Journal of Information Systems Engineering and Management
2025, 10(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

A Novel IChOA–CNN-LSTM Model for Android Malware Detection Using Opcode-Based Feature Selection and Optimization

Renuga S[1, *], Jose Reena K[2] and Kebila Anns Subi L[3]

[1]Department of Artificial Intelligence and Data Science, Panimalar Engineering College, Poonamallee, Chennai, Tamilnadu, India.

[2]Department of Computer Applications, B. S. Abdur Rahman Crescent Institute of Science and Technology, Chennai, Tamilnadu, India.

[3]Department of Electronics and Communication Engineering, Ponjesly College of Engineering, Nagercoil, Tamilnadu, India.

ARTICLE INFO

Received: 16 Nov 2024

Revised: 28 Dec 2024

Accepted: 20 Jan 2025

ABSTRACT

The rapid growth of Android applications has resulted in an increase in security concerns, specifically malware attacks. Traditional signature-based and heuristic detection technologies fail to keep up with the changing nature of malware. To address this issue, deep learning-based algorithms provide a viable solution by utilizing sophisticated feature extraction and classification techniques to improve detection accuracy. This paper presented an Improved Chimp Optimization Algorithm-based CNN-LSTM (IChOA-CNN-LSTM) technique for detecting Android malware. The procedure starts with an Android malware dataset, which goes through data pre-processing to clean and modify it for best analysis. To extract significant characteristics, a feature selection procedure is used in conjunction with an opcode-based model. Furthermore, the IChOA-CNN-LSTM technique uses the IChOA-CNN-LSTM methodology to improve malware detection. The model improves feature selection by combining an enhanced transformer with an RNN model and a softmax function for better classification. Finally, the Snake Optimizer Algorithm (SOA) is used to fine-tune parameters for the best detection performance. Extensive experimental findings show that the IChOA-CNN-LSTM technique is successful for detecting Android malware. The system's performance is measured using key measures like accuracy, precision, recall, and F-score in addition to a strong malware detection architecture.

Keywords: Android malware detection, Deep learning, CNN-LSTM, Improved Chimp Optimization Algorithm (IChOA), Opcode-based model, SOA model

Introduction

In recent years, practically every member of society has developed a daily need for the Internet. This is due to the fact that practically everything becomes difficult without the Internet, including marketing, social media, online banking, and health-related tasks. Because of how quickly the Internet has expanded, criminals are now committing crimes online rather than in person. Malicious software is commonly used by thieves to initiate cyberattacks on target machines. Computers, smartphones, computer networks, and other devices that intentionally install dangerous payloads are referred to as malware. Trojan horses, worms, ransomware, rootkits, and viruses are just a few of the many distinct types of malware. Malware was first developed with minimal objectives in mind, which made detection easier. Traditional (basic) malware is the term used to describe this type of malware. Malware that is more harmful and harder to identify than older malware is known as "next-generation" malware.

Journal of Information Systems Engineering and Management
2025, 10(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Malware that is capable of running in kernel mode is now categorized as such. When running in kernel mode, this kind of malware can readily bypass firewalls, antivirus software, and other security measures. In the digital era, security breaches brought on by malicious software (malware) attacks are becoming more frequent, which is a serious security risk. Current methods for detecting malware are based on the static and dynamic analysis of malware signatures and behavior patterns, which is time-consuming and has not been proven to be effective in real-time detection of new infections [1]. Android malware infiltrates and functions on cellphones without the users' awareness or approval. It poses serious risks to consumers, including advanced fraud and the loss of personal data. Scholars and professionals have offered a number of strategies to counter these dangers [2]. Using association rule analysis, one can uncover intriguing connections between data sets, such as benign and malicious [3]. The GWOANL-SMDC detection procedure is used for software malware detection and classification. First, for feature selection, the GWOANL-SMDC technique uses the NCM model [4]. According to performance validation, the DLBITM-AMD approach outperformed current Android malware recognition models with an accuracy value of 97.26% [5].

Contribution of this Work

The following are the key phases of the suggested approach, for further information:

1. Proposed an IChOA based CNN-LSTM approach for enhanced Android malware detection.
2. Utilizes an opcode-based model for effective feature selection, improving classification performance.
3. Enhances feature extraction by combining an improved transformer with an RNN model and a softmax function.
4. Fine-tunes model parameters using SOA to achieve better detection accuracy. Validates the model using key metrics include accuracy, precision, recall, and F-score to demonstrate its effectiveness.

The following is describes the way the paper is established: Section 1 illustrates the introduction. Section 2 describes the relevant works. The proposed approaches are described in Section 3, the experiment's results are displayed in Section 4, and a conclusion and recommendations for further research are made in Section 5.

1. Related Works

Majid et al. [6] have proposed, the program that damages files and information systems is known as malware. Different strategies have been employed by adversaries and cybercriminals to disseminate malware for financial gain as well as other objectives. Attackers derive economic gains from such attacks. Kilic et al. [7] have introduced, a possible to download and install apps on Android devices outside of the official application store. Users' security and privacy are at risk when third-party environments are used to install applications. Hein et al. [8] have recommend, a symantec and F-Secure databases were eventually rejected due to the impracticability of autonomously gathering data from them. Furthermore, because the researchers created the technical specifications by hand, there were discrepancies in the information regarding whether or not they included the rights that the malware required.

Mallidi et al. [9] have reported, the dataset that was used contained 25458 samples with 173 characteristics (8643 malware applications and 16815 benign apps). Three datasets were created using the proposed feature selection process; each had 25458 samples and 5, 10, and 20 features. Abubaker et al. [10] have described, an accuracy of 93.73%, the BOAWFS-DT outperformed after lowering the features from 4115 to 518 utilizing 200 agents and 100 iterations. One significant addition of BOAWFS is that it improves accuracy by 1.67% while reducing feature redundancy by 87.41% for the permission-based Android malware dataset with extremely high dimensions. Xing et al. [11] have introduced, the primary objective of the feature data pre-processing step is to supply input data for the neural network model.

Journal of Information Systems Engineering and Management
2025, 10(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Alamro rt al. [12] have proposed, to achieve this, data preprocessing is done at the preliminary stage via the AAMD-OELAC technique in order to accomplish this. The AAMD-OELAC technique detects malware on Android smartphones through an ensemble learning process. Aamir et al. [13] have proposed, the deep learning technique known as the AMDD Lmodel is based on a convolutional neural network. This model detects and classifies Android malware using a variety of parameters, filter sizes, epoch counts, learning rates, and layers. Baink et al. [14] have recommend, from the disassembled APK, the permission functionality can be extracted by Android application package. Then used the Frequent Pattern (FP) Growth method to find the most prevalent and similar pairings of legitimate permissions. Kim et al. [15] have presented, use a variety of characteristics to provide a multifaceted explanation of Android app functionality. Existence-based feature extraction or similarity-based feature extraction is used to enhance features for efficient feature representation in malware detection.

Iadarola et al. [16] have suggested, to capture local spatial correlations, the system combines CNNs and LSTM neurons, uses NLP techniques as a baseline, and learns from successive long-term dependencies. Mahindru et al. [17] have introduced, to employ five distinct feature selection strategies to empirically test 1,20,000 Android apps. Principal Component Analysis (PCA) developed a collection of features that allow for the detection of 94% of Android malware from physical apps. Bayazit et al. [18] have proposed, with an accuracy rate of 98.85%, experimental data demonstrate that the BiLSTM model works better than earlier RNN-based deep learning algorithm modes. Wasif et al. [19] have presented, the model displays its efficacy in accurately differentiating between harmful and benign applications with a multiclass accuracy rate of 95.61%, which is revolutionary [19].

2. Proposed system

A thorough process for detecting Android malware utilizing an IChOA–CNN-LSTM method is shown in Figure 1. An Android malware dataset, which includes both harmful and benign apps, is fed into the procedure at the start. The pre-processing procedure is used to clean and convert the raw data for better analysis, ensuring data quality. An opcode-based approach is used to pick features after pre-processing, extracting pertinent data to aid in distinguishing between malicious and trustworthy apps. The IChOA–CNN-LSTM model is used to detect Android malware after the important features have been chosen. At this point, CNNs are employed to extract features. In the meantime, LSTM controls the sequential dependencies in the data to increase the accuracy of classification. In order to improve the model's performance, the features are further selected. By fine-tuning crucial parameters, the SOA is utilized for hyperparameter optimization, which increases detection efficiency. Finally, some key measures that are utilized to evaluate the effectiveness of the proposed approach include accuracy, precision, recall, and F-score. These precautions guarantee a solid and efficient malware detection system that can accurately identify harmful Android apps.

Journal of Information Systems Engineering and Management
2025, 10(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

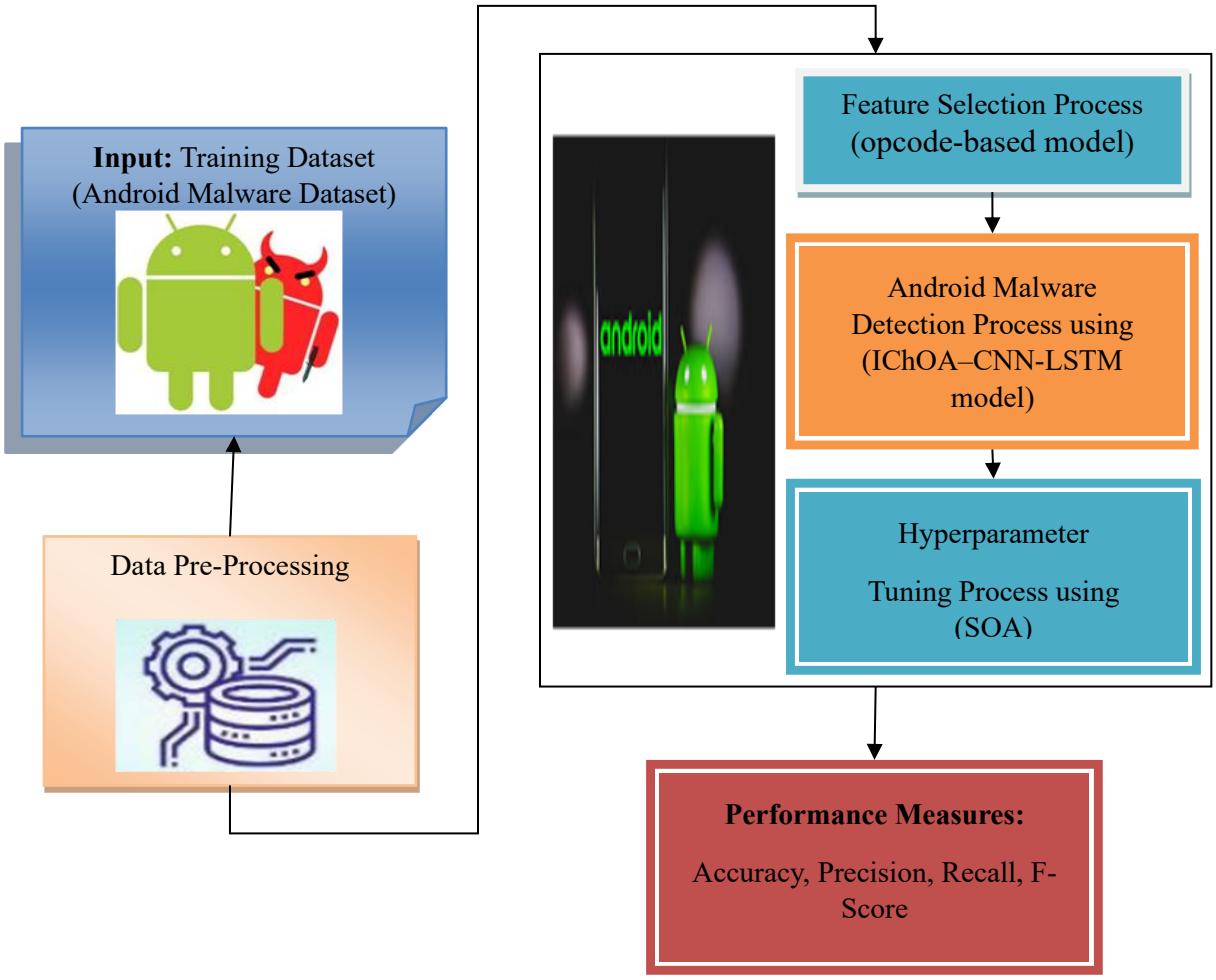

Figure 1. The IChOA–CNN-LSTM algorithm's overall flow for detecting Android malware

2.1. Data pre-processing

Cleaning is the initial stage in data preprocessing, which can enhance the dataset's quality. Duplicate values are eliminated and missing values are handled in this stage. The dataset utilized for training and testing included a variety of ransomware and non-ransomware samples, each of which was meticulously tagged to capture unique behavioral characteristics seen in modern settings. Techniques for feature scaling and normalization made ensuring that samples were consistent, which helped to maintain input representation consistency across the model's architecture. The application of dimensionality reduction, principal component analysis (PCA) in particular, to condense high-dimensional data greatly improved computational efficiency without causing a major loss of information.

The integrity of the classification results was maintained by using data balancing strategies including oversampling and undersampling to handle class imbalances between malware and benign samples. Extreme behavioral aberrations, or outliers, were purposefully kept in the dataset to improve the model's ability to detect unusual malware activity. By combining and converting indications into composite measures that fully captured ransomware behavioral patterns, feature engineering significantly improved the dataset. The actual implementation of DeepCodeLock in settings with high

Journal of Information Systems Engineering and Management
2025, 10(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

data throughput was made easier by this preprocessing approach, which significantly improved model accuracy.

2.2. Feature Selection in Opcode-based model

Feature selection methods were used to lower the dimensionality and improve classification performance due to the high dimensionality of opcode-based feature vectors. The ability of each attribute to differentiate between malware and benign samples was ranked using the Chi-Square test. Features with low Chi-Square scores that made minimal contributions to the classification job were eliminated, whereas those with high values were kept. Figure 2 shows this procedure, which reduced the model's complexity and increased its efficiency without sacrificing detection accuracy by filtering characteristics according to statistical significance and eliminating those that were unnecessary.

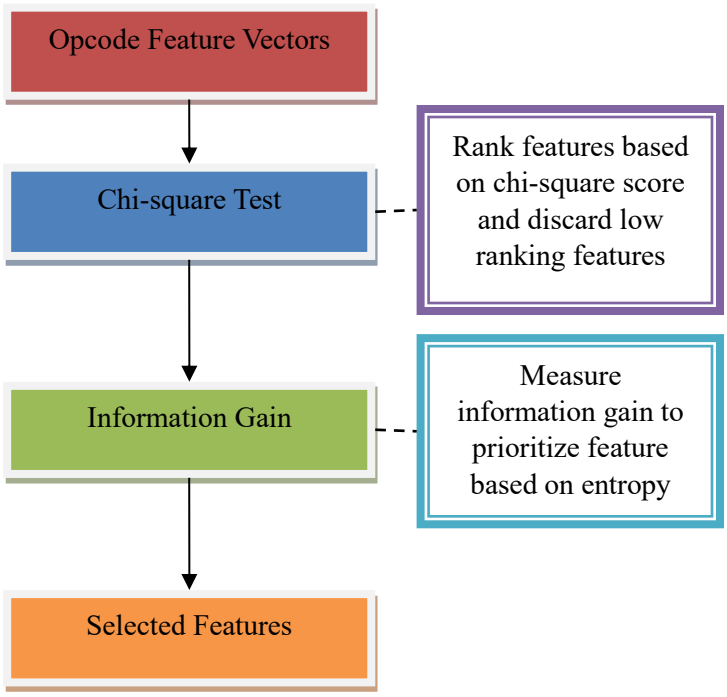

Figure 2. Feature selection process for reducing dimensionality and improving detection accuracy

To evaluate the entropy reduction attained by individual features, Information Gain, another feature selection technique, was applied. The model was made more interpretable and generalizable by giving priority to features that helped reduce the uncertainty in malware classification. This stage balanced classification accuracy and processing economy by ensuring that only the most pertinent features were kept. Furthermore, feature selection ensured that the model retained generality across unknown data by lowering the chance of overfitting. Both the overall efficacy of predictions and the speed of model training were significantly enhanced by the chosen features.

2.3. Proposed Android Malware Detection Process using IChOA–CNN-LSTM algorithm

Feature selection is followed by the detection of the extracted features 〖CD〗_n^fea using the IChOA–CNN-LSTM approach. Long Short-Term Memory (LSTM) and Convolution Neural Networks (CNN) are combined to form CNN-LSTM. Predicting the expected output is challenging since the CNN-LSTM model's weight values are initialized randomly, increasing the loss function. To improve the weight values produced in the convolution layer and increase the model's classification accuracy, the CNN-LSTM and IChOA hybridize. The present ChOA enhances the chimpanzees' foraging approach by utilizing the Basin chaotic map. In Figure 3, the proposed detection system's construction is displayed.

Journal of Information Systems Engineering and Management
2025, 10(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

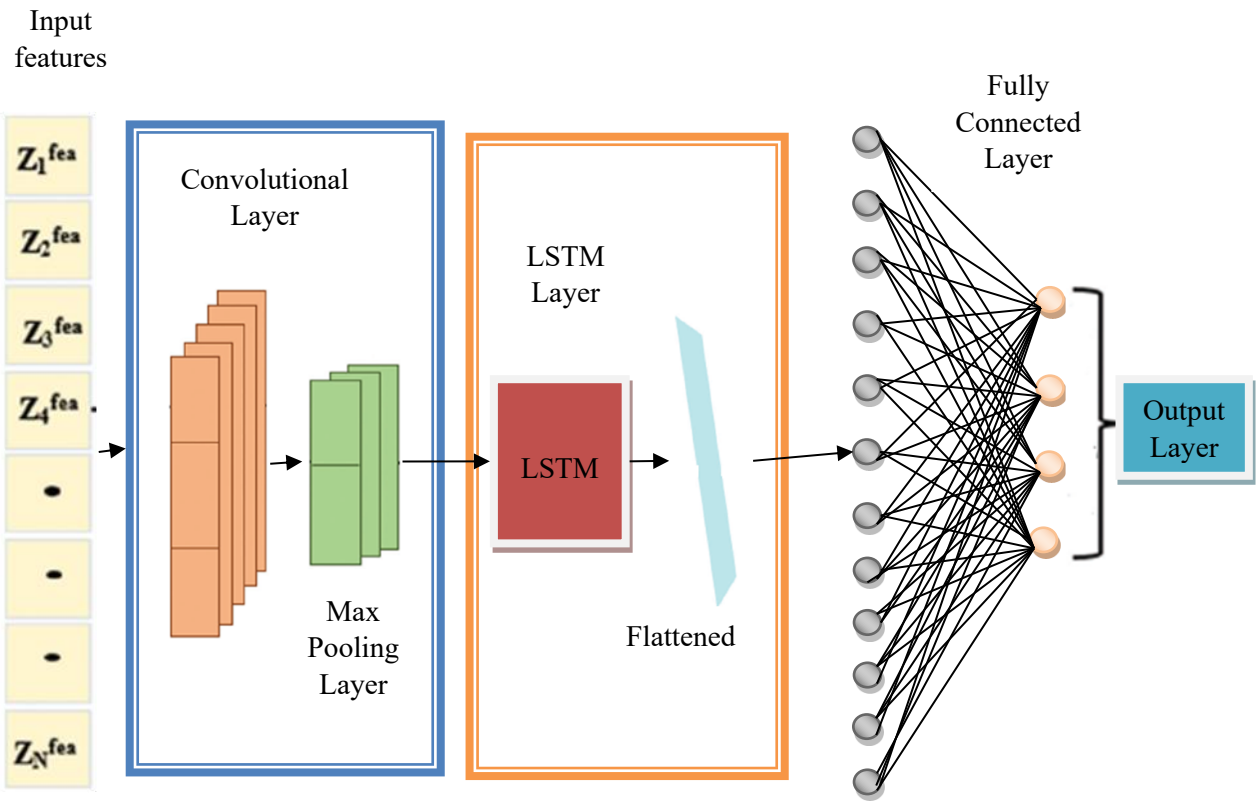

Figure 3. Establishment of IChOA-CNN-LSTM model

Input Layer

An artificial neural network's initial layer of nodes is referred to as the input layer. Input data from the external environment is received by this layer.

Convolution Layer

The following is the expression for the convolution between the input features and the kernels [20] of the convolution layer:

$$C_l(f_{m(i)}) = CD_n^{fea} * h_n(\varphi(f_{m(i)})) \quad (1)$$

Where, $C_l(f_{m(i)})$ is represents the feature map that was found using kernels φ, and h_n corresponds to the activation function that is not linear.

Pooling Layer:

Following convolution, pooling is the following stage, which eliminates superfluous features from the features map to reduce its size. This is accomplished by the network using the max-pooling function which is represented as,

$$P_l(f_{m(i)}) = \zeta_{mP_n}(C_l(f_{m(i)})) \quad (2)$$

Where, $P_l(f_{m(i)})$ indicates that the feature map has been pooled, and ζ_{mP_n} indicates that the max-pooling function was employed to make the feature map less dimensional. The behavior of each memory cell is then altered by feeding the pooled feature map $C_l(f_{m(i)})$ into the various LSTM gates.

Journal of Information Systems Engineering and Management
2025, 10(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ Research Article

Fully Connected Layer

At each layer, the entirely linked layer is found using the formula below:

$$f_{cl} = h_n(\Omega_{fc} O_g(m) + v_{fc}) \quad (3)$$

Where, h_n represents the activation function, the bias value is v_{fc}, and the completely connected layer, the weight matrix is Ω_{fc}.

Output layer

A neural network's output layer, which generates the network's classifications or predictions, is its last layer. It converts the data into a format that can be used after receiving it from the earlier levels.

2.3.1. Improved Chimp Optimization Algorithm (IChOA)

The ChOA was developed with inspiration from chimpanzees' foraging habits. The chimp is a mammal that is most similar to the human. They are boisterous, gregarious, inquisitive, and intelligent. A variety of groups, each with a certain number of members, make up the community in which chimpanzees reside. By employing the Basin chaotic map, the current ChOA improves the slow convergence rate and high entrapment in local optima in the chimps' hunting process. Each tribe has varying levels of hunting ability. The methodology begins with the initial chimp population and yields arbitrary solutions like,

$$\varphi_n = \{\varphi_1, \varphi_2, \varphi_3, \dots, \varphi_N\} \quad (4)$$

The coefficient vectors are calculated as,

$$\zeta = \alpha(2\gamma_1 - 1) \quad (5)$$

$$\varepsilon = 2\gamma_2 \quad (6)$$

$$\delta = Cht_{vec}^{B} \quad (7)$$

Where, α declines nonlinearly with each repetition throughout the exploration and exploitation stages. Cht_{vec}^{B}, while the random vectors $\tau, \gamma 1, \gamma 2$ are extracted from the intervals [0,1].

Chimpanzees have the ability to switch to chaotic behavior in order to obtain social rewards during the last stages of hunting. Chimp's location can be updated throughout this phase by switching between normal and chaotic behavior. The update for the position can be expressed as:

$$\varphi_{ch}(\tau + 1) = \begin{cases} \varphi_{pr}(\tau + 1) - r.s & if(\varphi < 0.5) \\ Cht_{vec}^{B} & if(\varphi > 0.5) \end{cases} \quad (8)$$

Where, f ∈ (0, 1) is a representation of the random vector used to calculate the likelihood of selecting the update behavior. When an attack is nearing its conclusion, chimpanzees use chaotic maps to tackle sluggish convergence rate issues and local optima. This generates the optimal weight values. Algorithm 1, which describes the fundamental steps in the procedure, is shown below together with the proposed IChOA–CNN-LSTM pseudo code. The proposed IChOA-CNN-LSTM employs a number of classification layers and optimizes the convolution layer's weight values based on the loss function using the IChOA technique.

Input: Selected Futures CD_n^{fea} Output: Detected output
Begin Set up the convolution, pooling, fully linked, and input feature layers, and weight value φ_n, and loss

Journal of Information Systems Engineering and Management
2025, 10(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ **Research Article**

Compute convolution layer $C_l(f_{m(i)})$

Compute pooling layer $P_l(f_{m(i)})$

Compute LSTM layer $f_g(m), I_g(m), C_s(m), O_g(m)$

Compute fully connected layer fc_l

Check loss Function

If $(loss > thr)$

Initialize population φ_n, coefficient vector $\varepsilon, \zeta, \delta$ s, maximum number of iteration τ_{max}

Calculate fitness for each chimp

Set $\tau = 0$

While $(\tau \leq \tau_{max})$

Update $\varepsilon, \zeta, \delta$

Update position using $P_{(A,C,B,D)}$

Evaluate fitness of the position of chimps

If $\varphi < 0.5 \&\& \varepsilon \in (0,1)$\{

Update position of the Chimp using $\varphi_{pr}(\tau + 1) - r.s$

\}

Else\{

Update position of the Chimp using Cht_{vec}^{B}

\}End if

Update $\varepsilon, \zeta, \delta$

Calculate fitness of the current position of the Chimp

Ser $\tau = \tau + 1$

End while

Return optimized weights

Else

Indicate that the output is the final result.

End if

End

Algorithm 1. A pseudocode for the proposed IChOA-CNN-LSTM technique

2.4. SOA-based parameter tuning

For the classification method37, the optimal parameter is finally chosen using the SOA methodology. Because the SOA technique is effective at exploring and adapting to high-dimensional spaces, it is used to optimize classification parameters. By effectively exploring intrinsic parameter

Journal of Information Systems Engineering and Management
2025, 10(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

landscapes and avoiding local minima, the SOA model dynamically searches for optimal solutions by mimicking snake behavior, in contrast to traditional approaches. This adaptive strategy enhances classification accuracy by selecting robust and precise parameters.

Initialization

The optimizer issue solution is related to the snake's placement in SOA; the grade of the solution is indicated by the modest function values, and each snake section is selected at random. An arbitrary population is produced by the SOA, which is similar to heuristic methods. The formulation is as follows:

$$X_i = X_{min} + r \times (X_{max} - X_{min}) \quad (9)$$

Where, X_i shows where the snake is, r is a randomly generated number, and X_{max} and X_{min} are the top and lower limits, respectively.

Grouping of snakes

Snake N's population has been divided into two groups: males and females. Both sets' mathematical formulations are provided below:

$$N_m = \frac{N}{2} \quad (10)$$

$$N_f = N - N_m \quad (11)$$

Where, N represents all of the different snakes, whereas N_m and N_f stand for the number of male and female snakes, respectively.

The effectiveness of SOA is significantly impacted by the FF. The SOA states that the main requirement for creating the FF in this study is precision, which is explained below.

$$Fitness = \max(P) \quad (12)$$

$$P = \frac{TP}{TP+FP} \quad (13)$$

Here, TP shows the actual positive value, while FP represents the false positive value.

2.5. Performance evaluation

Accuracy is defined as the proportion of accurately detected instances within the total number of instances.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (14)$$

The recall calculates the percentage of affirmative cases that are accurately classified.

$$Recall = \frac{TP}{TP+FN} \quad (15)$$

Out of all occurrences that are expected to be positive, the accuracy metric measures the percentage of correctly predicted positive cases.

$$Precision = \frac{TP}{TP+FP} \quad (16)$$

The F1-score is an estimate that takes the harmonic mean of precision and recall into account.

$$F1\_score = \frac{2TP}{2TP+FP+FN} \quad (17)$$

Experimental result and discussion

In this part, the IChOA-CNN-LSTM approach's Android malware recognition findings are validated using the 7500-case database listed in Table 1. The detection results are analyzed using a set of measures called recall, accuracy, precision, and F-score.

Journal of Information Systems Engineering and Management
2025, 10(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Table 1. Details on database

Classes	No.of instantances
Benign	5000
Malware	2500
Total Instances	7500

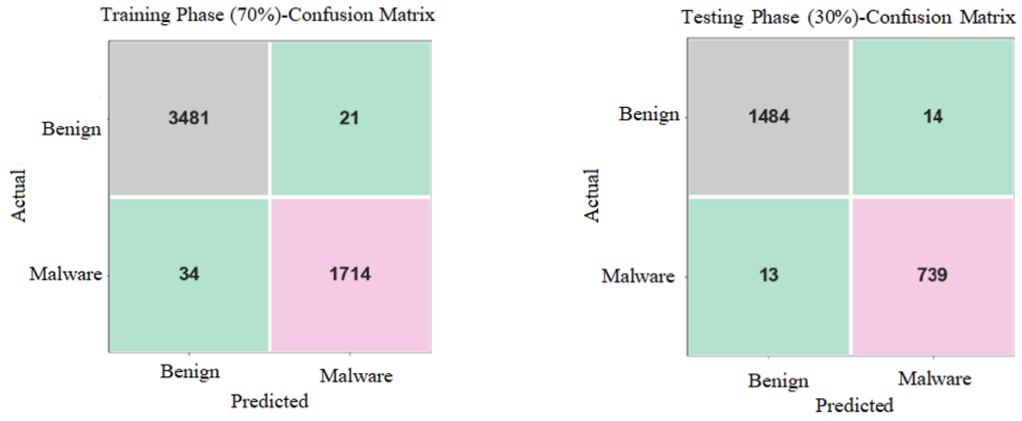

Figure 4. Confusion matrix for IChOA-CNN-LSTM model-based malware detection

The confusion matrices generated by the IChOA-CNN-LSTM model are shown in Figure 4 at a ratio of 70:30 for the training phase (TRAP) to the testing phase. The findings demonstrate that IChOA-CNN-LSTM is capable of efficiently identifying both benign and malicious samples across all classes.

Figure 5 illustrates the TRA and TES accuracy curves, which are used to measure the performance of the IChOA-CNN-LSTM approach of TRAP/TESP. The IChOA-CNN-LSTM system's solution is displayed across a large number of epochs via the TRA and TES accuracy curves. The figure provides important information about the ICHOA-CNN-LSTM model's learning tasks and general capabilities. As the epoch number increased, so did the TRA and TES accuracy curves. It has been demonstrated that the IChOA-CNN-LSTM methodology improves testing accuracy, allowing for the classification of designs in TRA and TES data.

Figure 5. Accuracy curve of IChOA-CNN-LSTM methodology

Journal of Information Systems Engineering and Management
2025, 10(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

The overall TRA and TES loss values for the IChOA-CNN-LSTM approach at the TRAP/TESP over epochs are shown in Figure 6. As the TRA loss decreases throughout epochs, so does the model loss. As the weight is changed by the algorithm to reduce the projected error, the TRA and TES data demonstrate a drop in loss values [21]. The degree to which the approach matches the TRA data is shown by the loss curves. The IChOA-CNN-LSTM model is successfully identifying the patterns in the TRA and TES data, as evidenced by the TRA and TES loss that progressively declines. Furthermore, it is evident that the IChOA-CNN-LSTM method adjusts the parameters to reduce the discrepancy between the original TRA labels and the forecast.

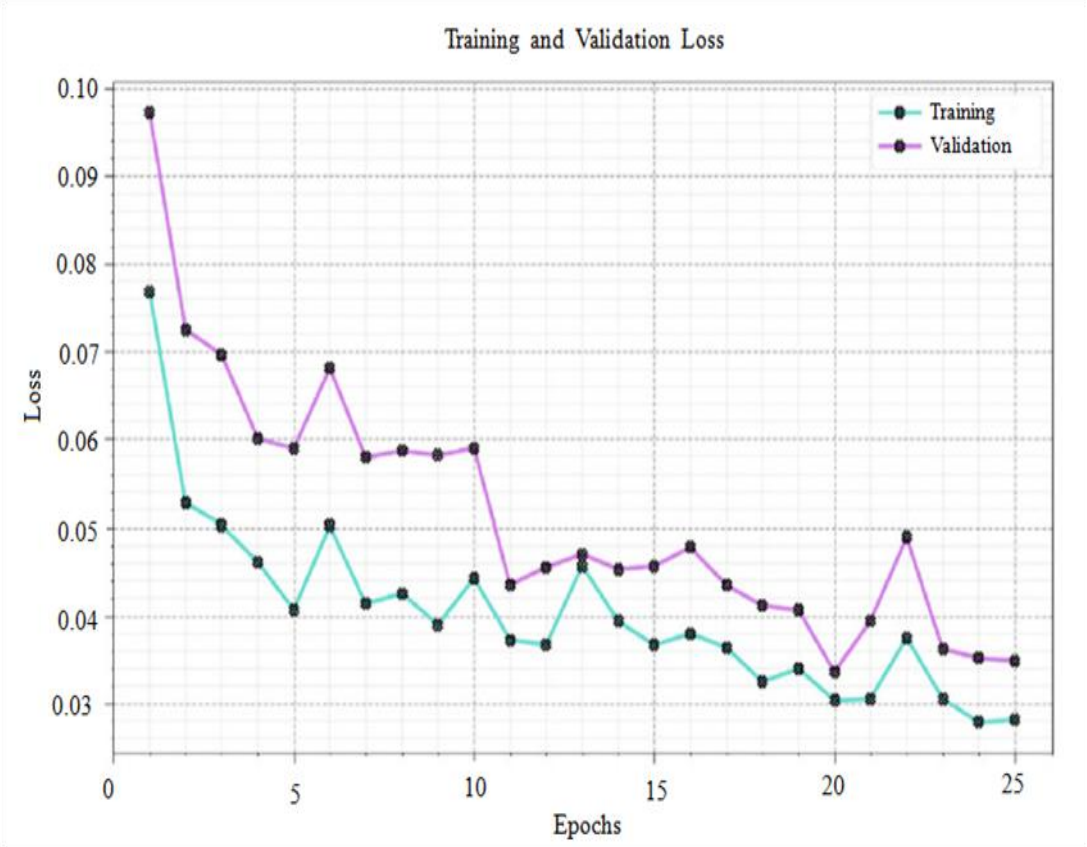

Figure 6. Loss curve of IChOA-CNN-LSTM methodology

Figure 7 shows the ROC curves produced by the IChOA-CNN-LSTM model, which is capable of classifying the data. The chart offers crucial information on the trade-off between TPR and FPR rates across various categorization criteria and epoch counts. It provides the precise prediction performance of the IChOA-CNN-LSTM method for identifying all two class labels.

Journal of Information Systems Engineering and Management
2025, 10(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

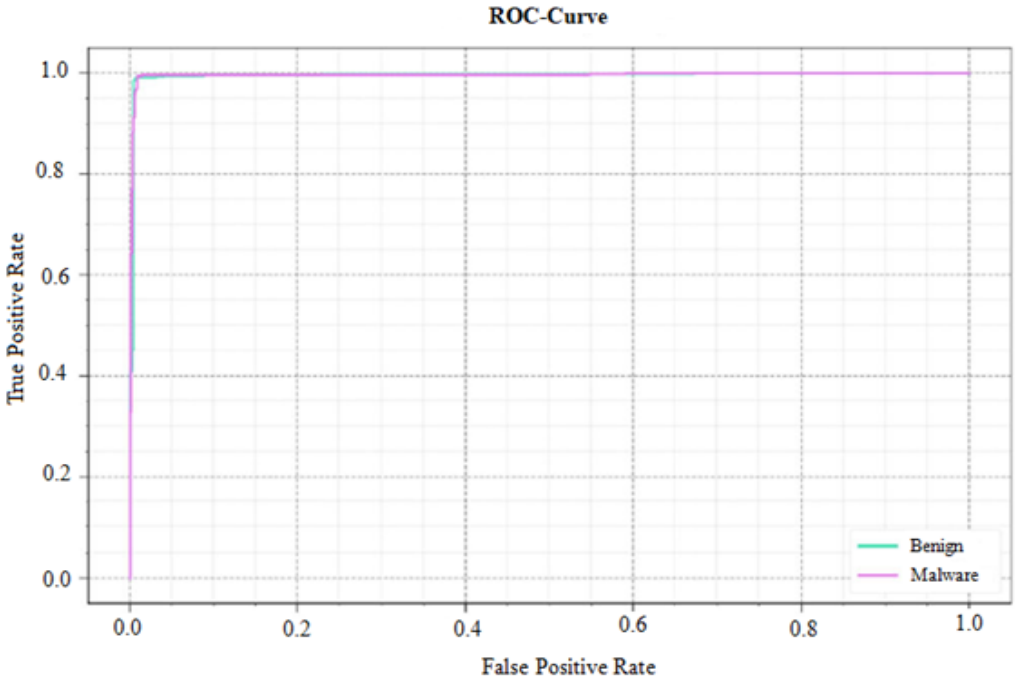

Figure 7. ROC curve of IChOA-CNN-LSTM methodology

Table 2. Comparative outcome of IChOA-CNN-LSTM methodology with exising systems

Algorithm	Accuracy	Precision	Recall	F1-score
AdaBoostMI [22]	88.40	81.70	91.93	94.25
MLP [23]	94.10	97.10	97.90	92.35
AAMD-OELAC [24]	96.93	97.05	89.93	95.04
IChOA-CNN-LSTM (Proposed)	**99.18**	**99.10**	**99.04**	**99.07**

A comparison of the proposed and existing models' performances is shown in Table 2 and Figure 8. Meanwhile, the IChOA-CNN-LSTM approach increases precision by 99.10%, whilst the MLP, AdaBoostMI, and AAMD-OELAC strategies reduce precision by 81.70%, 97.10%, and 99.05%, respectively. In terms of accuracy, the IChOA-CNN-LSTM method increases precision by 99.18%, whereas the MLP, AdaBoostMI, and AAMD-OELAC strategies reduce accuracy by 88.40%, 98.10%, and 98.93%, respectively.

Journal of Information Systems Engineering and Management
2025, 10(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

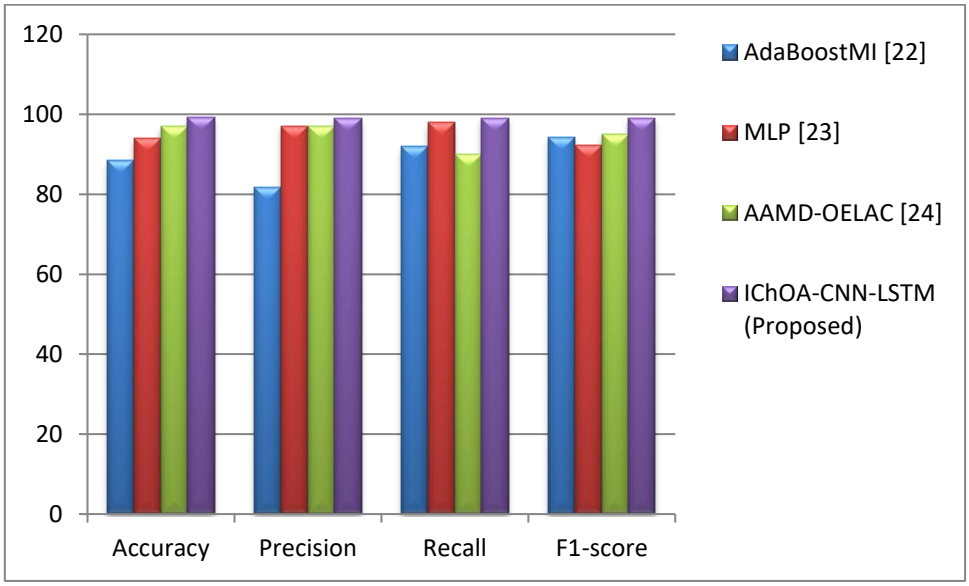

Figure 8. Performance comparison between the proposed and existing models

Based on the F1-score, the IChOA-CNN-LSTM approach achieves a higher F1-score of 99.07%, whereas the MLP, AdaBoostMI, and AAMD-OELAC strategies achieve lower accuracy values of 94.25%, 98.35%, and 99.04%. Finally, based on recall, the IPREODL technique delivers a greater value recall of 99.04%, whereas the MLP, AdaBoostMI, and AAMDOELAC models achieve lower values recall of 91.93%, 97.90%, and 98.93% respectively. Thus, the IChOA-CNN-LSTM approach outperforms other models.

Conclusion

The proposed IChOA–CNN-LSTM approach effectively enhances Android malware detection by integrating an ICHOA with a CNN-LSTM framework. The opcode-based feature selection and IChOA-CNN-LSTM technique improve the model's ability to extract relevant features, leading to higher detection accuracy. Additionally, the Snake Optimizer Algorithm (SOA) optimizes parameters for better classification performance. In terms of accuracy (99.18%), precision (99.10%), recall (99.04), and F-score (99.07), extensive experimental findings show that the proposed IChOA-CNN-LSTM approach performs better than conventional malware detection techniques, making it a reliable and effective way to counteract changing Android malware threats. In future incorporate larger and more diverse datasets to improve generalization and adaptability to emerging malware threats.

Reference

1. Vinayakumar, R., Mamoun Alazab, K. P. Soman, Prabaharan Poornachandran, and Sitalakshmi Venkatraman. "Robust intelligent malware detection using deep learning." IEEE access 7 (2019): 46717-46738.
2. Pan, Ya, Xiuting Ge, Chunrong Fang, and Yong Fan. "A systematic literature review of android malware detection using static analysis." IEEE Access 8 (2020): 116363-116379.
3. Zhang, Hanqing, Senlin Luo, Yifei Zhang, and Limin Pan. "An efficient Android malware detection system based on method-level behavioral semantic analysis." IEEE Access 7 (2019): 69246-69256.
4. Assiri, Mohammed. "Robust malicious software detection and classification using global whale optimization algorithm with deep learning approach." Scientific Reports 14, no. 1 (2024): 25383.
5. Almakayeel, Naif. "Deep learning-based improved transformer model on android malware detection and classification in internet of vehicles." Scientific Reports 14, no. 1 (2024): 25175.

6. Majid, Al-Ani Mustafa, Ahmed Jamal Alshaibi, Evgeny Kostyuchenko, and Alexander Shelupanov. "A review of artificial intelligence based malware detection using deep learning." Materials Today: Proceedings 80 (2023): 2678-2683.
7. Kılıç, Kazım, İsmail Atacak, and İbrahim Alper Doğru. "FABLDroid: Malware detection based on hybrid analysis with factor analysis and broad learning methods for android applications." Engineering Science and Technology, an International Journal 62 (2025): 101945.
8. Hein, C. L. P. M., and Khin Mar Myo. "Permission-based feature selection for android malware detection and analysis." International Journal of Computer Applications 181, no. 19 (2018): 29-39.
9. Mallidi, S. "Bat optimization algorithm for wrapper-based feature selection and performance improvement of android malware detection." IET Networks (Wiley-Blackwell) 10, no. 3 (2021).
10. Abubaker, Howida, Aida Ali, Siti Mariyam Shamsuddin, and Shafaatunnur Hassan. "Exploring permissions in android applications using ensemble-based extra tree feature selection." Indonesian Journal of Electrical Engineering and Computer Science 19, no. 1 (2020): 543-552.
11. Xing, Xiaofei, Xiang Jin, Haroon Elahi, Hai Jiang, and Guojun Wang. "A malware detection approach using autoencoder in deep learning." IEEE Access 10 (2022): 25696-25706.
12. Alamro, Hayam, Wafa Mtouaa, Sumayh Aljameel, Ahmed S. Salama, Manar Ahmed Hamza, and Aladdin Yahya Othman. "Automated android malware detection using optimal ensemble learning approach for cybersecurity." IEEE Access (2023).
13. Aamir, Muhammad, Muhammad Waseem Iqbal, Mariam Nosheen, M. Usman Ashraf, Ahmad Shaf, Khalid Ali Almarhabi, Ahmed Mohammed Alghamdi, and Adel A. Bahaddad. "AMDDLmodel: Android smartphones malware detection using deep learning model." Plos one 19, no. 1 (2024): e0296722.
14. Banik, Abhinandan, and Jyoti Prakash Singh. "Android malware detection by correlated real permission couples using FP growth algorithm and neural networks." IEEE Access (2023).
15. Kim, TaeGuen, BooJoong Kang, Mina Rho, Sakir Sezer, and Eul Gyu Im. "A multimodal deep learning method for android malware detection using various features." IEEE Transactions on Information Forensics and Security 14, no. 3 (2018): 773-788.
16. Iadarola, Giacomo, Fabio Martinelli, Francesco Mercaldo, and Antonella Santone. "Towards an interpretable deep learning model for mobile malware detection and family identification." Computers & Security 105 (2021): 102198.
17. Mahindru, Arvind, and A. L. Sangal. "Deepdroid: feature selection approach to detect android malware using deep learning." In 2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS), pp. 16-19. IEEE, 2019.
18. Bayazit, Esra Calik, Ozgur Koray Sahingoz, and Buket Dogan. "A deep learning based android malware detection system with static analysis." In 2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), pp. 1-6. IEEE, 2022.
19. Wasif, Md Shadman, Md Palash Miah, Md Shohrab Hossain, Mohammed JF Alenazi, and Mohammed Atiquzzaman. "CNN-ViT synergy: An efficient Android malware detection approach through deep learning." Computers and Electrical Engineering 123 (2025): 110039.
20. Vinayakumar, R., Mamoun Alazab, K. P. Soman, Prabaharan Poornachandran, and Sitalakshmi Venkatraman. "Robust intelligent malware detection using deep learning." IEEE access 7 (2019): 46717-46738.
21. Gopinath, Mohana, and Sibi Chakkaravarthy Sethuraman. "A comprehensive survey on deep learning based malware detection techniques." Computer Science Review 47 (2023): 100529.
22. Joseph, Janius Christabel. "Multi Classifier Models using Machine Learning Techniques for Malware Detection." PhD diss., Dublin, National College of Ireland, 2021.
23. Gopinath, Mohana, and Sibi Chakkaravarthy Sethuraman. "A comprehensive survey on deep learning based malware detection techniques." Computer Science Review 47 (2023): 100529.
24. Ababneh, Mustafa, and Aayat Aljarrah. "cybersecurity: Malware multi-attack detector on android-based devices using deep learning methods." Journal of Theoretical and Applied Information Technology 102, no. 1 (2024).