Received: July 30, 2022. Revised: September 22, 2022.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Deep Ensemble Classifier for Ransomware Identification Using Digitalized DNA Genotyping System

Yuvaraj Saminathan[1]* **Robert Lourdusamy[1]**

[1]*Department of Computer Science, Government Arts College, Coimbatore, Tamilnadu, India*
*Corresponding author's Email: syuvi84phd@gmail.com

Abstract: Ransomware is a kind of virus that enciphers data on a victim's machine and only allows it to be decrypted when the user makes the payment. Understanding the strategies employed by cyber thieves is necessary to design effective protection against them. So, an active learning-based digital DNA sequencing engine called DNAact-Ran was developed to predict and identify ransomware data. But, it was not suitable for temporally changed DNA sequences. Hence, this article proposes a deep ensemble ransomware prediction (DeepERPred) model to handle the temporally changed DNA sequences and identify ransomware effectively. In this model, the raw data is converted into the required form. After that, the most relevant attributes are chosen from the preprocessed data using optimization algorithms. The selected attributes are classified by using the ensemble convolutional neural network and long short-term memory (CNN-LSTM) model. This ensemble classification is introduced to handle temporal changes in the DNA sequences by learning dependencies among present and previous occurrences of a sequence for identifying ransomware data. Finally, the investigational outcomes reveal that the DeepERPred model achieves a 90.67% of accuracy, whereas the classical models such as the modified decision tree (MDT), LightGBM, random forest (RF), artificial neural network (ANN), RansomDroid and DNAact-Ran attain 75.83%, 78.52%, 83.22%, 85.48%, 87.91% and 89.14%, respectively to identify ransomware.

Keywords: Ransomware, DNAact-Ran, Temporal changes, Deep learning, CNN, LSTM.

1. Introduction

The growth of the web as a worldwide technologically configured infrastructure has led to a huge and extraordinary expansion in the usage of Internet-connected systems, computers, and advanced electronics that have enriched human interactions and committed to strengthening society's life [1]. But, the technology's growth as a worldwide technological infrastructure has led to a significant increase in several types of cyber-attacks. Malware is such a harmful piece of code that has caused considerable harm to computer networks and also illegal practices, deception, scamming, and tribal cyber-terrorism [2].

Ransomware is a type of virus that keeps the suspect's important files by secretly enciphering the information on a participant's system to create it invisible and only retrieves information after the victim makes the payment in the form of currency [3-5]. The crypto-ransomware assaults the suspect's system by silently looking for and encrypting information on the victim's device. Users can regain access to their information only after paying compensation and receiving the secret key from the hackers. Crypto-ransomware normally does not encipher the whole physical disc yet instead attacks client-produced documents with certain tags such as .pdf, .jpg, and .doc files, which usually involve important and confidential user data [6-9].

The increased frequency of ransomware threats has forced administrations, organizations, and people to safeguard and backup their essential information. Nonetheless, owing to the extremely cost-effective behavior of such intrusions, the newest ransomware damages are always changing, and invaders are constantly developing more complex malware [10]. The present defense techniques for detecting, analyzing, and defending

against ransomware are ineffective and cannot keep up with the number of attacks. Most of the techniques are focused on analyzing the behavior of the malware. But, malware developers apply obfuscation methods like binary code packing for avoiding identification. The process of protecting the source code, such as the information and recovery procedure code, with the wrapped program itself is known as code packing [11]. When the wrapped application is run, the recovery procedure code replaces the source data and functions to their initial state. Layered polymorphic malware, which changes its software and also its decoding scheme and simply exposes a fraction of the script at any implementation phase and metamorphic malware, which changes its software in its decoded structure, leading to various ransomware strands with all fresh metamorphosis, are more difficult software anonymization methods.

Encrypting the function code is essential for destructive behavior creates it harder for existing signature-based activity assessment anti-malware systems to identify illegal code and the malware's activity. But, these identification systems are not highly powerful. For this reason, a machine learning-based ransomware identification system called DNAact-Ran has been designed by Khan et al. [12]. They developed a novel technique that employs a machine learning-based digital DNA sequencing model to identify and categorize ransomware. First, the major attributes were obtained from the preprocessed database by the multi-objective grey-wolf optimization (MOGWO) and binary cuckoo search (BCS) algorithms. After that, the digital DNA string was created for the decided attributes based on the model restraints of the DNA string and k-mer frequency vector. Moreover, it was categorized by the active learning algorithm as either goodware or ransomware. On the other hand, this algorithm was not able to perform well when digital DNA sequences were temporally changed. This active learning for training temporally changed sequences was not able to learn effectively if digital DNA sequences were highly sequential.

A DeepERPred model is designed in this paper to handle the temporally changed DNA sequences for ransomware identification. This DeepERPred model combines both CNN and LSTM classifier models to learn the relationship between various current and past events of sequences effectively. Based on this learning, the selected attributes are categorized as either ransomware or goodware effectively. Thus, this model can enhance the classification efficacy when using more attributes mined from the temporally changed sequences.

The remaining portions of this article are prepared as the following: Section 2 presents the work associated with ransomware identification. Section 3 explains the presented methodology and Section 4 demonstrates its effectiveness. Section 5 summarizes the entire study and gives the upcoming possibilities.

2. Literature survey

A new technique was developed [13] depending on the static evaluation for identifying ransomware. First, the different executable data was preprocessed to mine the attributes and the frequent patterns associated with the remarkable items. Then, the gain ratio was used to remove the redundant attributes and the most relevant attributes were classified by the RF as either goodware or ransomware. But, the accuracy was degraded when increasing the number of attributes

A new ransomware identification method was designed [14] that detects ransomware threats by analyzing the present condition of the network with data of a ransomware threat. The finite-state machine (FSM) framework was utilized for synthesizing the knowledge of the ransomware threat regarding the victim machine. But, it may fail to provide alerts if the ransomware affected the network either with no impact on the network resources or with no enciphering of the client documents.

A non-signature-based identification method was developed [15] depending on the effectual windows API call chains using supervised machine learning algorithms. So, an enhanced maximum-relevance and minimum-redundancy (EmRmR) filter scheme was applied to eliminate the irrelevant attributes and choose the most significant subgroup of attributes for representing the actual characteristic of the ransomware. But, it has high time complexity while increasing the number of attributes.

A novel framework was designed [16] that mines the new attributes from the ransomware database and conducts the categorization of the ransomware with benign data. In this framework, API-call sequences were reused for characterizing the behavior-based attributes for predicting ransomware by employing the (MDT) algorithm. But, it cannot effective for small-scale databases.

A 2-step mixed ransomware identification framework called Markov and RF schemes was modeled [17]. Initially, the Windows API call series prototype was focused and a Markov scheme was constructed to obtain the attributes of ransomware. Then, an RF scheme was used to identify the

malicious and benign call series. But, it needs to learn this framework in a highly varied and dynamic manner.

A new RansomDroid model [18] was developed based on a clustering-based unsupervised machine learning method to identify unforeseen Android ransomware. First, the raw malicious and benign databases were collected. Then, the android ransomware attributes were mined and the important attributes were chosen. These were fed to the Gaussian mixture model to identify the ransomware. But, it does not effective when the attackers applied obfuscation methods to create the source code vaguely and did not mine the dynamic attributes.

A new scheme called DeepAMD [19] was designed to defend against real-time Android malware using deep ANN. This scheme has static binary categorization and dynamic malware categorization. Initially, the data were categorized as malware and benign in the static unit. After that, the data categorized by the static unit was further categorized into various classes like adware, ransomware, SMS malware and scareware in the dynamic unit. But, it did not learn the time-variant attributes from the data, which also supports identifying ransomware.

A new and flexible ransomware identification system called a bi-layered model using LightGBM classifier [20] was designed, which combines both malware detection and ransomware detection modules. Initially, the malware among benign documents was detected. Then, that malware was categorized into various ransomware families. But, it did not evaluate the robustness of the system and needs a deep learning classifier to increase its efficiency.

2.1 Research contribution

From the literature, it is observed that most of the researchers developed statistical or machine learning models for ransomware classification. Such models were not able to handle a large amount of data or classes simultaneously so their efficiency was not satisfactory while increasing the number of attributes. Also, those models were not efficient to learn attributes obtained from highly varied (dynamic) data.

So, this research concentrates on handling temporally changed DNA sequences by learning the correlations between the current and different historical sequences efficiently to classify ransomware.

3. Proposed methodology

In this portion, the DeepERPred framework using ensemble CNN-LSTM is explained briefly. Fig. 1 demonstrates the overall schematic representation of this model.

3.1 Preprocessing and attribute selection

First, the raw ransomware database is acquired and preprocessed to transform the data into a desirable form. By preprocessing the database the missing values, imperfect and outlier data are eliminated from the actual database. Then, the data dimensionality is reduced by choosing the most relevant attributes using MOGWO and BCS algorithms [12]. MOGWO has two key elements such as a grid and an archive. Similarly, the BCS has a hunting space modeled as a d-cube. After that, the design restraints are determined and the k-mer frequency map is computed to create the digital DNA string which produces the training database. This database is further fed to the ensemble CNN-LSTM classifier to identify ransomware classes.

3.2 Ensemble CNN and LSTM Classification

This ensemble classification model has 2 major structures: CNN and LSTM. In the input layer, the most relevant attributes from the pre-processed

Table 1. Lists of notations

Notations	Description
x_t	Input of the input state
h_t	Result of a hidden state
h_{t-1}	Result of a previous hidden state
C_t	Cell state at period t
f_t	Forget gate
$\widetilde{C_t}$	New memory, i.e. cell update
o_t	Result of an output gate
w_f	Weight coefficient vector associated with the forget gate
w_i	Weight coefficient vector associated with the input gate
w_o	Weight coefficient vector associated with the output gate
w_C	Weight coefficient vector associated with the neuron condition vector
b_f	Offset value associated with the forget gate
b_i	Offset value associated with the input gate
b_o	Offset value associated with the output gate
b_C	Offset value associated with the neuron condition vector
$w_i x_t$	Input-to-hidden shift
$w_i h_{t-1}$	Hidden-to-hidden shift
$tanh$ and $Sigmoid$	Activation factors

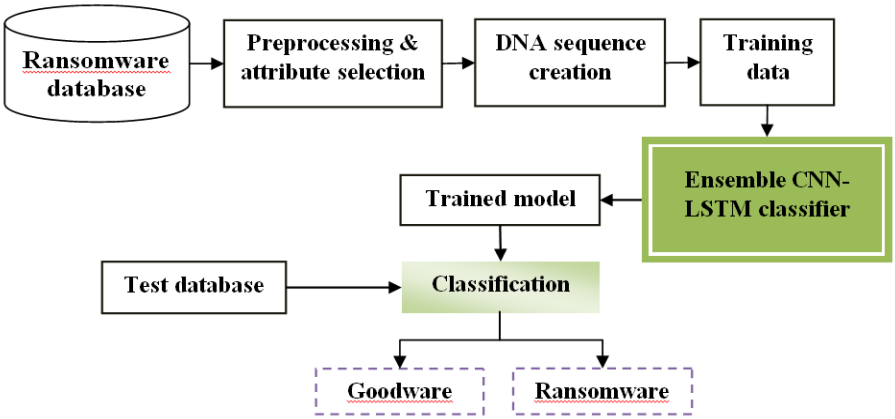

Figure. 1 Schematic overview of DeepERPred model for ransomware identification

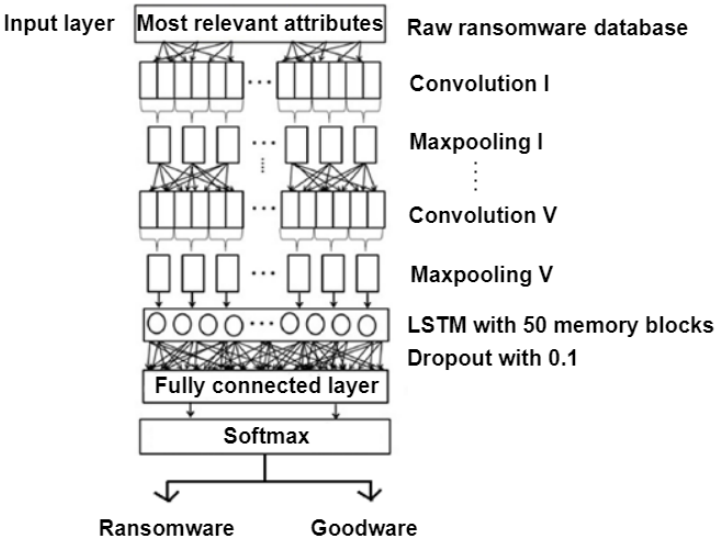

Figure. 2 Architecture of ensemble CNN-LSTM classification model

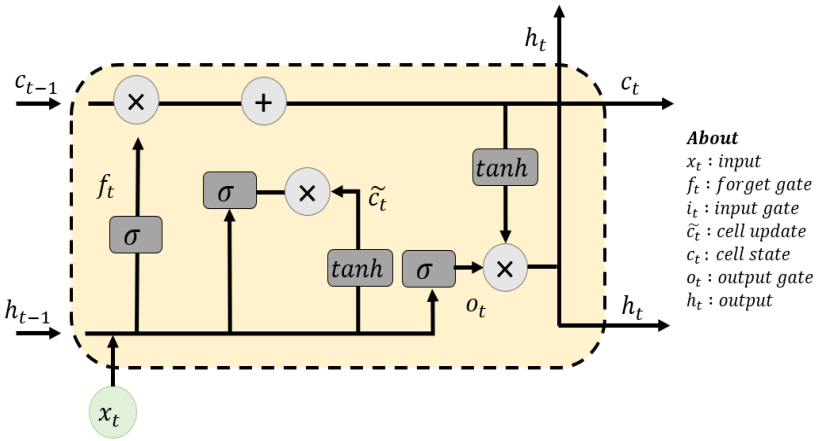

Figure. 3 Architecture of LSTM network model

database are given as input. It has 5 CNN layers and all layers follow max-pooling. Initial 2 CNN layers comprise 64 and 128 kernels with kernel size 3 and max-pooling with pooling size 2. Successive 2 CNN layers comprise 256 and 512 kernels with kernel size 3 and max-pooling with pooling size 4. The final CNN layer comprises 1024 kernels with kernel size 3 and max-pooling with pooling size 6.

This attribute map is then transferred to the LSTM layer which comprises 70 memory units to train the temporally changed attributes. These are further given to the fully connected and softmax layers to classify as either ransomware or goodware. Fig. 2 portrays the structure of the ensemble CNN-

LSTM classifier model for ransomware identification. Table 1 lists the notations utilized in this study.

In this classification, the LSTM illustrated in Fig. 3 has a 4-layer design with various connections. The input layer of the LSTM comprises three parameters: source, timeframe, and feature dimensions, where the timeframe is the sliding window size. The timeframe value evaluates how many past succeeding arriving data points can impact the current arriving data.

This variable's setting allows LSTM to learn long-term dependent information inside temporal data, increasing forecast accuracy. The hidden layer is made up of the number of neurons in it. Every gate's real processing is essentially a gate function performed by multiple hidden layer neurons that are coupled to the input layers. The weight coefficient vector weights and averages the input layers. Following that, the offset vector is incorporated to obtain the result of the hidden layer.

The output layer contains the number of hidden layer neurons as well as the output size. The LSTM memory cell's function is defined by

$$f_t = sigmoid\big(w_f[x_t, h_{t-1}] + b_f\big) \quad (1)$$

$$i_t = sigmoid(w_i[x_t, h_{t-1}] + b_i) \quad (2)$$

$$o_t = sigmoid(w_o[x_t, h_{t-1}] + b_o) \quad (3)$$

$$\widetilde{C_t} = tanh(w_C[x_t, h_{t-1}] + b_C) \quad (4)$$

In Eqns. (1)-(4), w_f, w_i, w_o and w_C are the weight coefficient vector associated with the hidden layers, input and output gates and the neuron condition vector, correspondingly. Additionally, b_f, b_i, b_o and b_C denote their corresponding offset values (i.e., bias values, which prevent overfitting problems). Based on this, C_t and h_t are determined by

$$C_t = f_t C_{t-1} + i_t C_t' \quad (5)$$

$$h_t = o_t tanh(C_t) \quad (6)$$

In Eqns. (5) & (6), C_t represent the cell-state and h_t represent the hidden state that serves as the solution of the block over t. The tensors i_t, f_t and o_t indicate the input, forget and output gates. The functions $w_i x_t$ and $w_i h_{t-1}$ denote the input-to-hidden shift and hidden-to-hidden shift.

The outcome from this LSTM is then passed to the fully connected and softmax layers to get the final solution i.e., the given attributes are either ransomware or goodware classes.

Algorithm:

Input: Raw ransomware database

Output: Ransomware or goodware

Begin

Pre-process the raw database to eliminate the missing values;

Apply MOGWO and BCS to choose the most important attributes;

Determine the design restraints and k-mer frequency map to create the digital DNA string;

Obtain the training database having the most relevant attributes;

Apply ensemble CNN-LSTM classifier;

Get the trained classifier model and validate using the test data;

Identify and categorize ransomware families and goodware data;

Evaluate the efficiency of the classifier;

End

4. Experimental results

In this section, the efficiency of the DeepERPred model is analyzed by implementing it in JAVA version 1.8. Also, the efficiency is compared with the existing algorithms (discussed in section 2): DNAact-Ran [12], RF [13], MDT [16], RansomDroid [18], ANN [19] and LightGBM [20]. The comparative analysis is conducted in terms of different metrics used in classifier analysis. This experiment uses the real-time database from https://github.com/PSJoshi/Notes/wiki/ databases.

The obtained database encompasses 1524 instances and 30970 attributes of which 582 are ransomware and 942 are goodware. The ransomware instances belong to various groups that are classified by this model including goodware, Critroni, CryptLocker, CryptoWall, KOLLAH, Kovter, Locker, MATSNU, PGPCODER, Reveton, TeslaCrypt and Trojan-Ransom. Also, various attributes extracted by this model are API instances (API), modifications of the lost documents (DROP), registry code functions (REG), document functions (FILES), a modification of the documents engaged in document functions (FILES_EXT), document index functions (DIR) and entrenched sequences (STR).

4.1 Accuracy

It determines the fraction of proper classifications over the total number of data analyzed.

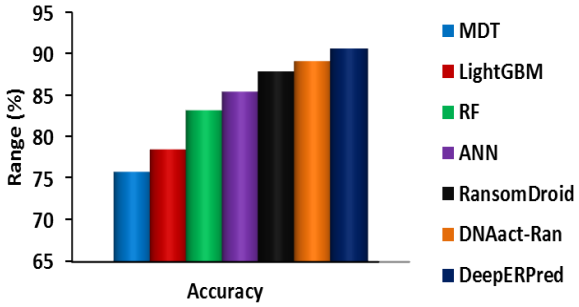

Figure. 4 Comparison of accuracy

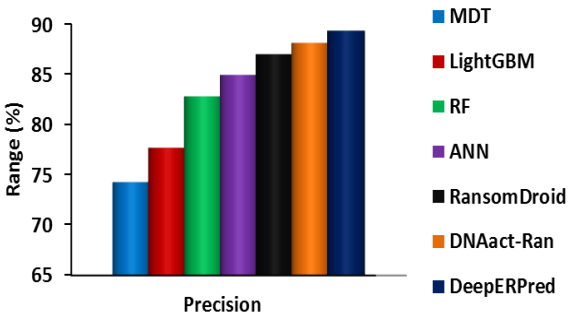

Figure. 5 Comparison of precision

$$Accuracy = \frac{True\ Positive\ (TP)+True\ Negative\ (TN)}{TP+TN+False\ Positive\ (FP)+False\ Negative\ (FN)} \quad (7)$$

In Eq. (7), TP is the result that the classifier properly classifies the ransomware data as themselves, TN is the result that the classifier properly classifies the goodware data as themselves, FP is the result that the classifier improperly classifies the ransomware data as goodware and FN is the result that the classifier improperly classifies the goodware data as ransomware.

Fig. 4 depicts the accuracy (in %) achieved by different classifiers to identify and categorize ransomware families. It indicates that the accuracy of DeepERPred is 19.57% greater than the MDT, 15.47% greater than the LightGBM, 8.95% greater than the RF, 6.07% greater than the ANN, 3.14% greater than the RansomDroid and 1.72% greater than the DNAact-Ran models. This is because of learning the relationship between present and previous occurrences of a string using an ensemble deep learner-based classifier appropriately. Thus, the DeepERPred model increases the accuracy of identifying and classifying ransomware tags.

4.2 Precision

It determines the ransomware data which are properly classified from the total classified data in a ransomware label.

$$Precision = \frac{TP}{TP+FP} \quad (8)$$

Fig. 5 portrays the precision (in %) of various classification models for ransomware identification and categorization. It analyzes that the precision of DeepERPred is 20.3% higher than the MDT, 15.02% higher than the LightGBM, 7.91% higher than the RF, 5.2% higher than the ANN, 2.7% higher than the RandomDroid and 1.4% higher than the DNAact-Ran models. This is due to the learning of temporal changes in DNA sequences effectively by the DeepERPred model. This realizes that the precision of DeepERPred is better than all other classification models for ransomware identification.

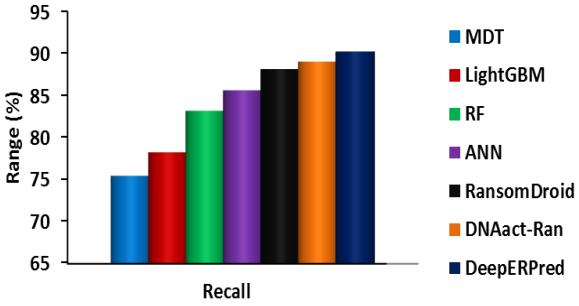

Figure. 6 Comparison of recall

4.3 Recall

It determines the percentage of ransomware data that is properly classified.

$$Recall = \frac{TP}{TP+FN} \quad (9)$$

Fig. 6 displays the recall (in %) of classical and presented classifiers to identify and categorize ransomware classes. It observes that the recall of DeepERPred is 19.62% larger than the MDT 15.32% larger than the LightGBM, 8.5% larger than the RF, 5.38% larger than the ANN, 2.4% larger than the RansomDroid and 1.37% larger than the DNAact-Ran models. This realizes that the recall of DeepERPred is superior to all other models because of training temporal changes in sequences effectively to identify and categorize ransomware classes.

4.4 F-measure

It is the harmonic average between precision and recall.

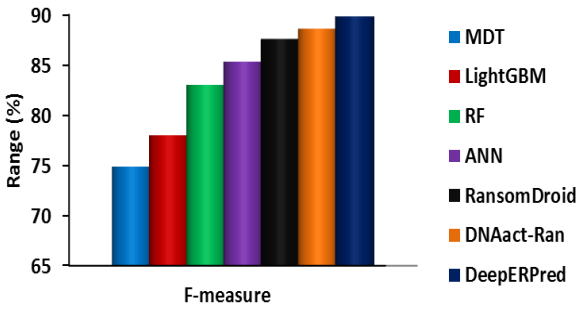

Figure. 7 Comparison of f-measure

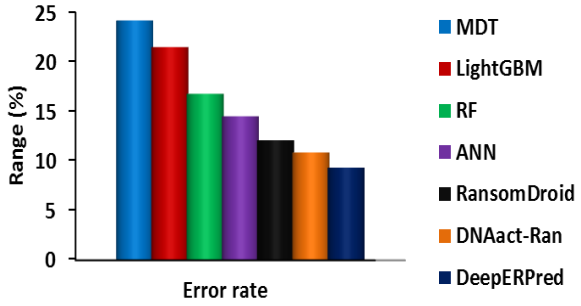

Figure. 8 Comparison of error rate

$$F-measure = \frac{2\times Precision\times Recall}{Precision+Recall} \quad (10)$$

Fig. 7 depicts the f-measure (in %) of different classifiers to identify and categorize ransomware families. It indicates that the f-measure of DeepERPred is 19.96% greater than the MDT, 15.17% greater than the LightGBM, 8.21% greater than the RF, 5.3% greater than the ANN, 2.55% greater than the RansomDroid and 1.38% greater than the DNAact-Ran models. This concludes that the DeepERPred model can increase the f-measure of identifying and categorizing ransomware classes by training the relationship between current and past events of sequences appropriately.

4.5 Error rate

It determines the fraction of improper classifications over the total number of data analyzed.

$$Error\ rate = \frac{FP+FN}{TP+TN+FP+FN} \quad (11)$$

Fig. 8 illustrates the error rate (in %) of different classifiers to identify and categorize ransomware families. It indicates that the error rate of DeepERPred is 61.4% less than the MDT, 56.56% less than the LightGBM, 44.4% less than the RF, 35.74% less than the ANN, 22.83% less than the RansomDroid and 14.1% less than the DNAact-Ran models. This summarizes that the DeepERPred model can minimize the error rate for ransomware identification compared to all other classification models.

Thus, these analyses realize that the DeepERPred model can achieve greater efficiency for identifying and categorizing ransomware families. This is because of learning temporal characteristics from the digital DNA sequences appropriately, whereas the other existing algorithms did not learn the temporal features from the time-varying digital DNA sequences for ransomware identification and categorization.

5. Conclusion

In this paper, the DeepERPred model was developed for identifying and categorizing ransomware families by learning the temporally changed DNA sequences. At first, a raw ransomware database was acquired and converted into the necessary form. Then, the MOGWO and BCS were applied to choose the most important attributes which create the new training database. This database was learned by the ensemble CNN-LSTM model to train the dependencies among present and previous occurrences of a string which helps to categorize ransomware data. At last, the testing results proved that the DeepERPred model has a 90.67% accuracy and 9.33% error rate compared to RansomDroid, LightGBM, DNAact-Ran, MDT and standard machine learning algorithm ANN ransomware classification models.

On the other hand, the creation of more learning data instances was highly influenced by manual annotation, which takes a longer time. So, future work will focus on adopting an automated data augmentation method to create more digital DNA sequences for ransomware classification.

Conflict of interest

The authors declare no conflict of interest.

Author contributions

Conceptualization, methodology, software, validation, Yuvaraj; formal analysis, investigation, Rober; resources, datacuration, writing—original draft preparation, Yuvaraj; writing—review and editing, Yuvaraj; visualization, supervision, Robert.

References

[1] A. Souri and R. Hosseini, “A State-of-the-art Survey of Malware Detection Approaches

using Data Mining Techniques", *Human-centric Computing and Information Sciences*, Vol. 8, No. 1, pp. 1-22, 2018.

[2] J. Singh and J. Singh, "A Survey on Machine Learning-based Malware Detection in Executable Files", *Journal of Systems Architecture*, Vol. 112, p. 101861, 2021.

[3] S. I. Popoola, S. O. Ojewande, F. O. Sweetwilliams, S. N. John, and A. Atayero, "Ransomware: Current Trend, Challenges, and Research Directions", In: *Proc. of the World Congress on Engineering and Computer Science*, Vol. 1, pp. 169-174, 2017.

[4] C. Beaman, A. Barkworth, T. D. Akande, S. Hakak, and M. K. Khan, "Ransomware: Recent Advances, Analysis, Challenges and Future Research Direction", *Computers & Security*, Vol. 111, p. 102490, 2021.

[5] S. Homayoun, A. Dehghantanha, M. Ahmadzadeh, S. Hashemi, and R. Khayami, "Know Abnormal, Find Evil: Frequent Pattern Mining for Ransomware Threat Hunting and Intelligence", *IEEE Transactions on Emerging Topics in Computing*, Vol. 8, No. 2, pp. 341-351, 2017.

[6] P. Singh, S. Tapaswi, and S. Gupta, "Malware Detection in PDF and Office Documents: Survey", *Information Security Journal: A Global Perspective*, Vol. 29, No. 3, pp. 134-153, 2020.

[7] A. Corum, D. Jenkins, and J. Zheng, "Robust PDF Malware Detection with Image Visualization and Processing Techniques", In: *IEEE 2nd International Conf. on Data Intelligence and Security*, pp. 108-114, 2019.

[8] G. DAngelo, M. Ficco, and F. Palmieri, "Malware Detection in Mobile Environments based on Autoencoders and API-images", *Journal of Parallel and Distributed Computing*, Vol. 137, pp. 26-33, 2020.

[9] T. Honda, K. Mukaiyama, T. Shirai, T. Ohki, and M. Nishigaki, "Ransomware Detection considering User's Document Editing", In: *Proc. of IEEE 32nd International Conf. on Advanced Information Networking and Applications*, pp. 907-914, 2018.

[10] O. M. Alhawi, J. Baldwin, and A. Dehghantanha, "Leveraging Machine Learning Techniques for Windows Ransomware Network Traffic Detection", *Cyber Threat Intelligence, Springer, Cham*, Vol. 70, pp. 93-10, 2018.

[11] J. Lim and J. H. Yi, "Structural Analysis of Packing Schemes for Extracting Hidden Codes in Mobile Malware", *EURASIP Journal on Wireless Communications and Networking*, pp. 1-12, 2016.

[12] F. Khan, C. Ncube, L. K. Ramasamy, S. Kadry, and Y. Nam, "A Digital DNA Sequencing Engine for Ransomware Detection using Machine Learning", *IEEE Access*, Vol. 8, pp. 119710-119719, 2020.

[13] B. M. Khammas, "Ransomware Detection using Random Forest Technique", *ICT Express*, Vol. 6, No. 4, pp. 325-331, 2020.

[14] G. Ramesh and A. Menen, "Automated Dynamic Approach for Detecting Ransomware using Finite-State Machine", *Decision Support Systems*, Vol. 138, pp. 1-10, 2020.

[15] Y. A. Ahmed, B. Koçer, S. Huda, B. A. S. A. Rimy, and M. M. Hassan, "A System Call Refinement-based Enhanced Minimum Redundancy Maximum Relevance Method for Ransomware Early Detection", *Journal of Network and Computer Applications*, Vol. 167, pp. 1-18, 2020.

[16] F. Ullah, Q. Javaid, A. Salam, M. Ahmad, N. Sarwar, D. Shah, and M. Abrar, "Modified Decision Tree Technique for Ransomware Detection at Runtime through API Calls", *Scientific Programming*, Vol. 2020, pp. 1-10, 2020.

[17] J. Hwang, J. Kim, S. Lee, and K. Kim, "Two-stage Ransomware Detection using Dynamic Analysis and Machine Learning Techniques", *Wireless Personal Communications*, Vol. 112, No. 2, pp. 2597-2609, 2020.

[18] S. Sharma, C. R. Krishna, and R. Kumar, "RansomDroid: Forensic Analysis and Detection of Android Ransomware using Unsupervised Machine Learning Technique", *Forensic Science International: Digital Investigation*, Vol. 37, pp. 1-11, 2021.

[19] S. I. Imtiaz, S. U. Rehman, A. R. Javed, Z. Jalil, X. Liu, and W. S. Alnumay, "DeepAMD: Detection and Identification of Android Malware using High-efficient Deep Artificial Neural Network", *Future Generation Computer Systems*, Vol. 115, pp. 844-856, 2021.

[20] H.Gunduz. "Malware detection framework based on graph variational autoencoder extracted embeddings from API-call graphs", *PeerJ Computer Science*, Vol. 8, p. e988, 2022.