International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

An Intelligent Recurrent Model with Deep Swarms for Early Malware Detection in Internet of Things Platforms

Jamilu Musa[1], Kabiru Musa Ibrahim[2], Usman Ali[3] & Mustapha Abdulrahman Lawal & Ismail Zahraddeen Yakubu[4]

[1,2,3]*Department of Computer Science, Abubakar Tafawa Balewa University, Bauchi*

[6]*Department of Management and Information Technology SRM Institute of Science and Technology, Chennai India*

A B S T R A C T

Malware poses a significant threat to cybersecurity, with new variants constantly emerging to evade detection by traditional antivirus software. Deep learning has shown promise in addressing this challenge by enabling the development of more sophisticated and accurate malware classifiers. However, training deep learning models often involves tuning a large number of hyperparameters, which can be a time-consuming and computationally expensive process. Previous research has explored various deep learning architectures for malware classification, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and LSTM networks. Additionally, optimization techniques such as grid search, random search, and Bayesian optimization have been used to tune hyperparameters. However, these methods may suffer from limitations such as high computational cost or suboptimal performance. Our approach seeks to overcome these limitations by leveraging PSO to efficiently search the hyperparameter space of LSTM networks. Experimental result shows that the proposed approach achieved the best performance in terms of accuracy, precision, recall and F-Measure. the proposed model achieved the best classification accuracy of 99.1% as against the existing algorithm including KNN which achieved 96.1%, SVM achieved 96.1%, RF achieved 97.8% and achieved DT 97.8 respectively. Similarly, the proposed system achieved higher precision, recall and F-score as against the existing algorithm including KNN which achieved 0.851, 0.892, 0.866 SVM achieved 0.862, 0.885, 0.871, RF achieved 0.873, 0.964, 0.912 and DT achieved 0.873, 0.964, 0.912 respectively.

Keywords: Malware, Machine Learning, Deep Learning, Particle swarm optimisation and long short-term memory.

1. Introduction

Malware, short for Malicious Software, encompasses a wide range of malicious code designed to harm computer systems and user data. The evolution of malware has led to variant forms such as viruses, Trojans, ransomware, and more, with a continuous surge in their creation and distribution across the internet. According to AV-Test Institute, approximately 350,000 new malicious codes and potentially unwanted applications emerge daily, highlighting the relentless growth of malware threats (Lu, 2019).

Despite the significant improvement of cyber security mechanisms and their continuous evolution, malware is still among the most effective threats in the cyber space. Malware analysis applies techniques from several different fields, such as program analysis and network analysis, for the study of malicious samples to develop a deeper understanding on several aspects, including their behavior and how they evolve over time(Ucci, Aniello, & Baldoni, 2019).

Nowadays, with the booming development of Internet and software industry, more and more malware variants are designed to perform various malicious activities. Traditional signature-based detection methods cannot detect variants of malware(Lu, 2019). Coping with malware is getting more and more challenging, given their relentless growth in complexity and volume. One of the most common approaches in literature is using machine learning techniques, to automatically learn models and patterns behind such complexity, and to develop technologies to keep pace with malware evolution(Ucci et al., 2019).

Machine learning (ML) techniques have gained traction in malware detection due to their ability to automatically learn patterns and models from data. The dynamic nature of malware, categorized into first and second-generation malware, poses challenges for traditional detection methods. First-generation malware maintains the same structure, while second-generation malware alters its structure while retaining its functionality, making detection and quarantine more challenging (Sharma, Krishna, & Sahay, 2019).

One prevalent approach in malware detection is the use of machine learning algorithms, particularly deep learning models such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs). These models can learn intricate patterns and behaviors from malware samples, enabling more accurate detection and classification (Shukla, Kolhe, PD, & Rafatirad, 2019). Dynamic behavior-based detection methods, although time-consuming, offer robust detection capabilities by analyzing interactions with the operating environment (Moser, Kruegel, & Kirda, 2007).

Recent research has focused on optimizing deep learning models like LSTM (Long Short-Term Memory) for malware classification (Li, Liu, Gao, & Reiter, 2010). LSTM networks are well-suited for handling temporal dependencies in data, making them effective in capturing the evolving nature of malware threats (Sharma et al., 2019) . These optimized models aim to reduce the reliance on manual feature engineering and improve overall prediction performance in malware detection tasks.

While existing studies (Shhadat, Hayajneh, & Al-Sharif, 2020) have shown promising results with machine learning algorithms in malware detection, challenges such as data skewness, low frequency of benign files, and lack of generalization remain. Addressing these challenges requires comprehensive datasets, balanced class distributions, and robust algorithms capable of handling complex malware behaviors (Sharma et al., 2019; Shukla et al., 2019). Therefore, malware detection and classification are critical areas of research in cybersecurity, with machine learning playing a pivotal role in developing effective detection mechanisms (Shhadat et al., 2020). Optimized deep learning models like LSTM offer promising avenues for improving accuracy and generalization in malware detection, paving the way for enhanced cybersecurity measures in the digital landscape (Rhode, Burnap, & Jones, 2018)..

2. Literature Review

Malwares are classified into two categories - first generation malware and second-generation malware. The category of malware depends on how it affects the system, functionality of the program and growing mechanism. The former deals with the concept that the structure of malware remains same, while the later states that the keeping the action as is, the structure of malware changes, after every iteration resulting in the generation of new structure. This dynamic characteristic of the malware makes it harder to detect, and quarantine. The most important techniques for malware detection are signature based, heuristic based, normalization and machine learning. In past years, machine learning has been an admired approach for malware defenders(Sharma et al., 2019).

Malware analysis, detection and classification has allured a lot of researchers in the past few years. Numerous methods based on machine learning (ML), computer vision and deep learning have been applied to this task and have accomplished some pragmatic results(Shukla et al., 2019). A large number of researches have been published on how to detect malware. Malware detection can be simply considered as a binary classification problem, and traditional anti-virus software usually relies on static signature-based detection method, which has a significant limitation. For example, (Vinayakumar, Alazab, Soman, Poornachandran, & Venkatraman, 2019) Proposed Android malware detection system using Long Short-term Memory (LSTM), The experiment achieved the Android malware detection of 0.939 on dynamic analysis and 0.975 on static analysis on well-known datasets. Similarly, (Lu, 2019) propose a novel and efficient approach to perform static malware analysis, which can automatically learn the opcode sequence patterns of malware on 123 benign files. In terms of malware detection and malware classification, the evaluation results show the proposed method can achieve average AUC of 0.99 and average AUC of 0.987 in best case, respectively.

However, some minor changes in malware can change the signature, so more malware could easily evade signature-based detection by encrypting, obfuscating or packing for example, (Sharma et al., 2019) study the frequency of opcode occurrence to detect unknown malware by using machine learning technique, Code obfuscation technique is a challenge for signature based techniques used by advanced malware to evade anti-malware tools. The proposed approach uses Fisher Score method for the feature selection and five classifiers used to uncover the unknown malware. In the proposed approach, Random forest, LMT, J48 Graft, and NBT detect malware with 100% accuracy which is better than the accuracy (99.8%) reported by (Ahmadi, Ulyanov, Semenov, Trofimov, & Giacinto, 2016). Meanwhile, the zero-day malware can also evade this detection approach. The dynamic analysis is not susceptible to code obfuscation techniques (Moser et al., 2007), so it is a more effective malware detection method. Dynamic behavior-based malware detection methods usually need a secure and controlled environment, such as virtual machine, simulator, sandbox, etc. Then the behavior analysis is performed by using the interaction information with the environment such as API calls and DLL calls. Although these techniques have been widely studied, they have also been confirmed to be less efficient enough when applied to large dataset(Li et al., 2010). Dynamic behavior-based malware detection methods are quite time consuming and require considerable attention to protect the operating environment from contaminated.

At present, a number of malware detection methods combined with machine learning techniques have been proposed. For example, (Roseline & Geetha, 2018) proposed an Intelligent Malware Detection using Oblique Random Forest Paradigm. The proposed oblique random forest model outperforms other decision tree models in terms of performance measures on three datasets, they author suggested that using deep learning techniques can reduce the risk of newly evolving malwares.

Similarly, (Zhu et al., 2019) puts forward a new method based on Machine Learning (ML), including Logistic Regression (LR), Random Forest (RF), Support Vector Machine (SVM), Boosted Tree (BT), Back- Propagation Neural Networks (BPNN) and Naive Bayes Classifier (NBC). The simulated experiments show that the new method is potential to detect a GSMem attack, with low False Positive Rates (FPR) and low False Negative Rates (FNR). Also, (Martín, Hernández, & de los Santos, 2019) proposed Machine-Learning based analysis and classification of Android malware signatures in which ML models provide outstanding classification results (F1-score of 0.84 in test) and have shown to further identify some of the Unknown family classes into either Harmful/Adware threat. the previous works have achieved good enough performance. However, most of these methods manually extract malware features which are used to train a machine learning classifier(Sharma et al., 2019).

Malware analysis, detection and classification has allured a lot of researchers in the past few years. Numerous methods based on machine learning (ML), computer vision and deep learning have been applied to this task and have accomplished some pragmatic results(Shukla et al., 2019). A large number of researches have been published on how to detect malware using deep learning techniques, for example, (Yan, Qi, & Rao, 2018) propose MalNet, a novel malware detection method that learns features automatically from the raw data using CNN and LSTM networks to learn from grayscale image and opcode sequence, respectively. The evaluation result shows that MalNet achieves 99.88% validation accuracy for malware detection. In addition, malware family classification experiment on 9 malware families was taking to compare MalNet with other related works, in which MalNet outperforms most of related works with 99.36% detection accuracy and achieves a considerable speed-up on detecting efficiency comparing with two state-of-the-art results on Microsoft malware dataset.

Moreover, (Rhode et al., 2018) investigate the possibility of predicting whether or not an executable is malicious based on a short snapshot of behavioral data and find that an ensemble of recurrent neural networks are able to predict whether an executable is malicious or benign within the first 5 seconds of execution with 94% accuracy. (Shukla et al., 2019) proposed Stealthy Malware Detection using RNN-based Automated Localized Feature Extraction and Classifier, the proposed approach achieves up to 11% higher detection accuracy compared to the CNN-based sequence classification and hidden Markov model (HMM). Similarly, (Vinayakumar et al., 2019) proposed Robust Intelligent Malware Detection Using Deep Learning. The performances obtained by deep learning architectures outperformed classical MLAs in static, dynamic and image processing-based malware detection and categorization

Furthermore, (Xiao, Li, Chen, & Li, 2020) proposed an Android Fragmentation in Malware Detection It was observed that there is improvement of detection rates in those fine-grained classifiers compared to a single classifier but prone to be an expensive task and is limited to small-size of dataset and code coverage. (Xiao et al., 2020) proposed an effective malware classification framework with automated feature extraction based on deep convolutional neural networks. Experimental results show that MalFCS can obtain excellent classification better results compare to the state-of-the-art although, there is Model Overfitting issues must be mitigated in future work.

More recently, (Shhadat et al., 2020) Uses Machine Learning Techniques to Advance the Detection and Classification of Unknown Malware, achieving accuracy improvements over all binary and multi-classifiers. The highest accuracy was achieved by DT is 98.2% for binary classification and 95.8% by RF for multi-class class. However, one of the major limitations and weakness of the work is that the precision and recall are slightly lower; this is maybe due to the data skewness and the low frequency of the benign files in the original dataset. Moreover, the work lack generalization since the dataset is not bigger and unbalanced.

The study in (Rhode et al., 2018) has shown that Recurrent neural network deals with temporal decencies in nonlinear data sets more comfortably when compare to other machine learning algorithms, RNN is more robust against overfitting to the other algorithms and can learnt a more generalizable representation of the difference between malicious and benign files, Thus, in order to address these challenges, this research work put forward an optimized LSTM algorithm to classify malware with higher prediction performance.

3. Methodology

This research addresses a new framework based on malware classification using deep recurrent neural network. The proposed malware detection approach uses LSTM deep recurrent neural network and optimized the training with swarm optimization algorithm in other to achieved an improve performance. As shown in Figure 1, the malware detection methodology can be simply divided into the data processing stage and modeling stage. The framework consists of the following steps:

A. Preprocessing: Convert malware samples into sequences of binary data or features suitable for input to an LSTM network.

B. LSTM Network Architecture: Design an LSTM network architecture capable of processing sequence data effectively.

C. PSO Optimization: Implement PSO to optimize the hyperparameters of the LSTM network, including the number of layers, number of units per layer, learning rate, dropout rate, etc.

D. Training and Validation: Train the LSTM network using the optimized hyperparameters on a training dataset and validate its performance on a separate validation dataset.

E. Evaluation: Evaluate the performance of the LSTM model on a testing dataset using metrics such as accuracy, precision, recall, and F1-score.

3.1 Model Architecture

This research addresses a new framework based on malware classification using deep recurrent neural network. Fig. 1 illustrates the flowchart of the proposed framework. Primarily, the first step of the framework involves the data set and a preprocessing layer. The first part of the entire strategy is the dataset stage, which reads all the defined dataset. Different operations are executed on the datasets including connecting to the database that includes dataset loading and reading files. The proposed strategy has been examined on well-known malware benchmark datasets to ensure the reliability of our proposed strategy. The sequence input layer inputs the preprocesses time series data into the LSTM network model.

3.1.1 The Long Short-Term Memory (LSTM)

LSTM is a modified network of RNN proposed to learn long-range dependencies across time-varying patterns. Generally, LSTM is a second order recurrent neural network that solves the vanishing and exploding gradients issue by replacing RNN simple units with the memory blocks in recurrent hidden layer. A memory block is a complex processing unit in LSTM with many units. It is composed of one or many memories cell, adaptive multiplicative gating units (input, output and forget) and a self-recurrent connection with a fixed weight 1.0. It serves as a short-term memory with a control from adaptive multiplicative gating units. The input and output flow of a cell activation of a memory cell is controlled by input and output gate respectively.

3.1.2 Model Integration with PSO for Parameter Selection and Optimization

Mostly PSO, have been widely applied to neural network models, such as MLP and RNN, and used in various hybrid approaches for financial time series forecasting to optimize technical analysis or train the neural network(Chung & Shin, 2018). In this study, we propose a hybrid approach of the LSTM network integrating PSO to find the customized time window and number of LSTM units for financial time series prediction. Since the LSTM network uses past information during the learning process, a suitably chosen time window plays an important role in the promising performance. If the window is too small, the model will neglect important information, while, if the window is too large, the model will be overfitted on the training data. This study consists of two stages, which are as follows. The first stage of the experiment involves designing the appropriate network parameters for the LSTM network.

We use LSTM network with sequential input layer followed by LSTM layers, a fully connected layer with optimal number of hidden neurons in each hidden layer is investigated by PSO. Initial weights of network are set as random values, and the network weight is adjusted by using a gradient-based “Adam” optimizer, which is famous for its simplicity, straightforwardness, and computational efficiency. The method is appropriate for problems which have large data and parameters, and also has strength in dealing with non-stationary problems with very noisy and sparse gradients. The proposed architecture is described below.

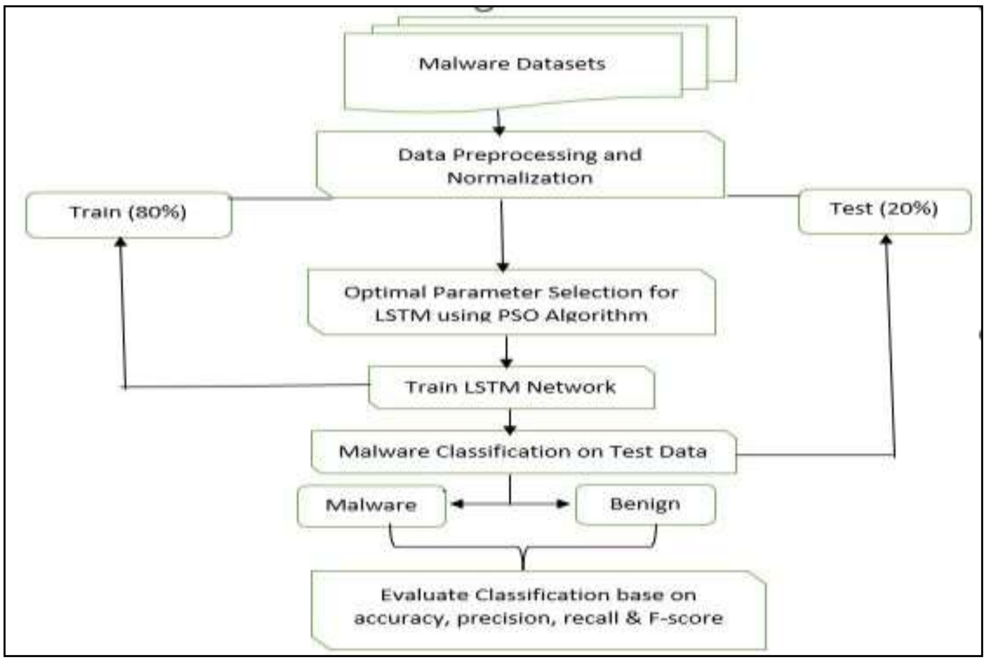
Malware Datasets
Data Preprocessing and Normalization
Train (80%)
Test (20%)
Optimal Parameter Selection for LSTM using PSO Algorithm
Train LSTM Network
Malware Classification on Test Data
Malware
Benign
Evaluate Classification base on accuracy, precision, recall & F-score

Fig 1. Framework for the proposed method

3.2 Dataset Description

We use publicly available data obtain from UCI machine learning repository. They collected a total of 1156 files consisting of 984 malicious files and 172 benign files. The files used in the dataset contain several formats such as *.exe*,*. pdf* and*. docx*. We consider the following malware in the study: Dridex, *Locky*, *TeslaCrypt*, *Vawtrak*, *Zeus*, *DarkComet*, *CyberGate*, *CTB-Locker* and *Xtreme*. We estimate the performance of the classifiers depending on different types of metrics: Accuracy, Precision, Recall, and F1-score.

4. Experimental Result

This chapter presents the result obtained after simulating the network on MATLAB 2020a. The results are presented in tabular and graphical forms which are analyzed using standard performance classification evaluation metrics as specified during the design. After the simulation, the classification accuracy was used to estimate the accuracy of the classifier.

4.2 Implementation of the System

The developed deep learning classifier model was implemented using MATLAB R2020a. The computer used is HP laptop running on Windows 7 Operating System with 8GB RAM, 1 TB HDD and Pentium ® Core i7 processor.

4.2.1 Experimental Settings

LSTM networks can learn long-term dependencies between time steps of sequence data. This study uses the LSTM layer to look at the sequence in both forward and backward directions. The malware dataset was read as xls file. The number of search agents was set to 30 and the number of iterations was set to 500. The details of the selected benchmark function were loaded. To achieve the desired output, the input Size was set to 4, a number of hidden units was set to 100 and the number of classes was also set to 2. The description of parameters is defined as;

1. Input Size is the argument to the sequence Input Layer function. it is the dimension of the % feature, that is, the number of rows in the matrix in each cell.

2. Num Hidden Units is a parameter of the LSTM Layer function that sets the number of hidden units contained in the LSTM network.

3. Number of Class is the argument of a fully Connected Layer, which is the number of labels. For this research, the number of wolves to be identified.

Table 1. Parameters Settings for the proposed algorithm

1. Parameter	**2. Settings**
3. Sequence Input Layer	4. 4
5. LSTM Layer	6. 100
7. Fully Connected Layer	8. 2
9. SoftMax Layer	10. 1
11. Classification Layer	12. 1
13. Max Epochs	14. 7
15. Mini Batch Size	16. 27
17. Verbose	18. False
19. Learn Rate Schedule	20. Piecewise
21. Maximum Iterations	22. 500
23. input Size	24. 4
25. Num Hidden Units	26. 100
27. Num Classes	28. 2

Next, we defined all the layers of the network as shown in Table 1. We specify the training options for the classifier and Set Max Epochs to 7 so that the network makes 7 passes through the training data. We set a Minibatch Size of 27 so that the network looks at 13 training signals at a time and Specify Plots as "training-progress" to generate plots that show the training progress as the number of iterations increases. we set Verbose to false to disable printing the table output that corresponds to the data shown in the plot. To classify the test data. We specify the same mini-batch size used for training which was then used to calculate the classification accuracy of the predictions. This research uses the PSO algorithm to optimize the best weight of the network by searching through the defined agent as specified during the design. The PSO has been shown to perform better with recurrent neural networks (LSTM) than the other optimizers discussed in the literature.

4.3 Results Presentation

Evolutionary algorithms, mostly PSO, have been widely applied to neural network models, such as MLP and RNN, and used in various hybrid approaches for prediction and classification to optimize technical analysis or train the neural network. We adopt an evaluation sub-module to benchmark the deep learning architectures based on Static analysis. The performance of various classical machine learning and deep learning for static malware detection and classification are evaluated on publicly available datasets with collected samples of benign and malware. The variants of deep learning architectures are proposed by carefully following a hyperparameter tuning approach. Various trials of experiments are run for different classical machine learning algorithms (MLAs) and deep learning architectures.

This research addresses a new framework based on malware classification using PSO-LSTM base architecture. In this section, the proposed techniques were tested on the benchmark datasets against the most widely malware classification approaches, which are SVM, RF, DT and KNN. For each one, the precision, recall, accuracy, and F-measure were recorded. The experiments were performed on a machine with Intel Corei7. the simulation result is analyzed and compared in three different phases including accuracy and training performance.

4.3.1 Results Evaluation Based on Classification Accuracy

Table 2. below show the results obtain by the proposed system when compare with the benchmark paper using the same datasets. It further demonstrates the superiority of the proposed model against the other existing algorithm. The proposed system achieved the best performance in terms of accuracy, precision, recall and F-Measure.

Table 2. Overall Results for binary classifications of malware on same datasets

Algorithm	**Accuracy %**	**Precision**	**Recall**	**F-score**
PSO-LSTM	99.1	0.961	0.972	0.950
KNN	96.1	0.851	0.892	0.866
SVM	96.1	0.862	0.885	0.871
RF	97.8	0.873	0.964	0.912
DT	97.8	0.873	0.964	0.912

The superior performance derived from the PSO–LSTM model may be explained by the fact that the globally investigated time window and architecture of the LSTM network enhanced the efficiency of the learning process and prevented unnecessary computations. The results suggest that appropriate tuning and optimization of the network parameters is an important condition to achieve satisfactory performance. Despite the fast growth of deep learning algorithms, it is a very difficult task to find an optimal set of parameters of deep architectures with expert knowledge. However, the experimental results demonstrate that the method used in this study can be an effective tool to determine the optimal or near-optimal model for deep learning algorithms, and showed the potential for its applicability.

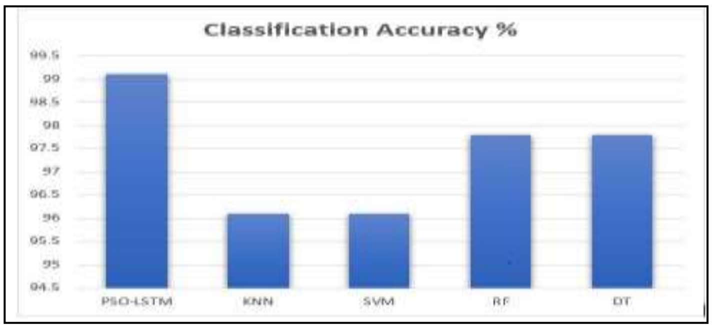

Fig.2: The accuracy achieved by our experiments against the referenced paper

Clearly, from Fig. 2 above, the proposed system achieved the best classification accuracy of 99.1 % as against the existing algorithm including KNN which achieved 96.1%, SVM 96.1%, RF 97.8% and DT 97.8 respectively.

4.3.2 Results Evaluation Based on Precision, Recall and F-Measure

Fig. 3 below shows the value obtained for both precision and recall from the simulation results obtained after several iterations. accuracy can be misleading in some cases. Precision and Recall help us further understand how strong the accuracy shown holds for a particular problem. F-Measure provides a single score that balances both the concerns of precision and recall in one number. In statistical analysis of binary classification, the F1 score (also F-score or F-measure) is a measure of a test’s accuracy. It is calculated from the precision and recall of the test, where the precision is the number of correctly identified positive results divided by the number of all positive results, including those not identified correctly, and the recall is the number of correctly identified positive results divided by the number of all samples that should have been identified as positive. In each case a higher value shows how confident the classification accuracy or performance can be relied upon.

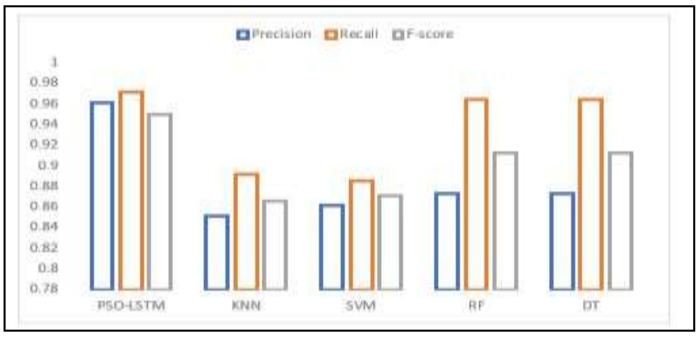

Fig. 3: The difference in precision, recall and F-score for the proposed PSO-LSTM and existing algorithms.

Clearly, from Fig. 3 above, the proposed system achieved higher precision, recall and F-score with 0.961, 0.972, 0.950 as against the existing algorithm including KNN which achieved 0.851, 0.892, 0.866 SVM achieved 0.862, 0.885, 0.871, RF achieved 0.873, 0.964, 0.912 and DT achieved 0.873, 0.964, 0.912 respectively. from the results presented above, it may be noticed that the proposed deep learning attains the first place in all the cases and DT attains the second place and is only inferior to SVM and KNN on the benchmark algorithm. Furthermore, it is also clear that the proposed deep learning attains the highest level of performance by obtaining a value very close to 99.1% in accuracy, recall and precision.

5. Conclusion and Future Work

Malware analysis, detection, and classification have attracted a lot of researchers in the past few years. Numerous methods based on machine learning (ML), computer vision, and deep learning have been applied to this task and have accomplished some pragmatic results. thus, this research proposes an optimized recurrent neural network (RNN) model to predict malicious behavior using machine activity data and demonstrate its capabilities are superior to other machine learning solutions that have previously been used for malware detection. This work is based on the basic idea of the deep learning technique to correctly determine whether the sample is malicious and non-malicious. The focus is on improving the classification performance using an optimized LSTM train using a particle swarm optimization algorithm. Based on the simulation results, we drive the following conclusions:

i. The proposed approach achieved the best performance in terms of accuracy, precision, recall and F-Measure. the proposed model achieved the best classification accuracy of 99.1% as against the existing algorithm including KNN which achieved 96.1%, SVM achieved 96.1%, RF achieved 97.8% and achieved DT 97.8 respectively. Similarly, the proposed system achieved higher precision, recall and F-score as against the existing algorithm including KNN which achieved 0.851, 0.892, 0.866 SVM achieved 0.862, 0.885, 0.871, RF achieved 0.873, 0.964, 0.912 and DT achieved 0.873, 0.964, 0.912 respectively.

ii. From the results presented above, it may be noticed that the proposed deep learning attains the first place in all cases and DT attains the second place and only inferior to SVM and KNN on the benchmark algorithms.

iii. The superior performance derived from the PSO –LSTM model may be explained by the fact that the globally investigated time window and architecture of LSTM network enhanced the efficiency of learning process and prevented unnecessary computations. The results suggest that appropriate tuning and optimization of the LSTM network parameters is an important condition to achieve satisfactory performance. Despite the fast growth of deep learning algorithms, it is very difficult task to find an optimal set of parameters of deep architectures by expert knowledge. However, the experimental results demonstrate that the method used in this study can be an effective tool to determine the optimal or near-optimal model for deep learning algorithms, and showed the potential for its applicability.

One of the major limitations of this study is data availability for ensuring more reliability of the model on wider datasets. Availability of data volume and understanding these malwares and their variability is much more complicated than other tasks. the amount of malware data that is needed to train an effective and robust deep learning model would be much more compared with other media hence training the model with larger data samples not only ensures reliability but also robustness and accuracy. Despite the promising results obtained using deep architectures, there remain several unsolved challenges facing the application of deep learning to malware detection. In particular, we highlight the following key issues: the previous works have achieved good enough performance. However, despite the result obtained in the aforementioned works as seen in the literature, there are still a lot of open issues that need to be addressed in the context of malware detection and classification since most of these methods manually extract malware features that are used to train a machine learning classifier. Thus, a need to reduce the cost of artificial feature engineering. It can be noticed that most of the methods are black-box solutions, which do not gain domain experts' insights. also, the application of the discussed network topologies on real raw Android malware samples instead of feature vectors of granular permissions in static analysis and profiled application features in the dynamic analysis is an open issue. the use of operands for malware detection will be a great enhancement to future research. Moreover, the problem of model overfitting and generalization still poses an open issue to most of the existing research.

Acknowledgments

I wish to appreciate the expert support of our supervisor Dr. K. I. Musa and Dr. Usman Ali for their immense contribution towards the success of this research.

References

Ahmadi, M., Ulyanov, D., Semenov, S., Trofimov, M., & Giacinto, G. (2016). Novel feature extraction, selection and fusion for effective malware family classification. Paper presented at the Proceedings of the sixth ACM conference on data and application security and privacy.

Chung, H., & Shin, K.-s. (2018). Genetic algorithm-optimized long short-term memory network for stock market prediction. Sustainability, 10(10), 3765.

Li, P., Liu, L., Gao, D., & Reiter, M. K. (2010). On challenges in evaluating malware clustering. Paper presented at the International Workshop on Recent Advances in Intrusion Detection.

Lu, R. (2019). Malware Detection with LSTM using Opcode Language. arXiv preprint arXiv:1906.04593.

Martín, I., Hernández, J. A., & de los Santos, S. (2019). Machine-Learning based analysis and classification of Android malware signatures. Future Generation Computer Systems, 97, 295-305.

Moser, A., Kruegel, C., & Kirda, E. (2007). Limits of static analysis for malware detection. Paper presented at the Twenty-Third Annual Computer Security Applications Conference (ACSAC 2007).

Rhode, M., Burnap, P., & Jones, K. (2018). Early-stage malware prediction using recurrent neural networks. computers & security, 77, 578-594.

Roseline, S. A., & Geetha, S. (2018). Intelligent Malware Detection using Oblique Random Forest Paradigm. Paper presented at the 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI).

Sharma, S., Krishna, C. R., & Sahay, S. K. (2019). Detection of advanced malware by machine learning techniques Soft Computing: Theories and Applications (pp. 333-342): Springer.

Shhadat, I., Hayajneh, A., & Al-Sharif, Z. A. (2020). The Use of Machine Learning Techniques to Advance the Detection and Classification of Unknown Malware. Procedia Computer Science, 170, 917-922.

Shukla, S., Kolhe, G., PD, S. M., & Rafatirad, S. (2019). Stealthy Malware Detection using RNN-Based Automated Localized Feature Extraction and Classifier. Paper presented at the 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI).

Ucci, D., Aniello, L., & Baldoni, R. (2019). Survey of machine learning techniques for malware analysis. computers & security, 81, 123-147.

Vinayakumar, R., Alazab, M., Soman, K., Poornachandran, P., & Venkatraman, S. (2019). Robust intelligent malware detection using deep learning. IEEE Access, 7, 46717-46738.

Xiao, G., Li, J., Chen, Y., & Li, K. (2020). MalFCS: An effective malware classification framework with automated feature extraction based on deep convolutional neural networks. Journal of Parallel and Distributed Computing.

Yan, J., Qi, Y., & Rao, Q. (2018). Detecting malware with an ensemble method based on deep neural network. Security and Communication Networks, 2018.

Zhu, W., Rodrigues, J. J., Niu, J., Zhou, Q., Li, Y., Xu, M., & Huang, B. (2019). Detecting air-gapped attacks using machine learning. Cognitive Systems Research, 57, 92-100.