EAI Endorsed Transactions
on Scalable Information Systems
Research Article EAI.EU

Malware detection for Android application using Aquila optimizer and Hybrid LSTM-SVM classifier

M. Grace[a,*] and Dr. M. Sughasiny[b]

[a]Research Scholar, Department of Computer Science, Srimad Andavan Arts and science College, Affiliated to Bharathidasan University, Tiruchirappalli, Tamil Nadu 620005, India.
[b]Assistant Professor, Department of Computer Science, Srimad Andavan Arts and science College, Affiliated to Bharathidasan University, Tiruchirappalli, Tamil Nadu 620005, India.

Abstract

INTRODUCTION: Android OS is the most recent used smartphone platform in the world that occupies about 80% in share market. In google play store, there are 3.48 million apps available for downloading. Unfortunately, the growth rate of malicious apps in google play store and third party app store has become a big concern, which holds back the development of the Android smartphone ecosystem.
OBJECTIVES: In recent survey, a new malicious app has been introduced for every 10 seconds. These malicious apps are built to accomplish a variety of threats, such as Trojans, worms, exploits, and viruses. To overcome this issue, a new efficient and effective approach of malware detection for android application using Aquila optimizer and Hybrid LSTM-SVM classifier is designed.
METHODS: In this paper, the optimal features are selected from the CSV file based on the prediction accuracy by cross validation using Aquila optimizer and the mean square error (MSE) obtained by the cross validation is consider as the fitness function for the Aquila to select the optimal features.
RESULTS: The extracted optimal features are given to the Hybrid LSTM-SVM classifier for training and testing the features to predict the malware type in the android system.
CONCLUSION: This proposed model is implemented on python 3.8 for performance metrics such as accuracy, precision, execution time, error, etc. The acquired accuracy for the proposed model is 97%, which is greater compared to the existing techniques such as LSTM, SVM, RF and NB. Thus, the proposed model instantly predicts the malware from the android application.

Keywords: Malware detection, Hybrid LSTM-SVM, Aquila optimizer, k-fold Cross validation.

Received on 18 March 2022, accepted on 23 July 2022, published on 22 August 2022

doi: 10.4108/eetsis.v9i4.2565

1. Introduction

Mobile phones has become an important thing in our life and they are similar to the personal computers. Because of that the android based mobile phones are recently developed and are ideal target for the attackers [1]. Many applications can be downloaded from the android application market for the users. But the uncertified apps may contain malware applications, which can steal secured information from the users. These apps are not only downloaded from the google play store but also by a number of third party apps that provides popularity boost to the apps [2]. Unfortunately, the popularity of the apps attracts the threat creators to invade threat on smartphones for malicious activities in that certain apps perform notorious and has more than 50 variants of malicious activities that makes very challenging to detect them all [3].

[*]Corresponding author. Email: grace.phd.2021@gmail.com

Aquila optimizer is a feature extractor, which is based on the behaviour of Aquila bird originated from the northern hemisphere [4]. Aquila bird belongs to the 'Accipitridae' group, young Aquila will have white colour on the tail and matured Aquila will be dark brown in colour with lighter golden-brown plumage on their back of the neck. The Aquila has four different methods of hunting behaviours in that the first approach is using a high soar with a vertical stoop to hunt their preys in flight from a high elevation above the ground, the second approach is contour fly that occurs at a lower level over the ground and has a brief glide attack [5]. The Aquila's third approach is low flying with a gradual descent attack, in which the bird glides to the ground and attacks the prey, the fourth approach is grabbing the prey by walking on the ground these are the different hunting behaviours of the Aquila bird. The Aquila is the most intelligent bird and skilful hunter next to the humans.

The LSTM (Long Short Term Memory) architecture is similar to the standard RNN that has a collection of repeated modules for each time step [6]. The set of gates is a function of input at the previous and current time step for each step of time that controls the output module. The LSTM consist of input gate, forget gate, and output gate [7]. Gates are used to decide whether to update the current memory cell and current hidden state. The memory dimension is denoted as d in the LSTM [8]. All vectors in the architecture shares the same dimension. LSTM is designed for time-series data to learn dependencies of long-term. SVM (Support Vector Machine) is a new learning method introduced by vapnik for solving pattern recognition of two-class [9]. This technique is defined to find a best decision surface on a vector space, which splits the data into two classes. The decision surface will be a hyper plane for separating the two classes. For objects with linearly inseparable will be transformed from original data to a linearity separable hyperplane and high dimensional space using a non-linear mapping can be establish in new space without increasing the complex computation of quadratic programming problem by using the kernel function [10].

In recent decade the most of the mobile phones uses the android platform. Due to this technology influence people are widely using the android devices for handling and sharing their information or data. But android is an open source and young operating system. So, there is no proper security mechanism developed for android. Because any one can develop and upload their own application or can install in android. So the application can access personal data or the application can easily use to modify the functionality of the device. It is a major hazard to the personal or secure information. Hence it is essential to monitor the activity of every installed application in android. Hence an android security or malware detection system is essential to enhance the security of android system.

Major contribution of this proposed model is given below:

- ❖ Prediction of different malware application type has been implemented by using machine learning algorithm
- ❖ CSV files from the android applications are collected and gathered in a tabular format for feature extraction.
- ❖ Extraction of features for machine learning algorithm has been done by using Aquila optimizer
- ❖ A hybrid LSTM-SVM model is used for an effective classification and prediction of malware types

The upcoming sections of the paper is ordered as follows, section 2 explains some research paper related to existing techniques used for malware prediction. Section 3 describes briefly about the description of proposed methodology. Section 4 describes the results and performance metrics of the proposed framework. Section 5 concludes the entire research work.

2.Literature Survey

Several researches has been executed with many techniques to predict malware type from android application. Most of the existing techniques are designed based on Naive Bayes, Random forest, SVM, DNN and LSTM among those existing techniques few of them are reviewed below.

Koundel, D *et al.* [11] had designed a Malware Classification using Naïve Bayes Classifier for Android app. In this model, Naive Bayes is used to classify an app as benign or malware by using data mining. For categorizing an application, various attributes of an app is used, which are based on battery permission, acquired rating by the application on android market and permission used by application. Naive Bayes is applied for classification and it deduces the results based on the application probability of being malware or not. The results from the classifier is uploaded to the cloud for future use.

Khammas, B. [12] had performed Detection of Ransomware by using Random Forest Technique. In this method, a new technique is introduced by the analysis of static to identify ransomware. An important characteristic of this model is to dispense the dismantle process by extracting directly the features by using the frequent pattern mining from the raw byte that remarkably enhances the speed of the detection. The technique named Gain ratio is used to select the features that exhibited 1000 optimal number of features for detection process. The random forest classifier is used in combination with a complete analysis to impact both seed numbers and tree on ransomware detection. The seed with 1 number and tree with 100 number is attained for the outcomes of bets in time consuming and accuracy.

Ye, Y *et al.* [13] had designed an interpretable string based malware detection system using SVM ensemble with bagging. In this model, the malware is detected by an interpretable string based system. This is an analysis of interpretable string and uses support vector machine (SVM) to classify the sample files and predict the exact types of the malware. Interpretable string contains both important semantic strings and application programming layer (API) that reflects an attacker' goal and intent. SBMDS (String Based Malware Detection System) achieves a better performance.

Li, Y *et al.* [14] had developed a machine learning framework for malware detection using domain generation methods. A machine learning framework has been implemented in this designed model for identifying the DGA to relieve threat. The significant real-time data is obtained from real-world traffic. Deep learning algorithm is used in this designed model to classify huge sum of DGA. For prediction, a time sequence method is implemented to predict the incoming features according to the Hidden Markov Model (HMM). Also, the gradually collected massive dataset is controlled by DNN (Deep Neural Network) algorithm to increase the framework of machine learning.

Lu, R. [15] had designed Malware detection with LSTM using opcode language. This model is designed based upon Natural Language Processing (NLP) for an efficient method to perform analysis of static malware, which can learn automatically the malware from opcode sequence. Word embedding technique is used for representing the feature vector of opcode. For malware detection, a two-stage LSTM model is used [16]. This LSTM model has two layers in that the first is mean-pooling layer to acquire feature vectors of opcode and the second layer is used of classification. This experiment is performed with 969 malware and 123 benign files [17].

Kouliaridis V et al. [21] have developed an application for malware detection in android platform using java script. Singh R et al. [22] have used location-sharing data from Swarm to explore spatio-temporal, geo-temporal and behaviour patterns within the city of Melbourne to understand the mobility patter and analysis the sentiment. Zhang F et al. [23] have evaluated the Decision-based evasion attacks on tree ensemble classifiers and found that random forests can be even more vulnerable than SVMs, either single or ensemble, to evasion attacks under both white-box and the more realistic black-box settings. Yin J et al. [24] have designed an Adaptive Sliding Window Weighted Learning (ASWWL) algorithm to tackle the dynamic multiclass imbalance problem existing in many industrial applications including exploitation time prediction.

From the above-mentioned reviews, various malware detection method is designed based on Naïve Bayes [11], Random forest [12], SVM [13], DNN [14] and LSTM [15] technique. The performance of the existing model is less compared to the proposed model. Therefore, selection of hybrid LSTM-SVM model in this proposed research is used for effective detection of malware from the android application.

3. Proposed Methodology

Development of smartphone has enhanced both the technology and threats for the smartphone users, these threats will perform malicious activities on sensitive data. The manual experimental review of malware analysis is no longer considered as an effective and efficient compared against the high spreading rate of malware. So, to detect and predict the malware apps instantly a design named malware detection for android application using an Aquila optimizer and Hybrid LSTM-SVM classifier has been implemented for the effective and efficient prediction of the malware apps.

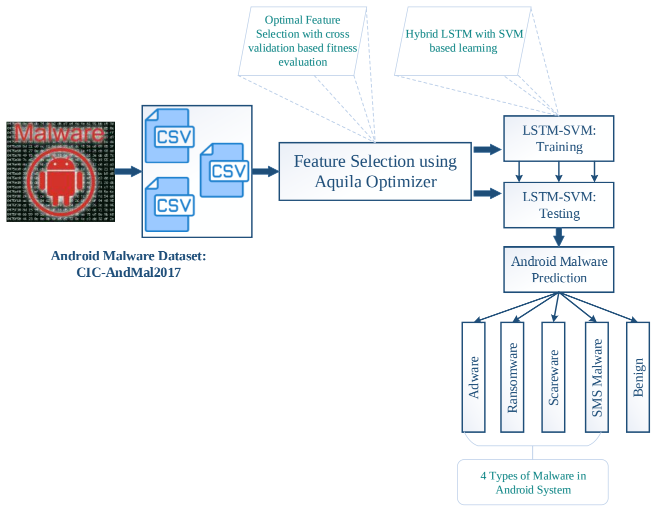
Optimal Feature Selection with cross validation based fitness evaluation
Hybrid LSTM with SVM based learning
Malware
CSV
CSV
CSV
Feature Selection using Aquila Optimizer
LSTM-SVM: Training
LSTM-SVM: Testing
Android Malware Dataset: CIC-AndMal2017
Android Malware Prediction
Adware
Ransomware
Scareware
SMS Malware
Benign
4 Types of Malware in Android System

Figure 1. Process flow of Malware Detection for Android Application

Figure 1 illustrates the process flow of malware detection for android application. The first step of the malware detection is to collect the data for feature extraction, which consist of android malware dataset in the form of CSV, apk, which is a collection of data saved in the tabular format. In the second step, the optimal features are selected from the CSV file based on the prediction accuracy by cross validation using Aquila optimizer. The mean square error (MSE) obtained by the cross validation is consider as the fitness function for the Aquila to select the optimal features. The extracted optimal features are given to the Hybrid LSTM-SVM classifier for training and testing the features to predict the malware type in the android system.

A. Step 1: Data Collection

Data collection is a process of collecting the data's of the log file from the malware apps. Log file or CSV file is a collection of data saved in the tabular format, these tabular formats consist of information about apk file such as schedule, processing and communication between apps.

Table.1 illustrates the sample log file of the android application; the format of the log file will be device name/date-time/Level/PID /tag: message. The first column of the log file denotes the device name and the second column denotes the date-time of the app, third column consist of log levels of the app such as info, error, warning and fatal. The fourth column consist of PID, which is a unique number that identifies the running processes. The fifth and sixth column consist of tag and text, where tag represents the services of the app and text represents the message for the services.

Table 1. Sample Log file of the android application

Device name	Time	Level	PID	Tag	Text
Samsung_ J7	01-27-2021 06:36:47.557	Info	1365	art	Debugger is active
Samsung_ J7	01-27-2021 06:36:47.600	Info	1365	System.Out	Application/True caller/permission/contact requested.
Samsung_ J7	01-27-2021 06:36:48.120	Info	1367	System.Out	Debugger is connected
Samsung_ J7	01-27-2021 06:36:48.540	Info	1368	Wifiservice	Datadownloaded.Android/storage/apps/hike/storage
Samsung_ J7	01-27-2021 06:36:49.260	Info	1368	Wifiservice	Datauploaded.Android/storage/apps/hike/sent
Samsung_ J7	01-27-2021 06:36:49.453	Info	1368	System.Out	Vodafone.5G/connection is terminated
Samsung_ J7	01-27-2021 06:36:49.340	warning	1368	Network provider	Application/Facebook/permission/storage requested.

B. Step 2: Feature Selection

Feature extraction is the process of decreasing the number of resources required to describe a large set of data. Large set of data will consume more memory and time. So, to reduce the time and memory the required data are extracted by using the feature extractor. In this proposed model, Aquila optimizer is used as a feature extractor (16) (17).

3.1 Selection of optimal features with cross validation

When a sufficient data is available for model development, the data is divided into three. A large data is split into training set and the remaining data is divided into two for validation set and test set.

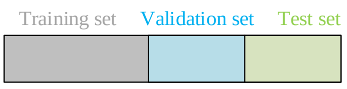

Figure 2. Data split

Cross validation method is used for evaluating the model, when the quantity of training data is limited. The overall training data is split into k segments and the model is trained using k-1 segments and the performance of mean square error is evaluated with held out segment. The same process is repeated with rest of the segments and the average score is considered for model evaluation.

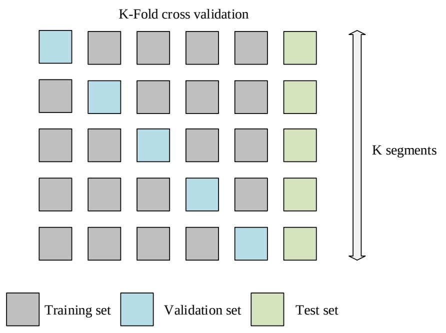

Figure 3. K-Fold cross validation

Figure 3 illustrates the K-fold cross validation, in that first step is to pick the value for K to split the data into folds. When K=3 with 6 observations, this means that the data is shuffled and split it into 3 groups because each group will have an equal number of 2 observations. Three models Fold 1, Fold 2 and Fold 3 are trained and evaluated with each fold given a chance to be the held out test set. The skill score of each model is evaluated and best model is chosen for classification.

3.2 MSE obtained by cross validation

The Mean Squared Error (MSE) explains the distance of regression line set to a data points. It can be done by squaring the distances in between regression line (these distances are the "errors") and points. The squaring is required to eliminate any negative indications and to give more weights to bigger differences. This is defined as the mean squared error. The lower MSE, the better forecast.

$$MSE = \frac{1}{k} * \sum (actual\text{-}forecast)^2 \tag{1}$$

Where, k=number of folds, actual = original value and forecast = value from regression.

3.3 Aquila optimizer for selecting optimal features

In this proposed work, optimal features are obtained by cross validation of log files gathered from the malware apps.

Step 1: Initialization

The features such as device name, Time, Level, PID, Tag and Text are gathered from the log files. These gathered features are considered as attributes $X_1, X_2, X_3 \ldots X_n$.

$$F = (X_1, X_2, X_3 \ldots X_n) \tag{2}$$

Step 2: Fitness Function

The first set attributes are given to the K-fold cross validation. These attributes are randomly divided into K subsets of almost equal size. At a time, one fold is treated as validation set and rest of the folds (K-1) as a training set. The process is repeated until each fold has been used as a validation set. The average of the k-test error is used to calculate the estimation of k-fold Cross Validation. For taking the average of the test error MSE is used and obtained by using equation (1)

$$fitness = \{minimize(MSE)\} \tag{3}$$

The fitness function is evaluated by minimizing the Mean squared error obtained from the cross validation.

Step 3: Updating

Likewise, each and every other set of attributes are updated after the process of cross validation and fitness evaluation

Step 4: Termination

Once the combination of all attributes are calculated, then the process will gets terminated and from that the best combination of attribute is selected and given as an optimal feature for the classifier.

Step 5: Hybrid LSTM-SVM Classification

In this process, the hybrid LSTM-SVM model is used for classification and prediction of malware detection. Hence, this segment will have the detail description of hybrid LSTM-SVM model.

3.4 LSTM

Long Short Term Memory (LSTM) classifier is a specialized form of Recurrent Neural Network (RNN), which can capture a sentence and store in the memory for long-term needs. This classifier performs a better classification than several existing classifier. LSTM is a state-of-the-art algorithm for learning semantic composition and allows it to compute illustration of a document from the representation of its words with multiple abstraction levels [18]. Each word is represented by a real-valued vector, continuous and low dimensional are also known as word embedding. All the word vectors in a word embedding matrix are stacked. The architecture of a single cell LSTM is illustrated in Figure 4.

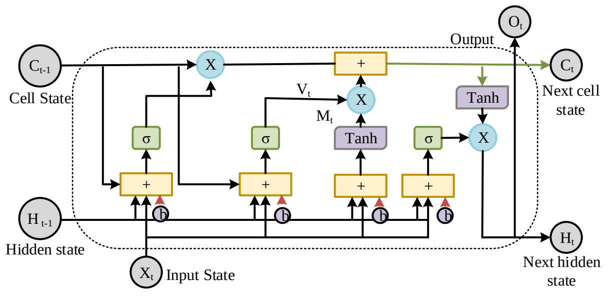

Figure 4. Single cell architecture of LSTM

Steps involved in the LSTM process:

- Forget gate
- Learn gate
- Remember gate
- Output gate

Step 1: Forget gate

The initial step is responsible for recognize the data that is no need for this process. The sigmoid function is responsible for this process and consider the current input X_t at time t and hidden state as H_{t-1} at time t-1. According to the previous output, the sigmoid function will take decision for removing the undesired portion of the output. This process is termed as forget gate(f_t). Likewise, (f_t) lies in-between the limits 0 and 1, which is almost matching to every value in the cell C_{t-1}.

$$f_t = \sigma(wt_f[H_{t-1}, X_t] + bi_f) \tag{4}$$

In equation (1), b_{if} is the bias of forgot gate, wt_f is the weight and σ is the sigmoid function.

Step 2: learn gate

In the second step, there will be two states such as one ignore state and storing state that forms the X_t in cell state. The current input X_t and H_{t-1} are combined together, thus the required data is learned from the hidden state H_{t-1} that can be applied to the X_t. In addition, this method has two layers such as, tanh layer and sigmoid layer. The tanh layer is responsible for passing the values among -1 to 1, this can update the weights. The sigmoid layer is responsible for deciding whether the new data is updated or removed utilizing 0 and 1. Depending on their level of importance, the values are selected.

$$V_t = \sigma(w_f[H_{t-1}, X_t] + b_V) \tag{5}$$
$$M_t = \tanh\,(w_f[H_{t-1}, X_t] + b_M) \tag{6}$$

Step 3: The Remember Gate

The information of cell state is not forgotten in this stage. The hidden state (H_{t-1}) and X_t are combined together (f_t) in remember gate, which can update the information of cell state C_{t-1}. Then it is added with the output of sigmoid layer (V_t) and tanh layer (M_t) to produce the remember gate.

$$C_t = C_{t-1} * f_t + V_t * M_t \quad (7)$$

Step 4: The Output Gate

This stage combines the significant data from previous cell state (C_{t-1}) and previous hidden state V_{t-1} to make a next hidden state(V_{t-1}). The output from the sigmoid and tanh layer are multiplied and added with the previous cell data to produce the next cell state and next hidden state by sigmoid and tanh layer. Thus the output of the current input is acquired from the next hidden state.

$$O_t = \sigma(w_o[V_{t-1}, X_t] + b_o) \quad (8)$$
$$O_t = O_t \tanh(C_t) \quad (9)$$

Where, b_o and w_o are the bias and weight respectively for the output gate. The output from the LSTM is given to the input of the SVM classifier.

3.5 SVM

Support Vector Machine (SVM) is a standard binary classification algorithm. The main objective of the classifier is to determine the optimum hyper plane, which can perfectly split data into two classes.

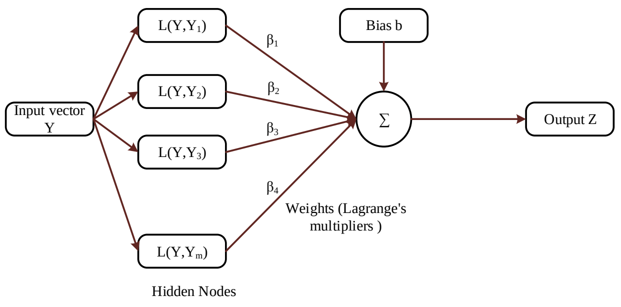

Figure 5. Architecture of SVM

Figure 5 illustrates the architecture of the SVM, Let i be the training instances$\{a_i, b_i\}, i = 1,..M$, each instance consist of an input a_i and a class label $b_i \in \{-1,1\}$. Each hyper plane is parameterized by a weight (w) and a bias (b) vector, which can be illustrated by the following equation (7).

$$w.a + b = 0 \quad (10)$$

The hyper plane function can be classified by training and testing the data and it can be expressed as the following equation (8)

$$f(x) = sign(w.a + b) \quad (11)$$

While dealing with kernel function the prior function will be given as equation (9),

$$f(x) = sign(\sum_{i=1}^{N} \beta_i c_i k(a_i, a) + b) \quad (12)$$

Where, N is the number of training instances, $c_i$is its corresponding class label, a_i is the input of training instance, b is a bias, and $K(a_i, a)$ is the used kernel function which maps the input vectors into an expanded features space [19]. The coefficients β_i are attained subject to two constraints expressed in (10) and (11).

$$0 \leq \beta_i, i = 1,..,N \quad (13)$$
$$\sum_{i=1}^{N} \beta_i c_i = 0 \quad (14)$$

3.6 Hybrid LSTM-SVM

In this proposed model, the hybrid LSTM-SVM model is used for classification of android malware detection. The extracted features from the Aquila optimizer is given as an input to the hybrid model.

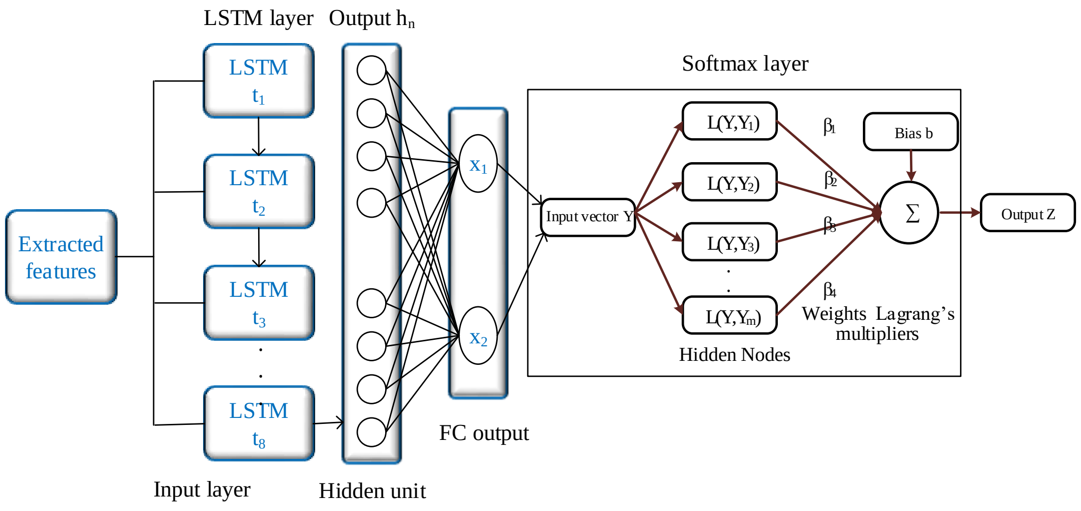

Figure 6. Hybrid LSTM-SVM classifier

Figure.6 illustrates the combined architecture of the hybrid LSTM-SVM model, extracted features are given to input layer. The hidden layer of the LSTM consist of four layers these layers will get interacted and given to the fully connected layer. LSTM is similar to RNN, but it has only one tanh layer. The LSTM has two tanh layer and 3 sigmoid layers these are process of LSTM are explained above. The output from the hidden layer is given to the fully connected layer. The softmax layer of the LSTM is replaced with SVM for classification, softmax layer is a softmax function, which generally used for multi-class classification and its formula is almost close to sigmoid function which is utilized for logistic regression. The SVM performs the classification by finding the hyperplanes that differentiate the classes. Thus, the prediction of android malware type will be acquired as an output from the classifier.

***Algorithm 2**: Pseudocode for malware detection*

```
Input dataset: CSV 1, CSV 2
#Data collection
DC1 = Logfile (CSV1,CSV2,CSV3)
#Feature selection (Aquila optimizer)
For  (optimal features)
 K-fold Cross validation = DC1 // Cross validation for each attributes
 MS = MSE (K-fold Cross validation) // Mean square error for each attributes by using
 equation (1)
 X = fitness evaluated (MS)
 LS = LSTM(X) // Net module created
 SV = SVM(LS) // Classified output for malware detection
end for
Output: Prediction of android malware app
```

4. Result and Discussion

The implementation of the proposed model for malware prediction is done by Python 3.8 software to validate its function and performance. The testing is performed with the processor of Intel(R) Core(TM) i5-10300H processor, CPU @ 2.50GHz, NVIDIA GTX 1650 4GB (GDDR6) GPU, 16.0 GB Memory (RAM) and System type of 64-bit operating system. The first step in this model is to collect the data for feature extraction that consist of android malware dataset in the form of CSV file. In the second step, the optimal features are selected from the CSV file based on the prediction accuracy by cross validation using Aquila optimizer. The mean square error (MSE) obtained by the cross validation is consider as the fitness function for the Aquila to select the optimal features. The extracted optimal features are given to the Hybrid LSTM-SVM classifier for training and testing the features to predict the malware type in the android system.

4.1 Dataset

Android malware dataset named CICAndMal2017 is used in this approach [20], this dataset contains collection of more than 10,854 samples (4,354 malware and 6,500 benign) from several sources. These are collected over six thousand benign apps from Google play market published in 2015, 2016 and 2017.

Table 2. Simulation parameters for malware detection using hybrid LSTM-SVM

Simulation parameters	**Value**
Epoch number	100
batch_size	10
Loss function	mean_squared_error
optimizer	adam
Training	9178
Testing	2294

Table 2 illustrates the simulation parameters for hybrid LSTM-SVM classifier. From the extracted dataset, 11472 (100%) of data is split into 9178 (80%) and 2294 (20%). The 80% of data is used for training and rest of the 20% data is used for testing the trained 80% data. To accomplish the analysis of the proposed model, some of performance metrics are evaluated with the existing techniques for comparison.

Table 3. illustrates the comparison of performance metrics for proposed and existing methods. The existing methods compared with proposed method are LSTM, SVM, RF and NB. From this table, the performance metrics such as Accuracy, Precision, Recall, Error, Specificity, F1_score, Negative Predictive Value (NPV), False Positive Rate (FPR) and False Negative Rate (FNR) of the proposed model is performed greater compared to the existing models.

Table 3. Comparison of performance metrics for proposed and existing method

Performance Metrics	**LSTM-SVM**	**LSTM**	**SVM**	**RF**	**NB**
Accuracy	0.97	0.92	0.91	0.83	0.70
Precision	0.94	0.91	0.85	0.80	0.60
Recall	0.90	0.83	0.86	0.71	0.62
Error	0.03	0.17	0.14	0.29	0.38
Specificity	0.95	0.89	0.88	0.71	0.60
F1_score	0.93	0.89	0.84	0.67	0.65

Negative Predictive Value (NPV)	0.96	0.86	0.78	0.65	0.63
False Positive Rate (FPR)	0.10	0.15	0.17	0.27	0.32
False Negative Rate (FNR)	0.02	0.09	0.10	0.17	0.20

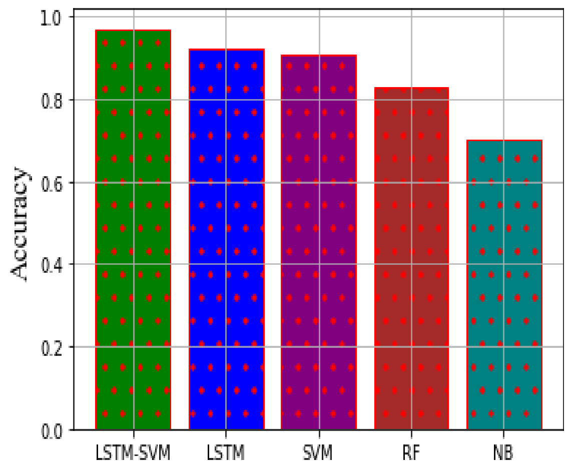

Figure 7. Comparison analysis of Accuracy for proposed and existing method

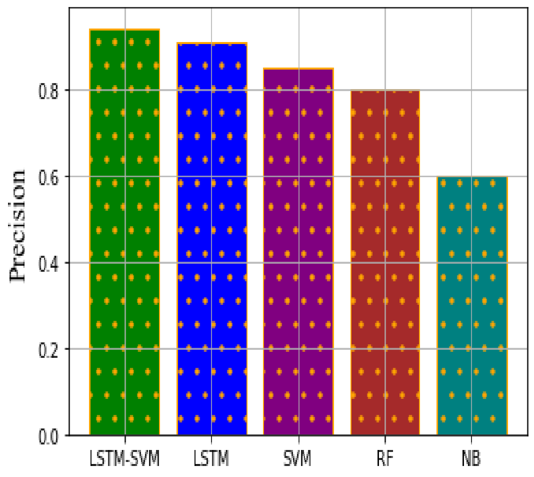

Figure 8. Comparison analysis of Precision for proposed and existing method

The figure 7 illustrates the comparison metrics of accuracy for proposed method with the existing method. The accuracy value of metrics for LSTM, SVM, RF and NB is 0.92, 0.91, 0.83 and 0.70. The accuracy for the hybrid LSTM-SVM is 0.97, which is greater compared to the existing techniques. The figure 8 illustrates the comparison of precision metrics, in this the acquired precision metrics for the hybrid LSTM-SVM is 0.94. Which is greater compared to LSTM, SVM, RF and NB, whose values are 0.91, 0.85, 0.8 and 0.60.

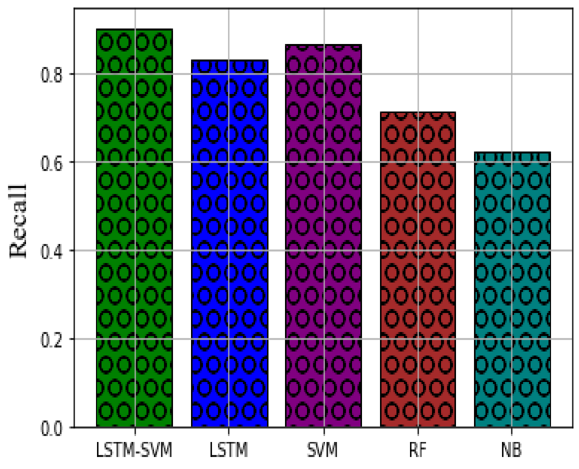

Figure 9. Comparison analysis of Recall for proposed and existing method

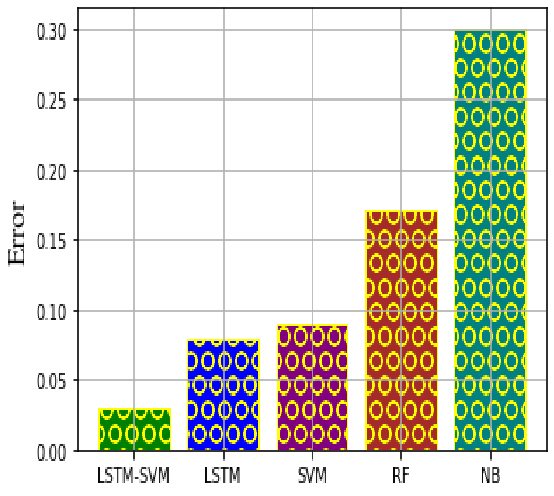

Figure 10. Comparison analysis of Error for proposed and existing method

The figure 9 illustrates the comparison metrics of recall for proposed method with the existing method. The recall value of metrics for LSTM, SVM, RF and NB is 0.83, 0.86, 0.71 and 0.62. The recall metrics of the hybrid LSTM-SVM is 0.90, which is greater compared to the existing techniques. The figure 10 illustrates the comparison of error metrics, in this the acquired error metrics of the hybrid LSTM-SVM is 0.03. Which is lesser compared to LSTM, SVM, RF and NB, whose values are 0.17, 0.14, 0.29 and 0.38.

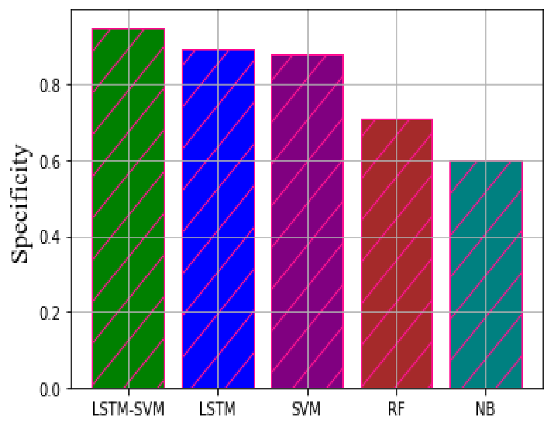

Figure 11. Comparison analysis of Specificity for proposed and existing method

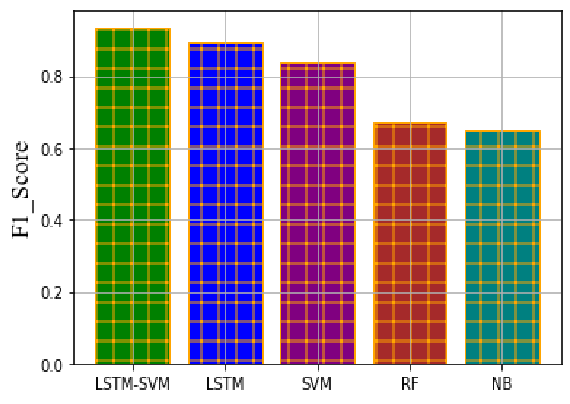

Figure 12. Comparison analysis of F1_score for proposed and existing method

The figure 11 illustrates the comparison metrics of specificity for proposed method with the existing method. The specificity value of metrics for LSTM, SVM, RF and NB is 0.89, 0.88, 0.71 and 0.6. The specificity of the hybrid LSTM-SVM is 0.95, which is greater compared to the existing techniques. The figure 12 illustrates the comparison metrics of F1_score, in this the acquired F1_score metrics of the hybrid LSTM-SVM is 0.93. Which is lesser compared to LSTM, SVM, RF and NB, whose values are 0.89, 0.84, 0.67 and 0.65.

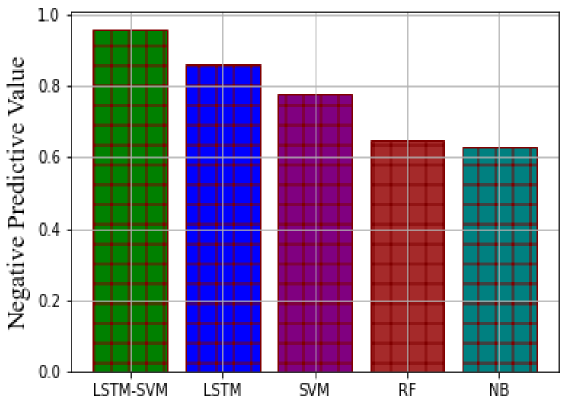

Figure 13. Comparison analysis of Negative Predictive Value (NPV) for proposed and existing method

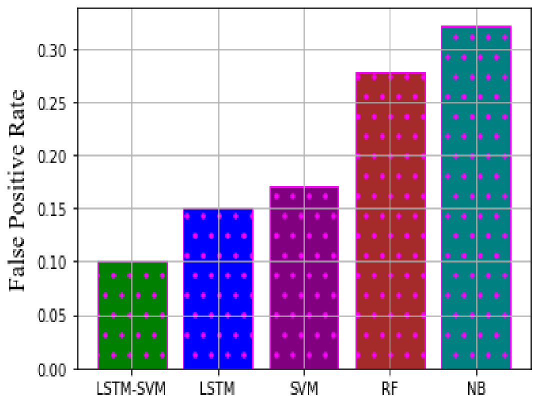

Figure 14. Comparison analysis of False Positive Rate (FPR) for proposed and existing method

The figure 13 illustrates the comparison metrics of Negative Predictive Value (NPV) for proposed method with the existing method. The NPV value of metrics for LSTM, SVM, RF and NB is 0.86, 0.78, 0.65 and 0.63 The NPV of the hybrid LSTM-SVM is 0.96, which is greater compared to the existing techniques. The figure 14 illustrates the comparison metrics of False Positive Rate (FPR), in this the acquired FPR metrics of the hybrid LSTM-SVM. Which is lesser compared to LSTM, SVM, RF and NB, whose values are 0.15, 0.17, 0.278 and 0.322.

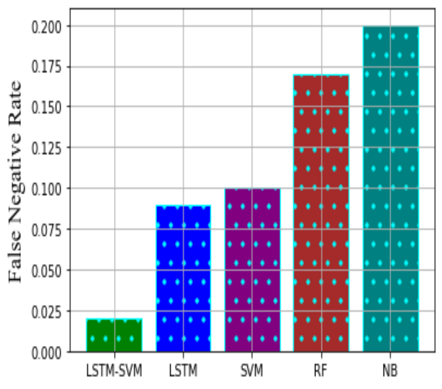

Figure 15. Comparison analysis of False Negative Rate (FNR) for proposed and existing method

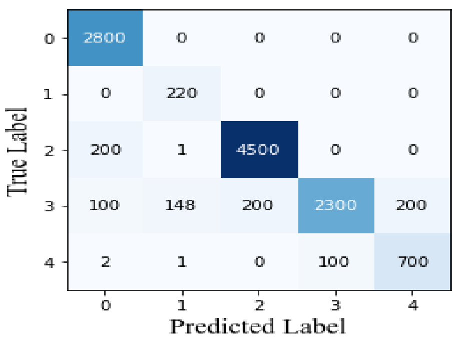

Figure 16. Confusion matrix attained for proposed model

The figure 15 illustrates the comparison metrics of False Negative Rate (FNR), in this the acquired FNR metrics of the hybrid LSTM-SVM is 0.02. Which is lesser compared to LSTM, SVM, RF and NB, whose values are 0.09, 0.1, 0.17 and 0.2. Figure.16 illustrates the confusion matrix of the proposed model, in this the class 0 attains 2800 data, class 1 attains 220 data, class 2 attains 4500 data, class 3 attains 2300 data and class 4 attains 700 data. From this analysis the performance metrics for the proposed hybrid LSTM-SVM model is better compared to the existing techniques such as LSTM, SVM, RF and NB.

5. Conclusion

Malware prediction for android application is done by using the proposed model. Many malicious app that performs various types of attacks such as viruses, exploits, Trojans, and worms. So, to overcome these malicious app the Aquila optimizer and hybrid LSTM-SVM model is used. The process of feature extraction is done by cross validation using Aquila optimizer and the mean square error obtained by the cross validation is consider as the fitness evaluation for the Aquila to select the optimal features. These extracted features are classified by using the hybrid LSTM-SVM model. The performance metrics such as Accuracy, Precision, Recall, Error, Specificity, F1_score, Negative Predictive Value (NPV), False Positive Rate (FPR) and False Negative Rate (FNR) of the proposed and existing model is evaluated and compared. The values for the performance metrics of proposed model are 0.97, 0.94, 0.90, 0.03, 0.95, 0.93, 0.96, 0.10 and 0.02. Thus, the comparison analysis reveals that the proposed model has outstanding performance metrics than the existing method.

Declarations

Funding: There is no funding provided to prepare the manuscript.
Conflict of Interest: The process of writing and the content of the article does not give grounds for raising the issue of a conflict of interest.
Ethical Approval: This article does not contain any studies with human participants or animals performed by any of the authors.
Data availability statement: If all data, models, and code generated or used during the study appear in the submitted article and no data needs to be specifically requested.
Code availability: No code is available for this manuscript.

References

[1] Aung WZZ. Permission-based android malware detection. International Journal of Scientific & Technology Research, 2013, 2(3), 228-234.
[2] Tsiatsikas Z, Kambourakis G, Geneiatakis D, Wang H. The devil is in the detail: SDP-driven malformed message attacks and mitigation in SIP ecosystems. IEEE Access, 2018, 7, 2401-2417.
[3] Ye Y, Li T, Adjeroh D, Iyengar SS. A survey on malware detection using data mining techniques. ACM Computing Surveys (CSUR), 2017, 50(3), 1-40.
[4] Li J, Sun L, Yan Q, Li Z, Srisa-An, W. Ye H. Significant permission identification for machine-learning-based android malware detection. IEEE Transactions on Industrial Informatics, 2018, 14(7), 3216-3225.
[5] Abualigah L, Yousri D, Abd Elaziz M, Ewees AA, Al-qaness MA, Gandomi AH. Aquila Optimizer: A novel meta-heuristic optimization Algorithm. Computers & Industrial Engineering, 2021, 157, 107250.
[6] Greff K, Srivastava RK, Koutník J, Steunebrink BR, Schmidhuber J. LSTM: A search space odyssey. IEEE transactions on neural networks and learning systems, 2016, 28(10), 2222-2232.
[7] Dai J, Chen C, Li, Y. A backdoor attack against LSTM-based text classification systems. IEEE Access, 2019, 7, 138872-138878.
[8] Karim F, Majumdar S, Darabi H, Chen S. LSTM fully convolutional networks for time series classification. IEEE access, 2017, 6, 1662-1669.
[9] Zhang W, Yoshida T, Tang X. Text classification based on multi-word with support vector machine. Knowledge-Based Systems, 2017, 21(8), 879-886

[10] Mitra V, Wang CJ & Banerjee S. Text classification: A least square support vector machine approach. Applied Soft Computing, 2007, 7(3), 908-914.
[11] Koundel D, Ithape S, Khobaragade, V, Jain R. Malware classification using Naïve Bayes classifier for android OS. The International Journal of Engineering and Science, 2014, 3(4), 59-63.
[12] Khammas BM Ransomware Detection Using Random Forest Technique. ICT Express, 2020, 6(4), 325-331.
[13] Ye Y, Chen L, Wang D, Li T, Jiang Q, Zhao M. SBMDS: an interpretable string based malware detection system using SVM ensemble with bagging. Journal in computer virology, 2020, 5(4), 283-293.
[14] Li Y, Xiong K, Chin T, Hu C (2019) A machine learning framework for domain generation algorithm-based malware detection. IEEE Access, 7, 32765-32782.
[15] Lu R. Malware detection with LSTM using opcode language. arXiv preprint arXiv:1906.04593, 2019.
[16] AlRassas AM, Al-qaness MA, Ewees AA, Ren S, Abd Elaziz M, Damaševičius R, Krilavičius T. Optimized ANFIS model using Aquila Optimizer for oil production forecasting. Processes, 2021, 9(7), 1194.
[17] Abualigah L, Yousri D, Abd Elaziz M, Ewees AA, Al-qaness MA, Gandomi AH. Aquila Optimizer: A novel meta-heuristic optimization Algorithm. Computers & Industrial Engineering, 2021, 157, 107250
[18] Lv, Sheng, Zhang H, He H. and Chen B. Microblog rumor detection based on comment sentiment and CNN-LSTM. In Artificial Intelligence in China, Springer, Singapore, 2020, 148-156.
[19] Tharwat, Alaa. Parameter investigation of support vector machine classifier with kernel functions. Knowledge and Information Systems, 2019, 61(3), 1269-1302.
[20] https://www.unb.ca/cic/datasets/andmal2017.html
[21] Kouliaridis V, Kambourakis G, Chatzoglou E, Geneiatakis D, Wang H. Dissecting contact tracing apps in the Android platform. Plos one, 2021, 16(5), e0251867.
[22] Singh R, Zhang Y, Wang H, Miao Y, Ahmed K. Investigation of Social Behaviour Patterns using Location-Based Data–A Melbourne Case Study. EAI Endorsed Transactions on Scalable Information Systems, 2020, 8(31), e2.
[23] Zhang F, Wang Y, Liu S, Wang H. Decision-based evasion attacks on tree ensemble classifiers. World Wide Web, 2020, 23(5), 2957-2977.
[24] Yin J, Tang M, Cao J, Wang H, You M, Lin Y. Vulnerability exploitation time prediction: an integrated framework for dynamic imbalanced learning. World Wide Web, 2022, 25(1), 401-423.