An intelligent obfuscated mobile malware detection using deep supervised learning algorithms

Padmavathi Ganapathi[1], Roshni Arumugam[2], Shanmugapriya Dhathathri[3]
[1]Department of Computer Science, School of Physical Sciences and Computational Sciences, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, India
[2]Centre for Cyber Intelligence, School of Physical Sciences and Computational Sciences, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, India
[3]Department of Information Technology, School of Physical Sciences and Computational Sciences, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, India

Article Info

Article history:

Received Jun 13, 2023
Revised Jan 2, 2024
Accepted Feb 12, 2024

Keywords:

Classification
Deep learning
Obfuscated malware
Performance evaluation
Supervised learning

ABSTRACT

Obfuscated mobile malware (OMM) is a malicious software in mobile that hides to avoid detection and annihilation. These types of malwares are thorny to identify due to their inevitable nature. Deep learning (DL) algorithms are the most desirable to detect obfuscated malware based on the 'n' number of iterations with adjustable weights and neurons. This study investigates the accurate detection of OMM using significant DL algorithms such as multi-layer perceptron (MLP), self-organizing maps (SOM), long short-term memory (LSTM) networks, auto encoders (AE), and convolutional neural network (CNN) based on appropriate parameter tuning. The dataset taken for the study is CICMalMem2022 that contains 58,596 samples with 57 features which is basically designed for OMM detection. The dataset comprises Spyware, Ransomware, Trojan horse, and Benign. The DL models are evaluated based on performance metrics such as precision, recall, accuracy, training accuracy, test accuracy, validation accuracy, training loss, validation loss and receiver operating characteristic (ROC) curve. Based on the experimental evaluation, the study reveals that LSTM outperforms with 100% accuracy and MLP achieves 99.9% accuracy in detecting and classifying the OMM using deep supervised learning (SL) mechanism.

Corresponding Author:

Padmavathi Ganapathi
Department of Computer Science, School of Physical Sciences and Computational Sciences
Avinashilingam Institute for Home Science and Higher Education for Women
Coimbatore –641043, Tamilnadu, India
Email: padmavathi_cs@avinuty.ac.in

1. INTRODUCTION

Obfuscated mobile malware (OMM) is one of the significant threats in mobile and network-connected devices. OMM acts as a non-malicious one to bypass the security check and enters the device anonymously. The malware identification system detects and blocks the malware based on the pre-defined signature patterns. Once the system encounters the related signature pattern, it takes the necessary actions to safeguard the device. OMM is increasing day by day due to its inevitable nature. These types of malwares are difficult to identify in terms of their newly created signature patterns, varying with the pre-defined patterns frequently [1]-[5]. A quarterly report published by the Kaspersky security network highlights that mobile malware is emerging as obfuscated in nature and targeted 4.9 million attacks in Q1 2023 [6]. Out of which, ransomware attacks have climbed by over 37% in 2023, with an average enterprise ransom demand of $5.3

million and a payment average of over $1,000,000 billion [7]. Over 1 billion malware programs are currently operational, and 560,000 new malware pieces are found every day, according to James [8].

Due to the growth of malware detection techniques using artificial intelligence methods, malware is transformed into obfuscated ones, which will be hard to interpret. There are three common techniques used to obfuscate malware, which includes encryption, tokenization, and data masking. During 2020, malware activity that propagated from one employee to another was noted in 61% of the organizations, according to Comparitech's statistical report 2022 [9]. The infection rate was at its highest since the SOES survey's launch in 2016 at 74% in 2021, 75% in 2022, and 78% in 2023 [9].

At present, OMM has become an eminent area in security and by employing various machine learning (ML) techniques [10]. OMM can be differentiated between advertent and normal activities. The critical factor is determining a pertinent ML technique for obtaining an optimal result for OMM detection. Currently, ML models are prone to misclassifying the results by considering the minimal malicious behaviour as an outlier and omitting them during classification [11]. Therefore, exploring ML algorithms to attain higher efficacy for OMM is a challenging one. In recent research, the deep learning (DL) models are widely applied since they have the capability of handling OMM data misinterpretation successfully. However, DL models are robust in nature to detect the malware obfuscation type of programs eminently [12], [13].

Intelligent OMM detection is the need of the day, and it can be done by developing significant DL models and the robust model based on the highest efficacy rate is recommended for OMM detection to subdue the challenges. As a result, the present work proclaims ineffectiveness in detecting such OMM. The primary contribution of this research are: i) a framework is developed using DL techniques for OMM detection and classification; ii) DL models such as multi-layer perceptron (MLP), self–organizing maps (SOM), long short term memory (LSTM), auto encoders (AE), and convolutional neural network (CNN) suitable for the CICMalMem_2022 dataset are developed to recognize OMM; iii) significant parameter tuning is carried out in DL model development to attain the accurate detection of OMM; and iv) comparison and evaluation of the developed DL models and recommending the most desirable model for OMM detection.

This paper is further divided as section 2 addresses the relevant existing research. Section 3 deals with the proposed framework and methods involved in the detection of OMM using DL techniques. Section 4 explains the experimental findings obtained in each step of the devised method and section 5 concludes the work with its future scope.

2. BACKGROUND

This section focuses on the existing related works on detecting malware/mobile malware/OMM using DL techniques. A DL-droid framework is presented in [14] to detect and classify the malware/benign samples. The results reveal that the DL model has a detection rate of 97.8% with dynamic features and a detection rate of 99.6% with dynamic + static features. The performance is evaluated using the true positive rate (TPR), false positive rate (FPR), true negative rate (TNR), false negative rate (FNR), precision, recall, accuracy, and running time. In this research, they have incorporated the models including Naive Bayes (NB), support vector machine (SVM) linear, SVM radial basis function (RBF), J48, Partial C4.5, random forest (RF), supervised learning (SL), and DL. A De-Lady DL framework is devised to identify the malware [15]. The neural network-based De-Lady model obtains a 98.08% detection rate and a 98.84% F-measure. The metrics used to assess the models include accuracy, F-measure, and error rate. The researchers applied the K-nearest neighbor (KNN), NB, SVM linear, decision tree (DT), RF, XGBoost, and De-Lady models.

A fuzzy logic framework is designed to detect the malwares [16]. Analytical hierarchy process (AHP) fuzzy model achieves 90.54% based on the info gain value evaluation parameter. This paper devices the model based on a fuzzy logic mechanism with AHP. Gohari *et al.* [17], consider the network traffic and develop the convolutional neural network and long short-term memory (CNN–LSTM) model that achieves 97.29-99.79% in classifying the malware families. The metrics, recall and precision are used to gauge the model's effectiveness. A CNN model is developed to classify malware/benign [18]. The model retains up to 97.60%. The efficacy of the model is determined based on its precision, recall, accuracy, and F1-score. The model includes ML and DL, such as CNN, NB, logistic regression (LR), KNN, and RF.

Alazab *et al.* [19] presents the model for an automated intrusion detection system (IDS) to detect obfuscated malicious JavaScript code, which identifies the malicious attacks with 94% for malicious samples and 81% for benign samples when the feature vector is 60. The IDS model is evaluated using TPR, FRP, and F-Measure. It examines the deep generative model for obfuscated malware detection leveraging both global and local features [20]. Class activation map (CAM) DL model is developed to detect malware with 97.47%, resulting in cutting-edge performance. Research gaps identified from the background study are: i) based on the existing relevant literature study, it is obvious that there are various DL models-based frameworks available to detect and classify the malware, mobile malware and OMM; ii) however, to detect OMM particularly, only limited works are available in the literature; iii) the comparison made between the various models is not done among the appropriate categories; and iv) apart from accuracy, precision, recall and F1–score, other parameters like training accuracy, test accuracy, validation accuracy, training loss, and validation loss must be considered to study the effectiveness of the models.

3. PROPOSED METHOD

This section describes the development of a significant DL model for OMM detection using five different DL algorithms. A standard framework is proposed to develop a suitable DL model based on the training and learning, so that the models are able to detect and classify the malicious OMM against a non-malicious one. Developing a significant DL model involves five phases, namely: i) data collection/data acquisition, ii) data cleansing/pre-processing, iii) feature scaling, iv) model deployment, and v) model evaluation. Figure 1 depicts the proposed framework to identify and classify the OMM and benign samples using DL techniques. The following sub-sections briefly discuss each phase of the proposed design in detail.

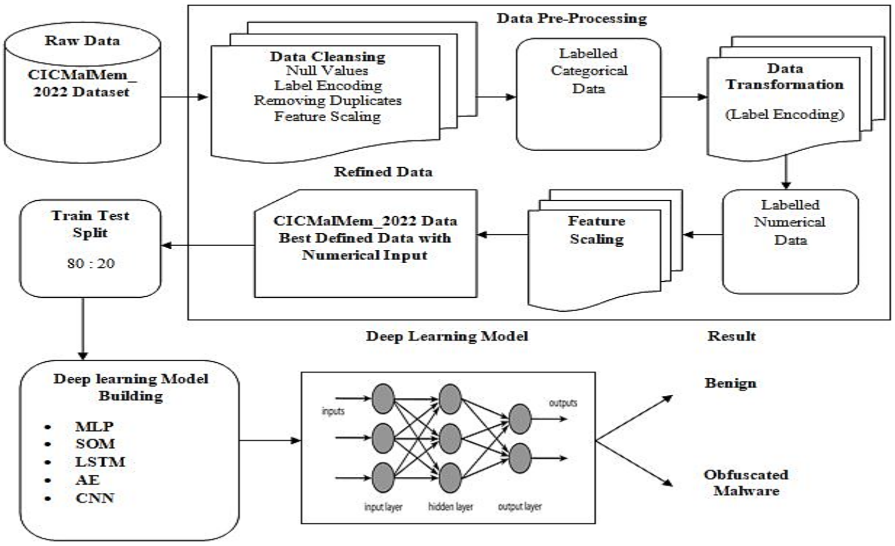


Figure 1. Proposed framework for OMM detection

3.1. Dataset

The first step is to acquire the data to develop a significant DL model for OMM detection. In this study, the CICMalMem_2022 dataset [21] is used. It is a benchmark dataset from the Canadian Institute of Cyber Security that contains malware memory analysis data. The dataset consists of 58,596 samples with 57 features that include OMM and benign data points.

3.2. Data pre-processing

To convert the raw data into a refined dataset, pre-processing is done. In this phase, the CICMalMem_2022 dataset undergoes appropriate methods of data pre-processing strategies to get further processed to support model development. It is used to ensure whether the dataset contains any missing values, irrelevant values or undefined values that may affect the performance of the models. The following pre-processing methods are applied to fine-tune the dataset: i) removing null values, ii) label encoding, iii) removing duplicates, and iv) feature scaling. After the data pre-processing phase, the raw CICMalMem_ 2022 dataset is transformed into a refined format by neglecting the extraneous values and replicas with 58,062 records with 57 features as labelled numerical data. Further, the data is split into 80% (46,449) for training and 20% (11,613) for testing.

3.3. Deep learning model for obfuscated mobile malware detection

DL has significant advantages over ML, owing to its ability to analyse a huge volume of data and develop the model with n number of iterations with varying neurons and weights to provide the best efficacy rate [22]-[25]. In this phase, five suitable DL models are developed to detect and classify the data points as OMM/benign.

3.3.1. Multi-layer perceptron

In an MLP, there are several neurons stacked on top of one another in the input, output, and one or more hidden layers. Perceptron neurons must have an activation function that enforces a threshold, like rectified linear units (ReLU) or sigmoid, in contrast to multilayer perceptron neurons, which can have any arbitrary activation function. Multiple layers of synthetic neurons or nodes make up a multi-layered neural network.

3.3.2. Self-organizing maps

The SOM output of an artificial neural network (ANN) is referred to as a “map.” It is a discretized low-dimensional (typically two-dimensional) representation of the input space for the training samples. SOMs are different from other ANNs in that they use competitive learning rather than error-correction learning (like backpropagation with gradient descent) and utilise a neighbourhood function to preserve the topological traits of the input space.

3.3.3. Long short-term memory

The LSTM, also known as a sequential network or an improved recurrent neural network (RNN), can preserve information. It can resolve the vanishing gradient problem of the RNN. LSTM networks were developed specifically to alleviate the long-term reliance problem that RNNs encounter. Because they have feedback connections, LSTMs differ from more traditional feed-forward neural networks. Due to this property, LSTMs may analyse whole data sequences (such as time series) without considering each individual data point. Instead, they can use the prior data in the sequence to assess new data.

3.3.4. Auto encoders

An unsupervised ML technique called an autoencoder neural network sets the target values to be the same as the inputs and uses backpropagation. If necessary, the original data can be reconstructed using compressed data. Three layers make up an autoencoder, including: i) encoder, ii) code, and iii) decoder.

3.3.5. Convolutional neural network

A ConvNet or CNN, models the connectivity arrangement between the neurons after how a creature’s visual brain is set up. Data is gathered, weights are assigned based on the distinct objects in the data, and then the objects are differentiated from one another through the neurons in CNN. The input layer, the convolutional layer, the ReLU layer, the pooling layer, and the fully connected (FC) layer make up the CNN architecture. For the purpose of learning non-linear functions, the FC layer is available. The OMM/benign data points in the CICMalMem_2022 dataset is detected and classified using the criteria listed in Table 1.

Table 1. DL models classification criteria for OMM detection

DL model	Mechanism	Activation function/classification criteria	Epoch	Batch size
MLP	Neural network	Back propagation	50	10/100
SOM	ANN	Best matching unit (BMU)	50	10/100
LSTM	RNN	ReLU SoftMax Sigmoid	50	10/100
AE	Feed forward neural network	Exponential linear unit (ELU)	50	10/100
CNN	CNN	ReLU Sigmoid	50	10/100

3.4. Evaluating the performance of the DL models developed

An important factor for assessing the efficacy of constructed DL models is performance evaluation. By evaluating the DL model’s performance for the taken data, one can derive the outcome of the models and be able to interpret the results in an appropriate way. This will also help to suggest a suitable model, especially for the taken problem, to provide an appropriate solution. In this research, the DL model performances are estimated using the metrics that include precision, recall, accuracy, training accuracy, test accuracy, validation accuracy, training loss, validation loss, and receiver operating characteristic (ROC) curve. Table 2 infers the description of the performance validation metrics with their respective formula.

Table 2. Performance evaluation metrics with its formula

Performance metrics	Description	Formula
Accuracy	The criterion utilized to evaluate the efficiency of the model across all classes is accuracy.	$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$ (13)
Precision	The ratio of successfully recognized positive samples to all positive samples (whether correctly or incorrectly detected) is used to calculate precision.	$Precision = TP/(TP + FP)$ (14)
Recall	It is determined by dividing the total samples that were correctly classified as positive by the percentage of samples that were positive overall.	$Recall = TP/(TP + FN)$ (15)
Train accuracy	The model's accuracy is measured against training data. It is also measuring the efficacy of model during the training process.	$Train\ data\ accuracy = (number\ of\ correct\ predictions)/(total\ number\ of\ predictions)$
Test accuracy	It is the correctness of the model as measured by validation data. It is typically a subset of the data that is used to validate the model's behaviour but is not utilized in the training process.	$Test\ data\ accuracy = (number\ of\ correct\ predictions)/(total\ number\ of\ predictions)$
Validation accuracy	The test results determine the model's correctness. This is usually checked once the actual training is over. The test accuracy is determined for model prediction based on the effective training data process.	$Validation\ data\ accuracy = (number\ of\ correct\ predictions)/(total\ number\ of\ predictions)$
Training loss	Examines the model's performance with the training data. To determine the training loss, errors for each sample in the training set are accumulated.	$Train\ data\ loss = sum\ of\ the\ errors\ in\ the\ training\ data$
Validation loss	Like how the training loss is computed, the mistakes for each sample in the validation set are combined to determine the validation loss.	$Validation\ data\ loss = sum\ of\ the\ errors\ in\ the\ validation\ data$
ROC curve	The efficacy of a classification model at each level of categorization is depicted on a graph called a ROC curve.	$TPR = TP/(TP + FN)$ (16) $FPR = FP/(FP + TN)$ (17)

*TP is true positive, TN is true negative, FP is false positive, and FN is false negative

4. EXPERIMENTAL RESULTS AND DISCUSSION

This section discusses the findings from the evaluation of DL models to identify and classify the OMM data points. Table 3 depicts the values obtained from the performance evaluation metrics carried out for the developed DL models. Table 3 clearly depicts the performance of the DL algorithms based on precision, recall, accuracy, training accuracy, test accuracy, validation accuracy, training loss, and validation loss. For a sensitive dataset, the evaluation is more focused towards precision parameter, since how well the DL model correctly identifies the positive samples is analyzed through precision metrics. Let us discuss the results of the model one by one, in MLP, the model provides an overall accuracy of 99.9%, precision and recall score of 99.9% with training loss ≤0.1728 and validation loss ≤0.1765. This indicates that the MLP model detects the positive samples correctly as positive as 99.9%, whereas the model training loss=validation loss (i.e.,) is equal which means the model perfectly fits with the data. Self-organizing maps achieves accuracy with 96.2%, precision and recall with 96.2%, training loss ≤0.3343 and validation loss ≤0.2947. This infers that the SOM model detects the positive samples appropriately as positive as 99.9% with training loss>validation loss, which in turn results in an underfitting model.

Table 3. Performance evaluation of DL models–analysis results

S.No.	DL model	Accuracy	Precision	Recall	Training accuracy	Validation accuracy	Test accuracy	Training loss	Validation loss
1.	MLP	0.9999	0.9999	0.9999	0.9999	0.9998	0.9997	0.1728	0.1765
2.	SOM	0.9624	0.9625	0.9624	0.9625	0.9625	0.9627	0.3343	0.2947
3.	LSTM	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.0061	0.0011
4.	AE	0.6156	0.5043	0.9997	0.6051	0.6750	0.9996	0.0037	0.0037
5.	CNN	1.000	0.6454	0.7657	1.0000	0.9500	1.0000	0.6107	0.6954

The LSTM model achieves accuracy, precision and recall with 100%, training loss, and validation ≤0.006. This indicate that the LSTM model performs excellently in detecting the positive samples accurately as positives with no loss in the system. In autoencoders, the model achieves an overall accuracy of 61.5%, precision of 0.50%, recall of 0.99%, training, and validation loss=0.0037. This denotes that the AE model detects the samples with a 99% recall value, but out of 99% detected samples, the positive samples that are correctly classified as positive are only 50% in precision value. However, AE model training loss=validation loss indicates that the model is fit with a low detection rate. In CNN, the accuracy is 100%, precision with 0.64%, recall with 0.76% and, training loss ≤0.6107 and validation loss ≤0.6954. In the CNN model, out of

76% recall values detected, 64% of samples are positively classified as positive samples and validation loss > training loss results in an overfitting issue.

The overall analysis of the performance evaluation results for the developed DL models shows that the MLP detects and classifies the OMM samples accurately classified as OMM with 99.9%, and LSTM model achieves the same by 100% efficacy rate. The accurate detection of OMM is achieved through appropriate hyperparameter tuning for DL models, as mentioned in Table 3. Since, all the DL models are tuned on the aspect of deep SL mechanism because the CICMalMem_2022 dataset is a labelled one which suits for SL paradigm. Hence, the DL models are well trained based on the target label, and this in turn, predicts the test samples perfectly, which results in a high efficacy rate. So, the models like MLP and LSTM based on supervised DL gained the results effectively and served as the best fit models for the detection and classification of OMM/benign data points. Figure 2 shows the performance assessment of the DL models graphical detecting obfuscated malware.

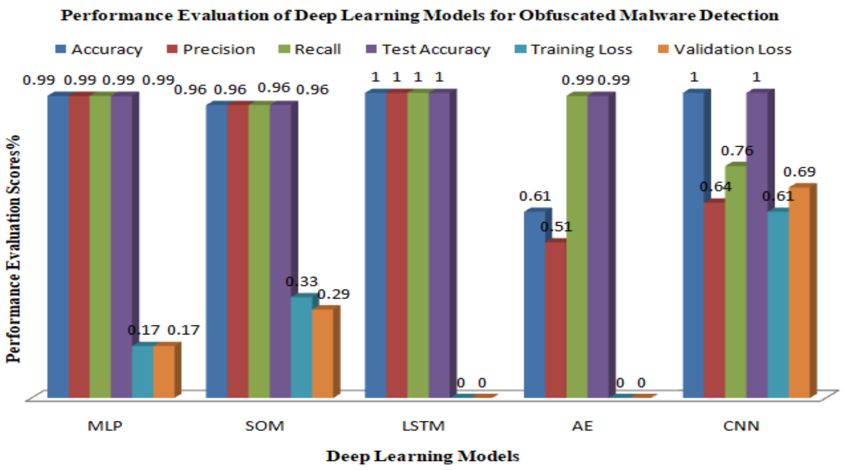
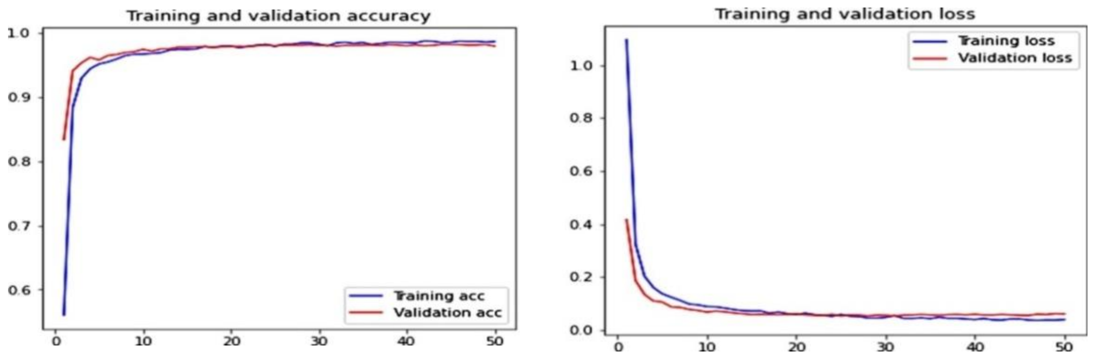


Figure 2. Performance assessment of DL models for detecting OMM

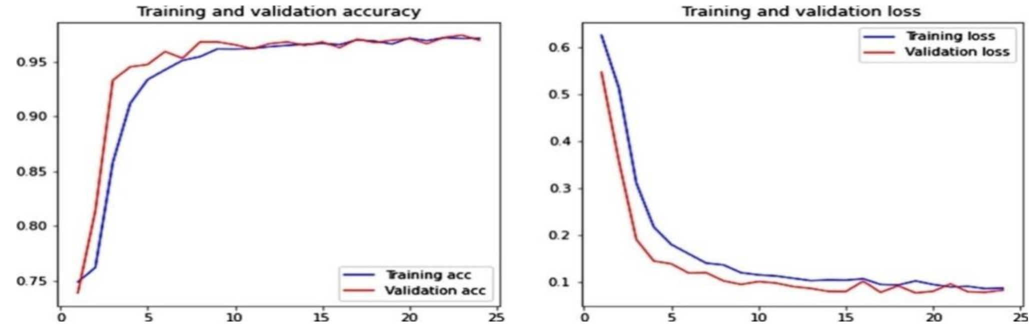
Figures 3 to 6 depict the efficiency of the LSTM and MLP DL models for OMM detection with batch sizes of 10 and 100. Figure 3(a) shows the performance of the LSTM model with a batch size of 10 in terms of training and validation accuracy and Figure 3(b) depicts its training and validation loss. Figure 4(a) shows the performance of the LSTM model with a batch size of 100 in terms of training and validation accuracy and Figure 4(b) depicts its training and validation loss. Figure 5(a) shows the performance of the MLP model with a batch size of 10 in terms of training and validation accuracy and Figure 5(b) depicts its training and validation loss. Figure 6(a) shows the performance of the MLP model with a batch size of 100 in terms of training and validation accuracy and Figure 6(b) depicts its training and validation loss. Table 4 presents the DL model appropriate hyperparameters for OMM detection.



batch size vs accuracy
(a)

batch size vs accuracy
(b)

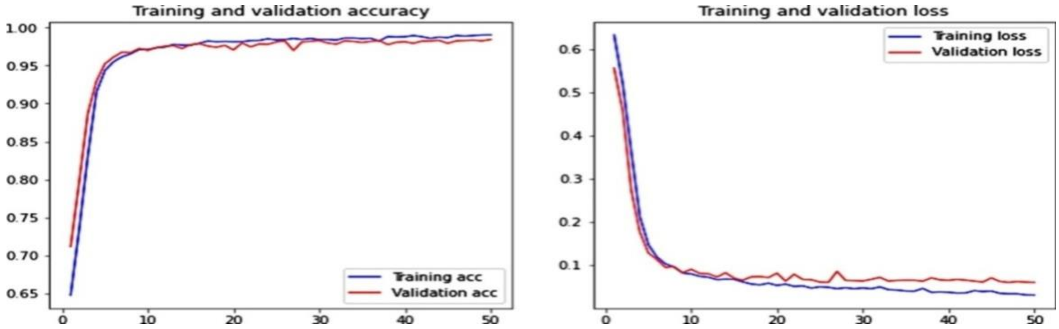
Figure 3. Performance of the LSTM model with a batch size of 10; (a) training and validation accuracy and (b) training and validation loss



batch size vs accuracy
(a)

batch size vs accuracy
(b)

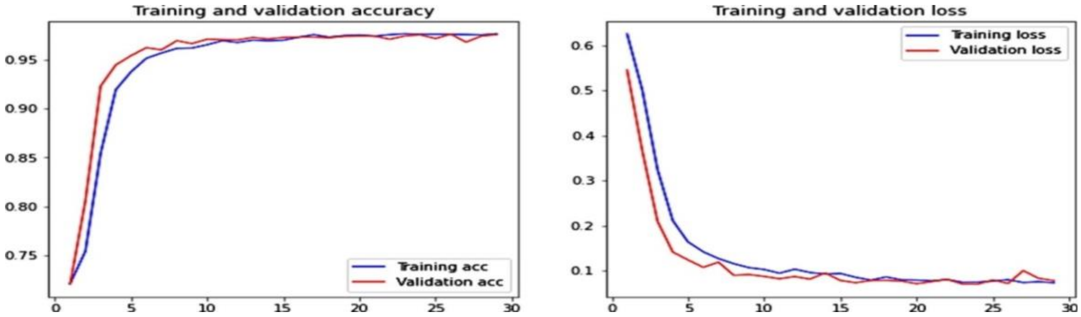
Figure 4. Performance of the LSTM model with a batch size of 100; (a) training and validation accuracy and (b) training and validation loss



batch size vs accuracy
(a)

batch size vs accuracy
(b)

Figure 5. Performance of the MLP model with a batch size of 10; (a) training and validation accuracy and (b) training and validation loss



batch size vs accuracy
(a)

batch size vs accuracy
(b)

Figure 6. Performance of the MLP model with a batch size of 100; (a) training and validation accuracy and (b) training and validation loss

Figures 3 to 6 indicate that the LSTM and MLP DL models achieve the highest accuracy with a batch size of 10 compared with a batch size of 100 in OMM detection accurately. Although, the developed DL models outperform best with a batch size of 10, attaining the highest efficacy rate as well as with less training and validation loss. In a nutshell, this study recommends that the LSTM model attain 100% accuracy and the MLP model obtains 99.9% accuracy with minimum validation loss between 0.1-0.001% in the detection and classification of OMM using DL techniques.

Table 4. DL models hyperparameter setup for OMM detection

Layer	Hyperparameter
Input layer	Input size: 58596×57
Hidden layer	Two hidden layers Hidden units-128 Activation function–ReLU Optimizer–adaptive moment estimation (Adam)
Output layer	Two–OMM/Benign Activation function–sigmoid
Spatial dropout layer	Spatialdropout1: droprate=0.6 Spatialdropout2: droprate=0.7 Spatialdropout3: droprate=0.3
Dropout layer	Dropout rate=0.7

5. CONCLUSION

OMM is progressively creating a major threat to networks and mobile devices. This research has proposed an intelligent framework to develop a significant DL model to detect and classify OMM and benign accurately. The CICMalMem_2022 dataset contains 58,596 samples with 57 features utilized to train and test the DL models, including MLP, SOM, LSTM, AE, and CNN. Based on the evaluation results, it is derived that the LSTM outperforms with 100% accuracy and MLP attains 99.9% accuracy to detect and classify the OMM samples as positive with low validation loss. Hence, the LSTM and MLP are recommended as the most desirable DL model to detect and classify the OMM in the CICMalMem_2022 dataset. The other models, like SOM, AE, and CNN perform well with reasonable detection rate along with high false negative rates, leading to misinterpretation, underfitting, and overfitting problems. In future, various DL models with different learning aspects such as unsupervised and semi-SL along with diverse model parameters with varying batch sizes are to be analyzed.

ACKNOWLEDGEMENT

This work is supported by DST-CURIE-AI project during 2021–2023 by Centre for Cyber Intelligence established under Centre for Machine Learning and Intelligence, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, Tamilnadu, India.

REFERENCES

[1] G. Canfora, F. Martinelli, F. Mercaldo, V. Nardone, A. Santone, and C. A. Visaggio, "LEILA: Formal Tool for Identifying Mobile Malicious Behaviour," *IEEE Transactions on Software Engineering*, vol. 45, no. 12, pp. 1230–1252, Dec. 2019, doi: 10.1109/tse.2018.2834344.

[2] G. Iadarola, F. Martinelli, F. Mercaldo, and A. Santone, "Call Graph and Model Checking for Fine-Grained Android Malicious Behaviour Detection," *Applied Sciences*, vol. 10, no. 22, p. 7975, Nov. 2020, doi: 10.3390/app10227975.

[3] G. Iadarola, F. Martinelli, F. Mercaldo, and A. Santone, "Image-based Malware Family Detection: An Assessment between Feature Extraction and Classification Techniques," in *Proceedings of the 5th International Conference on Internet of Things, Big Data and Security* 2020, pp. 499-506, doi: 10.5220/0009817804990506.

[4] M. Jerbi, Z. C. Dagdia, S. Bechikh, and L. B. Said, "On the use of artificial malicious patterns for android malware detection," *Computers& Security*, vol. 92, p. 101743, May 2020, doi: 10.1016/j.cose.2020.101743.

[5] T. L. Carrier, P. Victor, A. Tekeoglu, and A. H. Lashkari, "Detecting Obfuscated Malware using Memory Feature Engineering," in *Proceedings of the 8th International Conference on Information Systems Security and Privacy*, pp. 177-188, 2022, doi: 10.5220/0010908200003120.

[6] A. Kivva, "IT threat evolution Q1 2023. Mobile statistics," *Kaspersky*, Jun. 06, 2023. [Online]. Available: https://securelist.com/it-threat-evolution-q1-2023-mobile-statistics/109893/#:~:text=providing%20statistical%20data.Quarterly%20figures,34.8%25%20of%20all%20detected%20threats. (accessed: June 12, 2023).

[7] "2023 ThreatLabz State of Ransomware Report | Zscaler." https://info.zscaler.com/resources-industry-reports-2023-threatlabz-ransom-ware-report#:~:text=Ransomware%20attacks%20increased%20by%20over,a%20%245.3%20million%20average%20demand. (accessed May. 26, 2023).

[8] N. James, "AWS Penetration Testing report: Everything you should know!," Astra Security Blog, May 08, 2023.https://www.getastra.com/blog/security-audit/malware-statistics/#:~:text=560%2C000%20new%20pieces%20of%20malware, of%20%244.54%20million%20per%20incident. (accessed May. 26, 2023).

[9] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 618-626, 2017, doi: 10.1109/ICCV.2017.74.

[10] V. Kouliaridis, K. Barmpatsalou, G. Kambourakis, and S. Chen, "A Survey on Mobile Malware Detection Techniques," *IEICE Transactions on Information and Systems*, vol. E103.D, no. 2, pp. 204–211, Feb. 2020, doi: 10.1587/transinf.2019ini0003.

[11] S. V. Kudrekar and U. R. Vinayakamurthy, “Fisher exact Boschloo and polynomial vector learning for malware detection,” *International Journal of Power Electronics and Drive Systems*, vol. 13, no. 3, pp. 2942–2952, Jun. 2023, doi: 10.11591/ijece.v13i3.pp2942-2952.

[12] A. D. Jasim and R. I. Farhan, “Intelligent malware classification based on network traffic and data augmentation techniques,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 30, no. 2, pp. 903–908, May 2023, doi: 10.11591/ijeecs.v30.i2.pp903-908.

[13] S. V. Kudrekar and U. R. Vinayakamurthy, “Classification of malware using multinomial linked latent modular double q learning,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 28, no. 1, pp. 577–586, Oct. 2022, doi: 10.11591/ijeecs.v28.i1.pp577-586.

[14] G. Iadarola, F. Martinelli, F. Mercaldo, and A. Santone, “Towards an interpretable deep learning model for mobile malware detection and family identification,” *Computers & Security*, vol. 105, p. 102198, Jun. 2021, doi: 10.1016/j.cose.2021.102198.

[15] V. Sihag *et al.*, “De-LADY: Deep learning based Android malware detection using Dynamic features,” *Journal of Internet Services and Information Security*, vol. 11, no.2, pp. 34–45, 2021, doi: 10.22667/JISIS.2021.05.31.034.

[16] J. M. Arif, M. A. A. Razak, S. R. T. Mat, S. Awang, N. H. Ismail, and A. Firdaus, “Android mobile malware detection using fuzzy AHP,” *Journal of Information Security and Applications*, vol. 61, p. 102929, Sep. 2021, doi: 10.1016/j.jisa.2021.102929.

[17] M. Gohari, S. Hashemi, and L. Abdi, “Android malware detection and classification based on network traffic using deep learning,” in *2021 7th International Conference on Web Research (ICWR)*, pp. 71-77, 2021, doi: 10.1109/ICWR51868.2021.9443025

[18] N. Zhang, Y.-A. Tan, C. Yang, and Y. Li, “Deep learning feature exploration for Android malware detection,” *Applied Soft Computing*, vol. 102, p. 107069, Apr. 2021, doi: 10.1016/j.asoc.2020.107069.

[19] A. Alazab, A. Khraisat, M. Alazab, and S. Singh, “Detection of Obfuscated Malicious JavaScript Code,” *Future Internet*, vol. 14, no. 8, p. 217, Jul. 2022, doi: 10.3390/fi14080217.

[20] J.-S. Kim and S.-B. Cho, “Obfuscated Malware Detection Using Deep Generative Model based on Global/Local Features,” *Computers & Security*, vol. 112, p. 102501, Jan. 2022, doi: 10.1016/j.cose.2021.102501.

[21] “Malware Memory Analysis | Datasets | Canadian Institute for Cybersecurity | UNB.”https://www.unb.ca/cic/datasets/malmem-2022.html. (accessed: 06-Jun-2023).

[22] M. Dhalaria and E. Gandotra, “A Hybrid Approach for Android Malware Detection and Family Classification,” *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 6, no. 6, p. 174, May 2021, doi: 10.9781/ijimai.2020.09.001.

[23] G. D’Angelo, F. Palmieri, and A. Robustelli, “A federated approach to Android malware classification through Perm-Maps,” *Cluster Computing*, vol.25, no.4, pp. 2487-2500, Feb. 2022, doi: 10.1007/s10586-021-03490-2.

[24] Z. H. Qaisar and R. Li, “Multimodal information fusion for android malware detection using lazy learning,” *Multimedia Tools and Applications*, vol. 81, no. 9, pp. 12077–12091, Mar. 2021, doi: 10.1007/s11042-021-10749-8.

[25] M. Antunes, L. V. F. De Oliveira, A. Seguro, J. Veríssimo, R. Salgado, and T. Murteira, “Benchmarking Deep Learning Methods for Behaviour-Based Network Intrusion Detection,” *Informatics*, vol. 9, no. 1, p. 29, Mar. 2022, doi: 10.3390/informatics9010029.

BIOGRAPHIES OF AUTHORS

Padmavathi Ganapathi she is the Dean-School of Physical Sciences and Computational Sciences. Professor in the Department of Computer Science, Avinashilingam Institute for Home Science and Higher Education for Women (Deemed to be University), Coimbatore, India. She can be contacted at email: padmavathi_cs@avinuty.ac.in.

Roshni Arumugam she is working as a Research Assistant under Centre for Cyber Intelligence–DST–CURIE–AI in the Department of Computer Science, Avinashilingam Institute for Home Science and Higher Education for Women (Deemed to be University), Coimbatore, Tamil Nadu, India. She can be contacted at email: roshini_cci@avinuty.ac.in.

Shanmugapriya Dhathathri she is an Assistant Professor and Head, Department of Information Technology, Avinashilingam Institute for Home Science and Higher Education for Women (Deemed to be University), Coimbatore, India. She can be contacted at email: shanmugapriya_it@avinuty.ac.in.