International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10s
DOI: https://doi.org/10.17762/ijritcc.v11i10s.7629
***Article Received:** 02 June 2023 **Revised:** 27 July 2023 **Accepted:** 11 August 2023*

MalDet-Malware Detection Using Deep Learning and LSTM based Approach to Classify Malware

Gaurav Mehta[1], Prof. (Dr.) Shaily Jain[2], Dr. Prasenjit Das[3], Dr. Vikas Tripathi[4]
[1]Computer Science and Engineering
Chitkara University, Himachal Pradesh
Baddi, India
gaurav.mehta@chitkarauniversity.edu.in
[2]Computer Science and Engineering
Chitkara University, Himachal Pradesh
Baddi, India
shaily.jain@chitkarauniversity.edu.in
[3]Computer Science and Engineering
Chandigarh university
Chandigarh, India
Prasenjit.e14134@cumail.in
[4]Computer Science and Engineering
Graphic Era
DehraDoon, India
vikas.tripathi@gmail.com

Abstract— Computer security requires malware detection. Recent research manually uncovers hazardous features using machine learning-based techniques. MalDet, a cutting-edge malware detection method, is recommended in this paper. MalDet classifies malware using a stacking ensemble and learns from grayscale images and opcode sequences using CNN and LSTM networks. According to the evaluation, MalDet's malware detection validation accuracy is 99.89%. MalDet outperforms other previous research with 99.36% detection accuracy and a significant detection speedup on the Microsoft malware dataset. We classified nine malware families for MalDet.

I. INTRODUCTION

Malware detection is always a major topic of discussion in the computer security industry because many different types of software today offer users a plethora of resources but also pose some risks. A recent study found that the prevalence of fraudulent samples is rising. For example, in 2016 [1], 69,277,289 harmful items were found by Kaspersky Lab. McAfee Labs [2] reported 670 million malware samples in 2017, up 22% from the previous four quarters. Due to the overwhelming volume of instances, a highly efficient malware detection method is essential.

Numerous researchers have looked into different ways to analyse and spot malware. Most signature-based commercial antivirus programs require a local signature database with malware patterns obtained by professionals. Malware can easily elude detection by encrypting, obfuscating, or packaging itself. This method is limited because little virus alterations can affect the signature. Thus, static analysis [3, 4] and dynamic behavior analysis [5, 6] have been proposed as machine learning methods for malware detection. It is challenging to arouse all malware behaviours by dynamic analysis because of the necessity to recreate the operation environment for malware, even though dynamic analysis does not involve extensive reverse engineering. Some malicious behaviours hide long before they attack, making malware behaviour monitoring a time-consuming process. Static analysis's speed in detecting even the most widespread malware is one of its greatest strengths. However, various forms of encryption and obfuscation pose the most significant challenge for static analysis. Because attackers can intentionally modify malware, static analysis struggles to capture the malware's traits accurately. Malware exploits packaging technologies to thwart reverse engineering, driving up the price of static analysis.

Several machine-learning algorithms [5-7] have recently been applied to malware detection to overcome the challenges raised above. Many, however, rely significantly on domain knowledge to examine malware and extract fictitious traits. A classification machine learning model is then trained using these attributes to determine how to classify an unseen file sample. Malware, however, is a severe issue since it is continually being developed, updated, and altered.

Our primary goals are to increase malware detection accuracy and efficiency while decreasing the time and money spent on artificial feature engineering. This will be performed by identifying and exploiting patterns in raw data and allowing the model to develop self-learning abilities.

In this paper, we introduce MalDet, a new approach to determining whether or not a Windows executable file is malicious. Regarding static analysis, MalDet goes above and beyond with two cutting-edge techniques that leverage deep neural networks. Convolutional neural

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10s
DOI: https://doi.org/10.17762/ijritcc.v11i10s.7629
***Article Received:** 02 June 2023 **Revised:** 27 July 2023 **Accepted:** 11 August 2023*

networks (CNNs) can learn from grayscale images. CNN can determine its structure by analysing the malware's local picture patterns. The raw binary file was used to create this grayscale image. Alternatively, opcode sequences can be used to train a Long-Short-Term Memory (LSTM). Decompiler tools may extract opcode sequences, which can be used to train an LSTM on harmful code features. This study uses a subsequence-based truncated back propagation technique to address this issue and enhance training efficiency. The bad traits or behaviours only emerge in specific opcode subsequences, even though attackers can insert destructive codes into otherwise innocent files. We created a subsequence selection mechanism to prevent LSTM from being deceived by these benign subsequences. MalDet learns features from raw data using these two networks, then combines their discriminant output with a metadata feature using a stacking ensemble to get a binary classification result for malware detection.

Compared to the N-gram baseline result, MalDet's detection accuracy was 98.89% based on our validation dataset consisting of 1/10 samples.

We contribute the following in this paper:

a) Our technology utilises deep neural networks for the identification of malware. By analysing raw data this way, CNN and LSTM networks can identify potentially dangerous file structures and code sequence patterns.

b) We design and implement MalDet, a malware detection approach. The gradient vanishing problem for LSTM, grayscale picture production, and extended sequence learning are handled. We solve noisy data processing and LSTM parallel computation. To maximise MalDet detection accuracy, we stack the networks' results.

c) MalDet undergoes various tests, including malware identification and family classification.

Below is the structure of the remaining work. In Section 2, we'll discuss similar work. Section 3 covers the dataset and MalDet detection process. The analysis and experiment results are in Section 4. Section 5 has the conclusion.

II. LITERATURE REVIEW

Many pattern-based picture analyses can be grouped into either the first (projectional) or second (semantic) categories [8]. Advances in visualization tools for computer security have made it easier to process massive data sets. It is usual practice to consider similarity measures between entities in terms of a two-dimensional similarity matrix [9,10]. After the similarity space is established, clusters of equal relevance are formed, and connected groups are placed next to each other for visualization, the high dimensionality of the data is irrelevant [8]. Both two- and three-dimensional documents take the form of points on maps. The proximity of every pair of indicators indicates the degree of similarity between the two papers. The more closely the details match the documents [11,12], the closer the resemblance.

To assess potential threats to a network, researchers have recently turned to image analysis [13]. In addition to the UserIDs and IP addresses, some semi-automated approaches involved analyzing secure shell (SSH) brute force attempts, which were colour-coded for the other vulnerabilities detected [14]. Visualization methods were also employed to simultaneously offer a holistic picture of numerous containers. These visuals demonstrate the communication between network packets, facilitating in-depth analysis by security professionals [15,31]. Another study [16] employed color codes to symbolize different forms of successful system links and trace the evolution of a virus attack like spear phishing using image-based analysis. Distances to other domains can be roughly estimated using their IP addresses, which are depicted in Figure 1 together with the "what," "where," and "when" of these connections. Different alerts appear as segmented concentric rings on the screen at regular intervals. Various coloured links represent attacks that could be launched against the same host. Most of these graphical approaches used text analysis of network data and logs to produce colourful visualizations that could be used to evaluate and detect malware attacks qualitatively. These limited probes centre on data analysis and network penetration within confined networks [19,20,32]. Moreover, in the age of Big Data, these processes are time-consuming, and many other strategies for representing and analyzing image textures are being explored [21].

Dataset and Methodology

This section consists of two parts Dataset and Methodology.

a) Dataset

We ran tests on a vast dataset that included 21,736 malicious samples provided by Microsoft and 20,650 benign samples we collected ourselves [30] to see how well MalDet performed. As a validation dataset, we selected 10% of the total samples.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10s
DOI: https://doi.org/10.17762/ijritcc.v11i10s.7629
***Article Received:** 02 June 2023 **Revised:** 27 July 2023 **Accepted:** 11 August 2023*

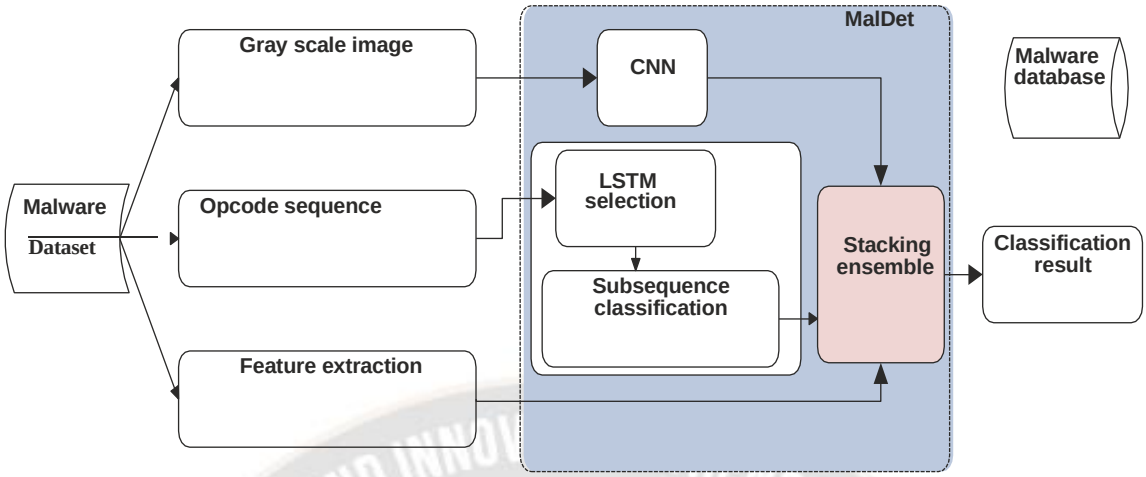

Figure 1. Proposed Mal Det malware detection process

b) Proposed Methodology

A. This section describes MalDet, a proposed approach for detecting malware based on CNN and LSTM networks. MalDet detects malware through binary categorization. It takes raw file *data and generates a malware likelihood. MalDet's method of finding objects can be divided into two parts (see Figure 1).*

Step 1 It begins with a binary Windows executable file, converts it to a grayscale image, and then decompiles it to obtain the opcode sequence and metadata feature.

Step 2: CNN/LSTM grayscale picture and opcode sequence training. To improve detection, we use a stacking ensemble to merge the outputs and metadata of two networks to get the final forecast. MalDet creates two feature sets from raw data.

First, using CNN, MalDet retains the structure of malicious files from the grayscale image. Second, it employs LSTM to learn the pattern of harmful code from the opcode sequence.

Further MalDet is explained into four major steps as below.

1. Grayscale Images through CNN: Nataraj et al. [17] use grayscale graphics to show how malware files are constructed.
2. Generation of Grayscale Images: Raw data must be processed and transformed into an image file to create a grayscale virus image. Only then can it be used? As receiving data, we use executable files and handle them like raw byte or binary stream files. If you take the binary stream file and convert each of its bits into a hexadecimal number, then you can conceive of the file as a hexadecimal stream file. When we add up every pair of hexadecimal digits, we get the exact grey value of a 256-level picture pixel. Hexadecimal numbers range from 0 to 16 and contain 0 to 16. This simple mapping change turns raw data into a grayscale image. The entire sequence of binary bits is divided into eight parts at regular intervals, corresponding to the degree of grey present in each pixel. These bits are then reassembled appropriately to produce the grayscale image.
3. Generation of Opcode Sequence using LSTM: A crucial component of MalDet that employs LSTM and opcode sequences to learn the characteristics and patterns of malicious configurations. Decompiled files yield opcodes. These steps demonstrate code logic and program operation. Thus, LSTM may identify harmful code sequences associated with high-level negative behaviour.
4. Extraction of Opcode Sequence: The opcode sequence must be extracted from the original executive files before it can be used for training. When we decode an executable with IDA Pro, we end up with an assembly file. Prevalent malware is decompiled into Intel x86 assembly instructions with the help of IDA Pro, a debugger and a decompiler. Then, we iterate through the. Asm file's lines, separating the sentences with the space character. This allows us to map each phrase to one of the typical Intel x86 assembly instructions in our specified opcode set. If the comparison succeeds, the opcode is kept; otherwise, the word is removed.

During this procedure, we discover that many decompiled files contain similar opcode subsequences. And hence, we need some new rules to filter out these redundant subsequences. Method 1 demonstrates how we parse the pseudocode into an opcode sequence. The average length of extracted opcode sequences is proportional to the size of the opcode collection. A wider variety of opcodes is available for use. Due to noisy data, LSTM will struggle to learn from a lengthy opcode set. Therefore, we consider all decompiled.asm files to be regular text files, and all instructions to be vocabularies. Then we generate frequency data and eliminate infrequently used words. Then, we employ a random forests model to categorise the words by using their frequencies as features. Each feature's importance can be ranked using a random forest. We prioritise languages that place a high value on the best quality. We use the 185-piece opcode set we obtain to derive opcode sequences. These opcode sequences must be converted to digital form before being fed into a neural network. In one-hot encoding, each of the N binary status bits represents N states with precisely one nonzero element, and a simple mapping change is needed to obtain a sparse vector like [0, 0, 0, 1, 0,..., 0]. Each opcode also has its one-hot variant.LSTM is used to learn very long sequences.

III. EXPERIMENTS AND EVALUATIONS

The main metrics to evaluate performance of proposed Method are defined as follows:

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10s
DOI: https://doi.org/10.17762/ijritcc.v11i10s.7629
***Article Received:** 02 June 2023 **Revised:** 27 July 2023 **Accepted:** 11 August 2023*

TPR is the probability of detecting the present malicious sample: (1)

FPR is the chance that a sample that is harmless will be mistakenly labelled as malware: (2)

Where,

(T+)x: It predicts the state when the state is already there.

(F-)y: False negative means that it doesn't predict the state when the state is present.

(T-)x: A true negative state doesn't predict the state when it's not there.

(F+)y: A false positive is when the state is predicted when it is not there.

Tables 1 and 2 show the proposed algorithm's CNN and LSTM tuning parameters. Figure 2,3 displays the ROC curves for Inception and VGG networks, while Figure 4 compares the proposed technique to previous literature.

TABLE 1. : Parameters list of CNN network

Type of CNN Network	layer's Name	Dimension	Size
Inception V3	In [Input]	[1, 1, 64, 64]	*
	Convo[Convolutional]	[1, 32, 64, 64]	32 5 × 5 Convolution kernel
	MP[Max Pooling]	[1, 32, 32, 32]	2 × 2, stride 1
	Drop[Dropout]	[1, 32, 32, 32]	*
	Convo[Convolutional]	[1, 64, 32, 32]	64 5 × 5 Convolution kernel
	MP[Max Pooling]	[1, 64, 16, 16]	2 × 2, stride 1
	Drop[Dropout]	[1, 64, 16, 16]	*
	FC[Fully Connected]	[1024, 1]	LR[Logistic Regression]
	Drop[Dropout	[1024, 1]	*
	Out[Output]	[1]	*
VGG	In [Input]	[1, 1, 64, 64]	*
	Convo[Convolutional]	[1, 32, 64, 64]	32 3 × 3 Convolution kernel
	Convo[Convolutional]	[1, 16, 64, 64]	16 3 × 3 Convolution kernel
	MP[Max Pooling]	[1, 16, 32, 32]	2 × 2, stride 1
	Drop[Dropout]	[1, 16, 32, 32]	*
	Convo[Convolutional]	[1, 32, 32, 32]	32 3 × 3 Convolution kernel
	MP[Max Pooling]	[1, 32, 16, 16]	2 × 2, stride 1
	Drop[Dropout]	[1, 32, 16, 16]	*
	FC[Fully Connected]	[32, 512]	512
	Drop[Dropout]	[32, 512]	*
	FC[Fully Connected]	[32, 1]	LR[Logistic Regression]
	Drop[Dropout]	[32, 1]	*
	Out[Output]	[1]	*

Table2. Parameters list of LSTM network

Parameter's Name	Value
Max Value of Iterations	$6.40E + 004$
Truncated BPTT length	120
Weights initialization	[−0.04, +0.04]
Sample Batch Size	30
Rate of learning	$2.00E$- 03
Activation function	Tanh
Drop Rate	0.50
Gradient regularization factor	10
Hidden Points	185

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10s
DOI: https://doi.org/10.17762/ijritcc.v11i10s.7629
***Article Received:** 02 June 2023 **Revised:** 27 July 2023 **Accepted:** 11 August 2023*

Table 3. MalDet Results.

Models	TPR (%)	FPR (%)	EER (%)	AUC	Accuracy (%)
N-gram	89.22	0.1	3.94	0.9864	93.21
LSTM	98.69	0.1	0.54	0.9999	99.13
CNN	96.92	0.1	1.55	0.9989	98.14
LSTM+ CNN+ MF (MalDet)	99.14	0.1	0.37	0.9999	99.89

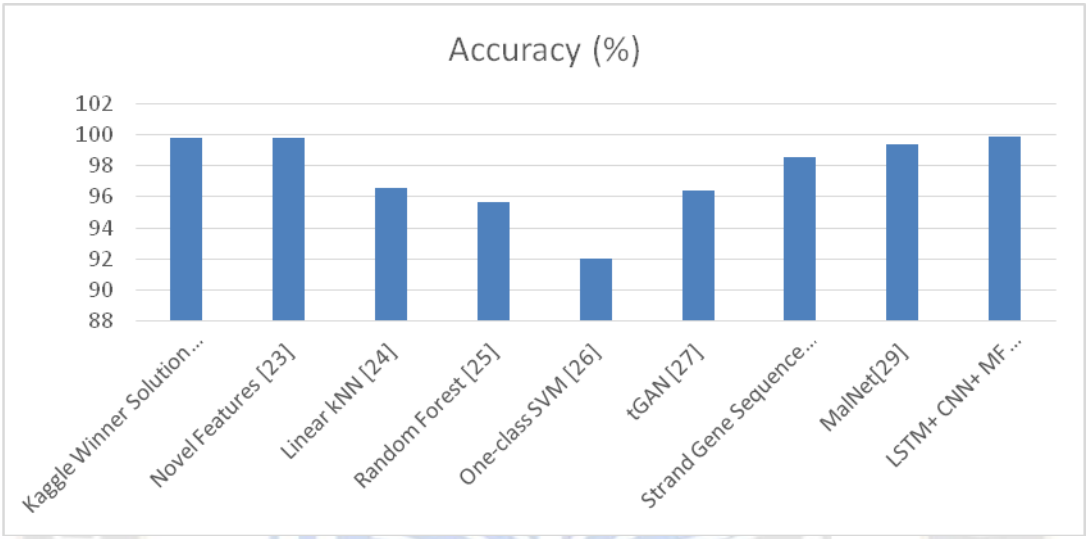

Figure 4. The comparison with other works.

IV. CONCLUSION

The presented method, MalDet, uses a stacking ensemble method on CNN and LSTM; the network is trained separately using grayscale images and opcode sequences from raw binary executive files. MalDet detected malware with a 99.89% level of accuracy, 99.14% TPR, and 0.1% FPR. We also classify malware families to compare our work to others.

REFRENCES

[1] "Kaspersky Security Bulletin 2016. Overall statistics for 2016," https://securelist.com/kaspersky-security-bulletin-2016-execu-tive-summary/76858/.

[2] "McAfeeLabsThreatsReportinJune2017,"https://www.mcafee.com/us/resources/reports/rp-quarterly-threats-jun-2017.pdf.

[3] Gopinath, M., and Sibi Chakkaravarthy Sethuraman. "A comprehensive survey on deep learning based malware detection techniques." Computer Science Review 47 (2023): 100529.

[4] Mishra, Anupama, and Ammar Almomani. "Malware Detection Techniques: A Comprehensive Study." Insights: An International Interdisciplinary Journal 1.1 (2023): 1-5.

[5] G. Willems, T. Holz, and F. Freiling, "Toward automated dynamic malware analysis using CWSandbox," IEEE Security and Privacy, vol. 5, no. 2, pp. 32–39, 2007.

[6] K.Rieck,P.Trinius,C.Willems,andT.Holz,"Automaticanal- ysis of malware behavior using machine learning," Technical Report, University of Mannheim, 2009.

[7] M.Zakeri,F.FarajiDaneshgar,andM.Abbaspour,"Astatic heuristic approach to detecting malware targets," Security and CommunicationNetworks,vol.8,no.17,pp.3015–3027,2015.

[8] Cao N, Cui W. Introduction to text visualisation. Atlantis Press; 2016.

[9] Keim D. Information visualisation and visual data mining. IEEE Trans Vis Com- put Graph 2002;8(1):1–8.

[10] Few S. Information dashboard design - the effective visual communication of data. Sebastopol, CA: O'Reilly; 2006.

[11] Diakopoulos N, Elgesem D, Salway A, Zhang A, Hofland K. Compare clouds: vi- sualizing text corpora to compare media frames. In: Proceedings of IUI Work- shop on Visual Text Analytics; 2015.

[12] Diakopoulos N, Elgesem D, Salway A, Zhang A, Hofland K. Compare clouds: vi- sualizing text corpora to compare media frames. In: Proceedings of IUI Work- shop on Visual Text Analytics; 2015.

[13] Shiravi H, Shiravi A, Ghorbani A. A survey of visualisation systems for network security. IEEE Trans Vis Comput Graph 2012;18(8):1313–29.

[14] Balakrishnan WB. Security data visualisation. SANS Institute Inc; 2014.

[15] Zhang TY, Wang XM, Li ZZ, Guo F, Ma Y, Chen W. A survey of network anomaly visualisation. Sci China Inform Sci 2017;60(12):121101.

[16] Shanks W. Enhancing intrusion analysis through data visualisation. SANS Insti- tute,Inc; 2015.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10s
DOI: https://doi.org/10.17762/ijritcc.v11i10s.7629
***Article Received:** 02 June 2023 **Revised:** 27 July 2023 **Accepted:** 11 August 2023*

[17] Foresti S, Agutter J, Livnat Y, et al. Visual correlation of network alerts. IEEE Comput Graph 2006;26:48–59.

[18] Wagner M, Sacha D, Rind A, Fischer F, Luh R, Schrittwieser S, Keim DA, Aigner W. Visual Analytics: foundations and experiences in malware analysis. In: Othmane Lotfi ben, Jaatun Martin Gilje, Weippl Edgar, editors. CRC/Taylor and francis in book: empirical research for software Security: foundations and experience. Publisher: CRC/Taylor and Francis; 2017. p. 139–71.

[19] Thu Thi Minh Nguyen, Ngan Truong Nguyen, Du Xuan Nguyen, Cong Thanh Tran, Doan Quang Tri. (2023). Mapping of Top-Soil Salinity Zoning in the Coastal Area of Ben Tre Province, Vietnam. International Journal of Intelligent Systems and Applications in Engineering, 11(4s), 473–490. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/2705

[20] Conti G. Security data visualisation - graphical techniques for network analysis. San Francisco: No Starch Press; 2007.

[21] Marty R. Applied security visualisation. Upper saddle river. NJ: AddisonWesley; 2009.

[22] Nataraj L, Karthikeyan S, Jacob G, Manjunath BS. Malware images: visualisation and automatic classification. In: Proceedings of the 8th international sympo- sium on visualisation for cyber security. ACM; 2011. p. 4.

[23] L. Wang, "Microsoft Malware Classification Challenge (BIG 2015) First Place Team: Say No To Overfitting," https://github.com/xiaozhouwang/kaggle Microsoft Malware/blob/master/Saynotooverfitting.pdf.

[24] M. Ahmadi, D. Ulyanov, S. Semenov, M. Trofimov, and G. Giacinto, "Novel feature extraction, selection and fusion for effective malware family classification," in Proceedings of the 6th ACM Conference on Data and Application Security and Privacy, (CODASPY '16), pp. 183–194, USA, March 2016.

[25] Robert Roberts, Daniel Taylor, Juan Herrera, Juan Castro, Mette Christensen. Leveraging Machine Learning for Educational Data Mining. Kuwait Journal of Machine Learning, 2(1). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/176

[26] B. N. Narayanan, O. Djaneye-Boundjou, and T. M. Kebede, "Performance analysis of machine learning and pattern recog- nition algorithms for Malware classification," in Proceedings of the 2016 IEEE National Aerospace and Electronics Conference and Ohio Innovation Summit, (NAECON-OIS '16), pp. 338–342, USA, July 2016.

[27] F. C. Garcia and F. P. Muga, "Random forest for malware classification," https://arxiv.org/abs/1609.07770.

[28] E. Burnaev and D. Smolyakov, "One-class SVM with privi- leged information and its application to malware detection," https://arxiv.org/abs/1609.08039.

[29] J. Kim, S. Bu, and S. Cho, "Malware detection using deep transferred generative adversarial networks," in Proceedings of the International Conference on Neural Information Processing (ICONIP), 2017.

[30] Pallathadka, D. H. . (2021). Mining Restaurant Data to Assess Contributions and Margins Data . International Journal of New Practices in Management and Engineering, 10(03), 06–11. https://doi.org/10.17762/ijnpme.v10i03.128

[31] J. Drew, M. Hahsler, and T. Moore, "Polymorphic malware detection using sequence classification methods and ensembles: BioSTAR 2016 Recommended Submission - EURASIP Journal on Information Security," EURASIP Journal on Information Security, vol. 2017, no. 1, article no. 2, 2017.

[32] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel, "Adversarial perturbations against deep neu- ral networks for malware classification," https://arxiv.org/abs/1606.04435.

[33] Microsoft Malware, https://www.kaggle.com/c/malware-classification.

[34] Bajaj, K., Jain, S. and Singh, R., 2023. Context-Aware Offloading for IoT Application using Fog-Cloud Computing. International Journal of Electrical and Electronics Research, 11(1), pp.69-83.

[35] Bajaj, K., Sharma, B., Singh, R., Kumar, M. and Chowdhury, S., 2022, December. A comparative analysis of cloud based services platform. In 6th Smart Cities Symposium (SCS 2022) (Vol. 2022, pp. 243-247). IET.