	0.98	0.99

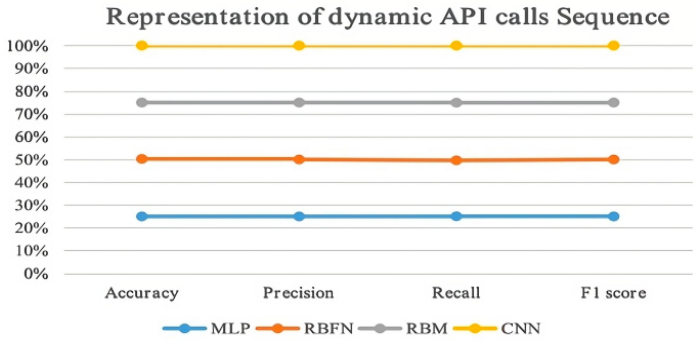

Figure 3. Sequence of changing API calls illustrated

The RBM in Figure 3 is considered the best method because it involves the input and storage of binary data, enhances standard by placing a current time collection of information and monitoring, and prioritizes developments and statistical abilities. It also achieves a high success rate in the representation of dynamic API call sequence. A representation of classification is provided in Table 5. The most effective method is the individual planning map, which, as depicted in Figure 4, can divide the data, present syntax clearly in two perspectives, and achieve a ninety percent accuracy in the representation of classification. A display of a dangerous URL is detailed in Table 6.

Table 5. Representation of classification

Algorithms	Accuracy (%)	Precision	Recall	F1 score
MLP	60	.65	.89	.82
RBFN	57	1	0.27	0.39
RBM	52	0.76	0.99	0.78
CNN	50	0.34	0.39	0.22

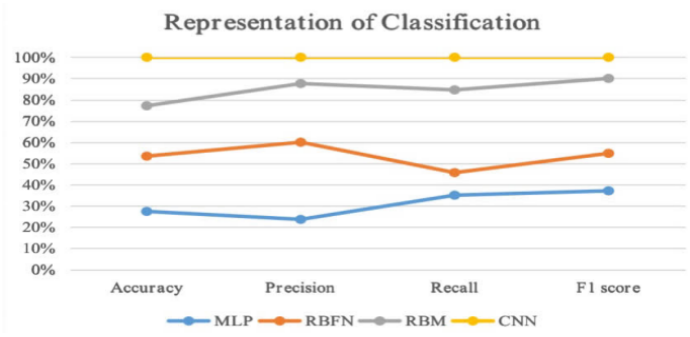

Figure 4. Representation of classification

Table 6. Display of a dangerous URL

Algorithms	Accuracy (%)	Precision	Recall	F1 score
MLP	83	0.71	0.96	0.77
RBFN	66	0.76	0.99	0.87
RBM	66	0.77	0.76	0.65
CNN	86	0.76	0.92	0.89

Less computing effort is required by the CNN, as illustrated in Figure 5, compared to other approaches. The CNN is the favored method for recognizing crucial elements without human supervision because an accuracy of eighty-five percent is achieved in the representation of malicious URLs. CNNs exhibit high precision in image detection and categorization. Furthermore, CNNs [23] provide the significant advantage of weight sharing, reducing computation compared to standard neural networks. Complex structures and characteristics in data can be identified by deep learning models, which is valuable for detecting hidden or covert malware operations. The degree to which a deep learning model can detect rootkits depends on the kind and volume of collected information the complexity of the rootkit, and the training phases of the model phase. Accuracy, recall, and F1 score are used to assess the approach efficiency in terms of classification and how well it can recognise samples that belong to a particular class. These criteria are crucial for assessing the effectiveness of a deep learning model and its suitability for practical applications. GCP [24] provides a comprehensive set of privacy rules and oversight to ensure the confidentiality, security, and accessibility of customer data in the public web. The concept of shared responsibility holds clients and cloud service providers (CSPs) like GCP jointly accountable for cloud security. There are responsibilities shared by the CSP and the customer with regard to the security of physical facilities, network surveillance, and server confidentiality in the internet. On the other side, the client is in responsibility of protecting their cloud-based OS, apps, and information. This involves setting allocated security and entry regulations. In support of these initiatives, GCP provides a secure cloud architecture.

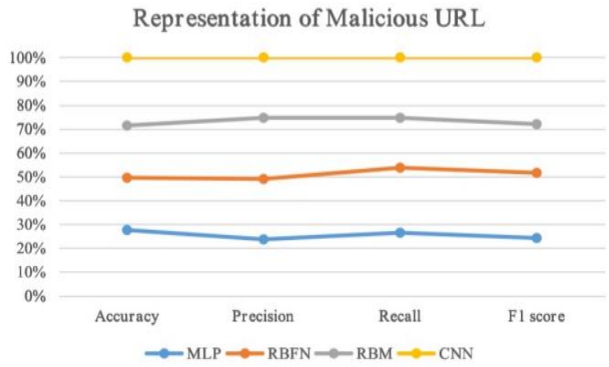

Figure 5. Representation of malicious URL

When the MLP algorithm was applied, an accuracy of 99.96% was achieved in representing the malware dataset. Conversely, when the same dataset was subjected to the RBFN [25] algorithm, an accuracy of 99.82% was obtained. An accuracy of 98.62% was achieved when the RBFN algorithm was utilized to represent the dynamic API dataset. Meanwhile, the application of the CNN, MLP, and RBM algorithms to the same dataset resulted in accuracies of 97.53%, 97.42%, and 97.42%, respectively. For the representation of malicious URLs using algorithms, accuracies of 82.82% and 85.62% were attained by the MLP and CNN, respectively. In summary, as indicated in Table 7, the representation of the malware dataset using the MLP and RBFN algorithms yields notably high accuracy levels.

Table 7. Compare the dataset, algorithms, and accuracy

Dataset	Algorithms	Accuracy (%)
Representation of malware dataset	MLP	99.96
Representation of malware dataset	RBFN	99.82
Representation of dynamic API	RBFN	98.62
Representation of dynamic API	CNN	97.53
Representation of dynamic API	MLP	97.42
Representation of dynamic API	RBM	97.42
Representation of malicious URL	MLP	82.82
Representation of malicious URL	CNN	85.62

4. CONCLUSION

Rootkit removal from infected machines and rootkit prevention depend heavily on the effectiveness of malware identification. Since each type of rootkit operates uniquely and presents specific hazards,

detecting rootkits is essential for determining the most effective measures to defend against and prevent rootkit infections. In this study, the efficacy of diverse datasets is evaluated KRDMCDLT. In comparison to several traditional methodologies, artificial intelligence models perform well at detecting rootkits in a GCP cloud service. The four algorithms are transformed into models through data preprocessing and dataset splitting. The files are downloaded, uploaded to Google Cloud, and compared with the models. After the comparison, the models determine whether the files are rootkits or benign. A total of sixty-four sample rootkits are implemented in the GCP cloud environment and tested with various datasets. High precision is achieved through hyperparameter tuning, which involves modifying parameters such as batch size, number of cycles, and number of layers to optimize the models. The four datasets are analyzed using a variety of algorithms, including CNN, MLP, RBM, and RBFN. MLP and RBFN are recognized as excellent technologies for achieving high accuracy. Future enhancement work may include implementing other techniques using various algorithms on the same datasets.

ACKNOWLEDGEMENTS

The Department of Computer Science and Engineering at the Hindustan Institute of Technology and Science in Chennai, India, provided the infrastructure needed to create the models for the study issue. The writers would like to thank you for your assistance.

REFERENCES

[1] D. Tian, Q. Ying, X. Jia, R. Ma, C. Hu, and W. Liu, "MDCHD: A novel malware detection method in thecloudusinghardware trace and deep learning,"*Computer Networks*, vol. 198, 2021, doi: 10.1016/j.comnet.2021.108394.

[2] L. F. Ilca, O. P. Lucian, and T. C. Balan, "Enhancing cyber-resilience for small and medium-sized organizations with prescriptivemalware analysis, detection and response," *Sensors*, vol. 23, no. 15, p. 6757, Jul. 2023, doi: 10.3390/s23156757.

[3] A. Tsohou, V. Diamantopoulou, S. Gritzalis, and C. Lambrinoudakis, "Cyber insurance: state of the art, trends, and future directions," *International Journal of Information Security*, vol. 22, no. 3, pp. 737–748, 2023, doi: 10.1007/s10207-023-00660-8.

[4] M. Shobana and S. Poonkuzhali, "An efficient botnet detection approach for green iot devices using machine learning techniques," *Journal of Green Engineering*, vol. 10, no. 3, pp. 1053–1076, 2020.

[5] K. R. Sowmia and S. Poonkuzhali, "Artificial intelligence in the field of education: a systematic study of artificial intelligence impact on safe teaching learning process with digital technology," *Journal of Green Engineering*, vol. 10, no. 4, pp. 1566–1583, 2020.

[6] R. Vadivel and T. Sudalaimuthu, "Cauchy particle swarm optimization (CPSO) based migrations of tasks in a virtual machine," *Wireless Personal Communications*, vol. 127, no. 3, pp. 2229–2246, 2022, doi: 10.1007/s11277-021-08784-7.

[7] J. Jeyalakshmi and S. Poonkuzhali, "Prescriptive analytics of constraint optimisation of diabetes diet exhortation by using information systems," *Journal of Environmental Protection and Ecology*, vol. 22, no. 6, pp. 2672–2681, 2021.

[8] I. Ahmed, "Technology organization environment framework in cloud computing," *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 18, no. 2, pp. 716–725, 2020, doi: 10.12928/TELKOMNIKA.v18i2.13871.

[9] S. Poonkuzhali and J. Jeyalakshmi, "Study of diabetes mellitus patients for thyroid related co-morbidities using data analytics," *Basic and Clinical Pharmacology and Toxicology*, vol. 124, no. S3, pp. 19–20, 2019.

[10] C.-L. Chen and S. Punya, "An enhanced WPA2/PSK for preventing authentication cracking," *International Journal of Informatics and Communication Technology (IJ-ICT)*, vol. 10, no. 2, pp. 85–92, Aug. 2021, doi: 10.11591/ijict.v10i2.pp85-92.

[11] A. Vijayaraj, R. M. Suresh, and S. Poonkuzhali, "Node discovery with development of routing tree in wireless networks," *Cluster Computing*, vol. 22, pp. 10861–10871, 2019, doi: 10.1007/s10586-017-1211-y.

[12] A. Vijayaraj, R. M. Suresh, and S. Poonkuzhali, "Load balancing in wireless networks using reputation-ReDS in the magnified distributed hash table," *Multimedia Tools and Applications*, vol. 77, no. 8, pp. 10347–10364, 2018, doi: 10.1007/s11042-018-5620- 6.

[13] P. Sugumaran, K. K. Ravi, and T. Shanmugam, "A novel algorithm for enhancing search results by detecting dissimilar patterns based on correlation method," *International Arab Journal of Information Technology*, vol. 14, no. 1, pp. 60–69, 2017.

[14] O. E. Taylor, P. S. Ezekiel, D. J. S. Sako, "A Deep Learning Based Approach for Malware Detection and Classification," *iJournals: International Journal of Software & Hardware Research in Engineering (IJSHRE)*, vol. 9, no 4, pp. 32-40, Apr. 2021.

[15] Q. Wang and Q. Qian, "Malicious code classification based on opcode sequences and textCNN network," *Journal of Information Security and Applications*, vol. 67, 2022, doi: 10.1016/j.jisa.2022.103151.

[16] N. Aman, Y. Saleem, F. H. Abbasi, and F. Shahzad, "A hybrid approach for malware family classification," *Communications in Computer and Information Science*, vol. 719, pp. 169–180, 2017, doi: 10.1007/978-981-10-5421-1_14.

[17] B. Liu *et al*., "An approach based on the improved SVM algorithm for identifying malware in network traffic," *Security and Communication Networks*, vol. 2021, pp. 1–14, Apr. 2021, doi: 10.1155/2021/5518909.

[18] B. Liu, J. Chen, S. Qin, Z. Zhang, Y. Liu, and L. Zhao, "An Approach Based on the Improved SVM Algorithm for Identifying Malwarein Network Traffic," *Security and Communication Networks*, vol. 1, pp. 1-14, 2021, doi: 10.1155/2021/5518909.

[19] B. U. A. Barathi and S. Poonkuzhali, "Design and implementation of interactive data analytics model for predicting the survivability of breast cancer patients," *Journal of Environmental Protection and Ecology*, vol. 21, no. 4, pp. 1455–1468, 2020.

[20] A. Ullah, I. Laassar, C. B. Şahin, O. B. Dinle, and H. Aznaoui, "Cloud and internet-of-things secure integration along with security concerns," *International Journal of Informatics and Communication Technology (IJ-ICT)*, vol. 12, no. 1, pp. 62-71, Apr. 2023, doi: 10.11591/ijict.v12i1.pp62-71.

[21] M. Awais, Q. Abbas, S. Tariq, and S. H. Warraich, "Blockchain based secure energy marketplace scheme to motivate P2P microgrids," *International Journal of Informatics and Communication Technology (IJ-ICT)*, vol. 11, no. 3, pp. 177–184, Dec. 2022, doi: 10.11591/ijict.v11i3.pp177-184.

[22] X. Wang, J. Zhang, A. Zhang, and J. Ren, "TKRD: Trusted kernel rootkit detection for cybersecurity of VMs based on machine learning and memory forensic analysis," *Mathematical Biosciences and Engineering*, vol. 16, no. 4, pp. 2650–2667, 2019, doi: 10.3934/mbe.2019132.
[23] B. Yergaliyeva, Y. Seitkulov, D. Satybaldina, and R. Ospanov, "On some methods of storing data in the cloud for a given time," *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 20, no. 2, pp. 366–372, 2022, doi: 10.12928/TELKOMNIKA.v20i2.21887.
[24] H. S. Hamid, B. AlKindy, A. H. Abbas, and W. B. Al-Kendi, "An intelligent strabismus detection method based on convolution neural network," *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 20, no. 6, pp. 1288–1296, 2022, doi: 10.12928/TELKOMNIKA.v20i6.24232.
[25] S. M. Kareem and A. M. S. Rahma, "A new multi-level key block cypher based on the Blowfish algorithm," *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 18, no. 2, pp. 685–694, 2020, doi: 10.12928/TELKOMNIKA.V18I2.13556.

BIOGRAPHIES OF AUTHORS

Suresh Kumar Srinivasan is a doctoral candidate at the Department of Computer Science and Engineering at the Hindustan Institute of Technology and Science in Chennai, India. Suresh holds undergraduate degrees in computer science and engineering from the University of Madras in Chennai and master's degrees from Anna University in Chennai. His most recent studies concentrate on the security capabilities of cloud computing. He has been a lifelong member of CSI. He can be contacted at email: sureshkumarphd2018@gmail.com.

SudalaiMuthu Thalavaipillai works at the Hindustan Institute of Technology and Science in Chennai, India, where he is a part of the School of Computing Science. His doctorate was obtained at Chennai, India's Hindustan Institute of Technology and Science. He has been certified as an ethical hacker. 50 research papers that he wrote have been published in prestigious international journals and conferences, and he has been given both Indian and Australian patents. He has won various awards throughout his professional career, including the best entrepreneur award for patent rights and the Pearson Award for Outstanding Professor. Machine learning, grid and cloud computing, and cyber network security are some of his areas of interest. Lifetime members of the IEEE, CSI, and ACM include Sudalaimuthu Thalavaipillai. He can be contacted at email: sudalaimuthut@gmail.com.