KeAi
CHINESE ROOTS
GLOBAL IMPACT

International Journal of Innovation Studies

AN IMPROVED FEATURE SELECTION TECHNIQUE FOR ANDROID MALWARE DETECTION SYSTEM IN SOCIAL MEDIA USING AN ENSEMBLE OF MACHINE LEARNING MODEL

[*]Saqib Malik[1], Narendra Sharma[2]

[1]Research Scholar, Department of Computer Application, Sri Satya Sai University of Technology and Medical Sciences, Sehore, 466001, India.

[2]Associate Professor, Department of Computer Science, Sri Satya Sai University of Technology and Medical Sciences, Sehore, 466001, India.

***Corresponding author email**: sayyed.saqib531@gmail.com;

ORCID ID: 0009-0007-2100-8705

Email of co-author: [2]dr.narendrasharma88@gmail.com

Acknowledgement

This research was conducted without external funding support. The software used for data analysis and computation was Python.

Abstract

To propose a cross-platform detection system that provides a complete defense by monitoring and analyzing data across various social media platforms (Twitter, Facebook, and Instagram). Ensemble of machine learning methods is used to address real-time challenges such as detecting Android malware in multiple social media platforms to prevent cyber fraud. The method first involved four ML models: random forest, decision tree, naive Bayes, and Stochastic Gradient Tree Boosting. The meta-learner support vector machine combined the outputs from both pre-trained models. The TF-IDF with bag-of-words were utilized in feature extraction for these ML algorithms. Finally, this approach was evaluated on a Kaggle datasets and a preprocessor was applied through Natural Language Processing (NLP) to improve the data quality. Comparing the suggested model to prior models, we found it performed best. The model proposed in this study had the best accuracy when applying this feature selection method, with 98.23% and 96.54% for the two datasets. This study proposes a novel feature selection technique that improves android malware detection system performance more successfully in real-time environment. This study also used NLP for classification of social media text containing malware. The use of ML models in an ensemble with NLP for Android malware detection on social platforms hasn't been widely investigated. This method also improves detection accuracy by using unique feature extraction techniques.

Keywords- Android Malware Detection, Natural Language Processing, Social Media, Stacking Ensemble Model, Machine Learning.

1. Introduction

Android malware seriously threatens smartphone users, compromising personal privacy, financial security, and device operation. Social networks are among the most popular online

services that people use. Social networks are now a great way to share knowledge, reaching locations people never thought possible. Users were the main characters in these interaction spaces, unintentionally contributing to the data generated when they posted comments on an activity that caught their interest. The amount of data that is shared on social media every second is large, in addition to individual users, both private and public organizations, and government departments, along with smart devices such as sensors that incorporate intelligent systems as operations on the IOT technology [1], also generate and publish data. A total of 800,944 complaints about criminal activity provided by the Internet were received in 2022, based on the report of the FBI's Internet Crime Complaint Centre (IC3) [2]. These facts have worried numerous academics, organizations, and businesses to the point where they have encouraged the creation of various tasks, defense systems, and technologies to fend off various threats through malicious individuals or groups that have considered exploiting social media as a means of attack to harm their victims' assets. It is uncertain whether an individual ML method is sufficient due to the difficulty of spotting Android malware in social media conversations [3]. Stacking is a technique that combines the features of multiple algorithms to enhance the performance of an individual algorithm and produce more reliable and accurate predictions. This approach is particularly useful for identifying Android malware in textual data because false positives and false negatives can be severe [4]. Generally, ensemble learning approaches are classified into three types: bagging, boosting, and stacking.

The identification of malware has proven to be a difficult issue that addresses various related issues, including spam emails and pointless remarks on social networking sites. For many decades, several techniques and algorithms have been established for certain areas and diverse goals, including malware propagation, money laundering, spying on industries, theft identification, spam detection, and password collection. A variety of ML algorithms have been employed in present studies on Android malware detection. Djaballah *et al.* [5] focused largely on making individuals aware of their security in this form of attack while utilizing social media platforms. It is a three-step strategy that advises the user to detect questionable links, resulting in a 95% detection rate for phishing emails. Mbungang et al. (2024) [6] demonstrated encouraging findings in static analysis when they suggested using ML in conjunction with Hilbert Space-Filling Curves to detect malware patterns. An innovative malware detection approach based on social network analysis and community detection was developed by Reddy et al. (2021) [7]. The authors showed increased malware detection accuracy by using graph-based community identification techniques to identify harmful entities. In order to get a better knowledge of how malware spreads throughout networks, our study underlined the need of integrating social network dynamics into malware categorization. A feature engineering method for malware family classification was presented by Ahmadi et al. (2016) [8] and integrated innovative feature extraction, selection, and fusion approaches. Their research shown that the accuracy and effectiveness of ML models might be greatly increased by carefully choosing and integrating elements. The method made clear how important it is to reduce high-dimensional feature spaces, which frequently make analyzing big malware datasets more difficult. In order to identify fake news, Hakak et al. (2021) [9] used feature extraction approaches, which are methodologically comparable to malware classification. Word frequencies, sentiment scores, and readability metrics were among the linguistic and semantic data they concentrated on obtaining. These characteristics played a key role in enhancing the ensemble model's capacity to identify minute patterns in the data. Zelinka and Amer (2019) [10] presented a feature extraction method for malware detection that is minimalistic. In order to lower computing complexity without

sacrificing detection accuracy, their approach required choosing a minimal feature set, such as byte frequency and entropy measurements. This low-tech method works especially well in resource-constrained areas. In their approach for intrusion detection, Das et al. (2021) [11] placed a strong emphasis on feature selection. In order to identify and choose the characteristics that contributed most to the classification job, they used methods such as principal component analysis (PCA) and mutual information. The study emphasized how crucial dimensionality reduction is to machine learning process optimization. For Android malware categorization, Islam et al. (2023) [12] used both dynamic and static features. They created a strong feature set by examining authorization patterns, API requests, and behavioral logs, which raised the ensemble models' classification accuracy. Their research showed how domain-specific feature extraction may be used to detect mobile malware. A heterogeneous ensemble framework for social network spam detection was introduced by Zhao et al. (2020) [13]. The approach addressed the issues of data imbalances by extracting text-based and graph-based variables, including word frequencies and node centrality. Their model's outstanding accuracy in classification, especially for minority groups, was made possible by these qualities. Finally, recursive feature elimination (RFE) was used by Garg and Singh (2021) [14] to improve their feature extracting procedure for spam identification. The effectiveness and interpretability of their ensemble frameworks for learning were enhanced by RFE's ability to identify the most pertinent characteristics, such as user activity patterns and message metadata. These methods, which are widely applied in natural language processing, have the potential to greatly improve feature extraction by capturing the semantic connections between system calls and code elements. The creation of real-time, lightweight detection frameworks that can function well on devices with limited resources is still a problem, even with the progress made in ML-based malware detection. Although lightweight models have been presented by Mbungang et al. (2024) and others, their main objective is to reduce computational overhead, and they do not completely address the trade-off between speed and accuracy in real-time detection systems. Malware detection systems may perform much better when these models are combined with ML approaches.

The main research gaps that have been found are the following:

- The necessity of optimal feature extraction technique of ensemble ML algorithms, that can handle both static and dynamic malware features;
- The underutilization of sophisticated feature extraction techniques like BOW and TF-IDF embedding's;
- Combine the NLP methods with ML algorithms to classify the malicious texts in social media platforms;
- The difficulty of developing effective, lightweight feature extraction models that can be used on mobile devices with constrained resources. Android malware detection systems that close these gaps by creating innovative ensemble of ML models that use these methods should become more precise, effective, and flexible.

Thus, to identify and categorize various kinds of malware that originate from social media, this study provides a unique technique that relies on ensemble ML models. This study's primary goal is to construct an ensemble system of ML capable of recognizing malicious text on social media across platforms. The crucial aspect of this methodology involves leveraging natural language processing (NLP) with a stacking ensemble models. NLP techniques include filtering, lemmatization, tokenization, stopword removal, and part-of-speech (POS) tagging. A large

quantity of text may be automatically processed to extract important data and emotions using ML approaches. Recently, the effectiveness of malware identification has significantly improved with Light weight feature selection method with ensemble machine learning model. The greatest challenge with the single bag-of-words method is that, when creating feature vectors, only specific words and their frequencies are considered. In supervised learning systems, classification involves only keywords when feature vector selection depends on those keywords. We will not see the connection between the terms. To address the above issue, this work combines TF-IDF with bag-of-words techniques for feature extraction from a conventional machine learning model. TF-IDF is ideal for highlighting potential keywords for malware discussions and converting the dataset into corresponding TF-IDF feature vectors. After the extraction of features, we use the ensemble ML classifiers (Stochastic gradient boosting Tree, decision trees, random forests, and naïve byes) to determine whether a particular text is malicious. Using the Kaggle dataset, we also conducted a comparative study of the Ensemble model with a single machine learning model. These algorithms extract certain features from textual data within all communications to represent and create the corresponding inputs for the ensemble learning framework.

Consequently, using SVM as a meta-learner, this study presented an enhanced ensemble model for Android malware classification on social media conversation based on combinations of the ML algortiyhms. Following optimization, the final model had the best result in comparison to the existing models.

The following is a summary of the contributions of this paper:

- This paper presented four machine learning architectures: a stochastic gradient boosting Tree, decision trees, random forests, and naïve byes. The hyperparameters of these machine learning models were optimized via a grid search.
- This paper presented an ensemble machine learning model as a single heterogeneous stacking ensemble model. The outputs of each base ML models were combined using SVM as the meta-learner.
- Using two popular social media datasets, namely, “Malignant Comment Classification” and “Instagram posts with #cybersecurity”, the outputs of the proposed model were analyzed by contrasting the stacking model's results with the performances of several conventional machine learning models.
- The proposed stacking ensemble approach for feature extraction outperformed the other models in terms of accuracy, precision, recall, and F1 score.

The findings of this study will be an innovation in the area of Android malware identification over various social network sites. The remainder of the paper is organized as follows: In section 2, the methodology of evaluation used to measure the effectiveness of the suggested strategy are described. Section 3 presents the results and discussion. In Section 6, we provide a conclusion and discuss the paper's potential applications in the future.

2. Methodology

There are four main phases of the proposed ensemble system—the collection of data, data preprocessing, feature extraction, and training—and the predictions are shown graphically in [Figure 1]. Following the preprocessing stages outlined in Section 2.2, the input texts are processed. The produced texts are then subjected to feature extraction through extraction methods. The ML classifiers evolved during the training phase with exploited features. Finally, this model can be applied for classification in the prediction stage. The results show a comparison of the performances of different ML models. The following subsections provide in-depth explanations of the proposed system's key elements.

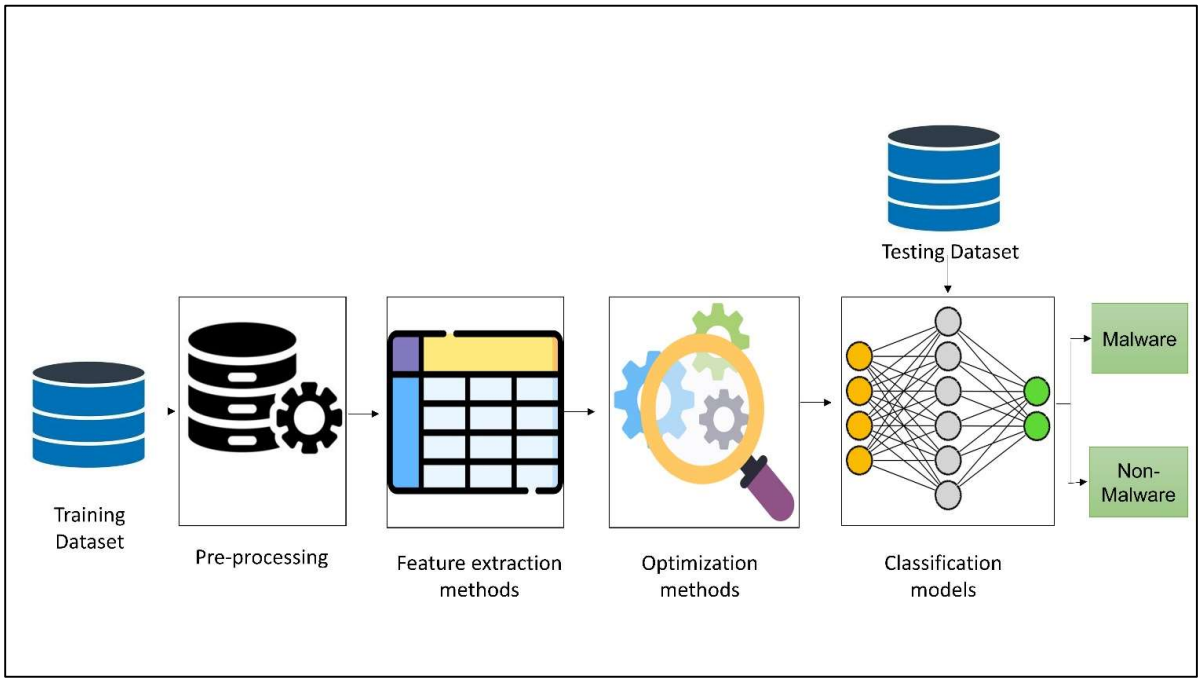


Figure 1. Stages of malware detection on social media platforms.

3.1 Dataset

No dataset combines texts from all three social media sites (Twitter, Instagram, and Facebook) to determine whether they are suspicious. Consequently, we collect two different datasets from Kaggle to accomplish our goals.

3.1.1 Malignant Comment Classification (MCC)

The first dataset, “Malignant Comment Classification”, collected data from Facebook and Twitter groups, mostly about cyberbullying. There are over 1,59,000 samples in the training set and nearly 1,53,000 samples in the test set, which make up the data set. Eight fields—"Id," "Comments," "Malignant," "Highly malignant," "Rude," "Threat," "Abuse," and "Loathe"—are present in every data set. A label with a value of 0 or 1 indicates a NO, whereas a value of 1 indicates a YES. A variety of remarks have more than one label. Each comment has a unique ID, which is the first property.

3.1.2 Instagram posts with #cybersecurity

Another dataset, named "Instagram posts with #cybersecurity", contains collected data from Instagram posts, related mostly to cybersecurity with columns Link, author, img_link, date of upload, likes, timestamp, and caption. The dataset contains 19423 posts in the caption column. These datasets are commonly utilized in sentiment analysis tasks to ensure that the experimental outcomes are accurately evaluated.

3.2 Preprocessing of data

Preprocessing is the second phase where data are prepared using the natural language processing method. The following steps are utilized for preprocessing.

- **Cleaning the data:** This process eliminates unwanted letters in the text without changing its core concept. Various tasks are carried out to filter each message, including removing unknown characters and replacing multiple line breaks and spaces of a single message—applying regular expression rules to remove emojis and emoticons.
- **Lemmatization:** The process of lemmatization reduces a word's morphological variations to its lexeme or roots. As a result, there are fewer words in this dataset, which allows for a reduction in the diversity of words required to convey a message's meaning.
- **Part-of-Speech Tagging (PoS):** Instead of considering a transformation of a generic lemma of the word, in this study, the part-of-speech (POS) tags aim to precisely describe the supposed lemma grammar-based and the context of the content. When the POS tagging technique is used to accurately translate words into lemmas, the dataset's word count is lower than when the raw text of the message is considered.
- **Tokenization:** Tokenization is the process of dissecting each message in a dataset at the word level to create a list of separate elements known as tokens.
- **Stopwords:** Stopwords are essential to the text because they act as the link between grammar and syntax. However, when written by individuals, they lack significance. Therefore, the stopwords in each token array were eliminated to convey each message's essential notion with fewer tokens.

3.3 Feature Extraction For Ensemble ML models:

The texts we generated are not possible for machine learning models to learn from. To extract some meaning from these texts, feature extraction maps these texts numerically. This work investigated the use of BoW with TF-IDF strategies within texts to extract features.

Term Frequency-Inverse Document Frequency (TF-IDF): Particularly for modelling textual data, the most prominent feature extraction method in NLP is TF-IDF [15]. The entire amount (n) with a term of feature (f_{ti}) on a character document (c_i) is indicated in each cell. Using this strategy, unwanted words could be given a larger weight than terms related to the context. The Tf-idf approach applies in Equation (1) to address this weighting issue to obtain the tf-idf value.

$$tf - idf(f_{ti}, c_i) = tf(f_{ti}, c_i)\, log \frac{m}{|t \in m : f_t \in c|} \quad (1)$$

The words connected to the context are given greater weight than other words according to the feature terms tf-idf value ((f_t)). Once the tf-idf value of every word in a sentence has been determined, the Euclidean norm is computed to obtain the overall weighted form of the words. The feature words with lower variance were given more weight by this normalization. The norm is computed via equation (2):

$$Z_{norm}(i) = {Z_i}\Big/{\sqrt{(Z_1)^2 + (Z_2)^2 + \cdots + (Z_n)^2}} \quad (2)$$

In this case, $Z_{norm(i)}$ is the normalized value for the feature term f_i, and $Z_1, Z_2, \ldots, Z_n$ are the tf-idf values of the feature terms f_{t1}, f_{t2},…, f_{tn}, respectively. The classifier was modified with the features selected by both methods.

Bag of Words (BOW): One popular method for obtaining features from textual data is BOW. Counting the words that appear in a document can assist in converting text to numbers, and machine learning methods can be used to process these numerical data.

N-Gram: The N-Gram methodology is widely used in natural language processing, where n denotes a continuous series of phrases or words. This signifies a unigram for n=1. Similarly, the bigram of n = 2 and trigram of n = 3 can be stated. To fit machine learning models, several N–gram approaches have been applied in combination with the TF-IDF and BOW methods for independent feature extraction.

3.4 Optimization of Hyperparameters

Finding the best values for both models' hyperparameters is known as hyperparameter optimization. Cross-validation and grid searches are used to enhance the architecture and hyperparameters of the ML models.

Cross-validation. This is an algorithm assessment technique that divides data among 2 sets: verification of models and model training. This algorithm is sometimes known as k-fold cross-validation since it uses one parameter, k, to specify what number of groups a particular sample of data should be split into. In contrast, the additional k-1 folds will be supplied to the learning model, which guarantees that the data used to make the predictions do not exist during training.

Grid Search. It is a model-tuning strategy that was used to obtain the best possible values for the hyperparameters. If a model includes multiple hyperparameters, it is necessary to look into a space of multiple dimensions to determine the most suitable set of values for the hyperparameter [17]. A certain amount of hyperparameters in complex models might increase significantly, setting tuning manually challenging and highlighting the necessity of the procedures. As seen in [Table 1], we worked to improve a few of the SGTB, and RF parameter values.

Table 1. The possible ranges for the SGTB and RF hyperparameters.

3.5 Training

Several widely used ML models were employed for training the ensemble stacking model with the extracted features. The ML techniques used were stochastic gradient tree boosting (SGTB) decision tree (DT), random forest (RF), and naïve Bayes (NB) describe how they are organized inside our system.

Parameters	Values
Batch-size	Range [32,256]
Num-filters	Range [32,512]
Kernel-size	[3-6]
Pool-Size	[2-6]
LSTM-Unit	Range (25,1500)
Learning-Rate	Between 1×10-1 and 1×10-6

3.5.1 Machine Learning Model

Stochastic gradient tree boosting (SGTB). A variation of the well-known gradient boosting technique, stochastic gradient boosting uses randomization to enhance generalization and lessen overfitting. It is very helpful for tasks like regression and classification. With this approach, an ensemble of decision trees is constructed step-by-step, but with additional stochasticity introduced at each iteration.

Random forest (RF). Regression and classification tasks are combined in RF, a supervised classifier, where the decision tree is the basic component. It is composed of many single DTs and relies on ensemble machine learning.

Decision tree (DT). DTs are a type of supervised machine learning. Using previous data—training data—to learn basic decision rules, a decision tree (DT) is utilized to create a model for training to identify the variable's class to be targeted. A trained tree is produced when the initial set is split into subsections using a test value of an attribute. The method of performing this for every subset is called recursive partitioning.

Naïve Byes (NB). When a feature is present in a class, the NB algorithm determines whether its presence is independent of the presence of another feature. When an additional event has already happened, the probability of an event occurring may be determined using the Bayes theorem.

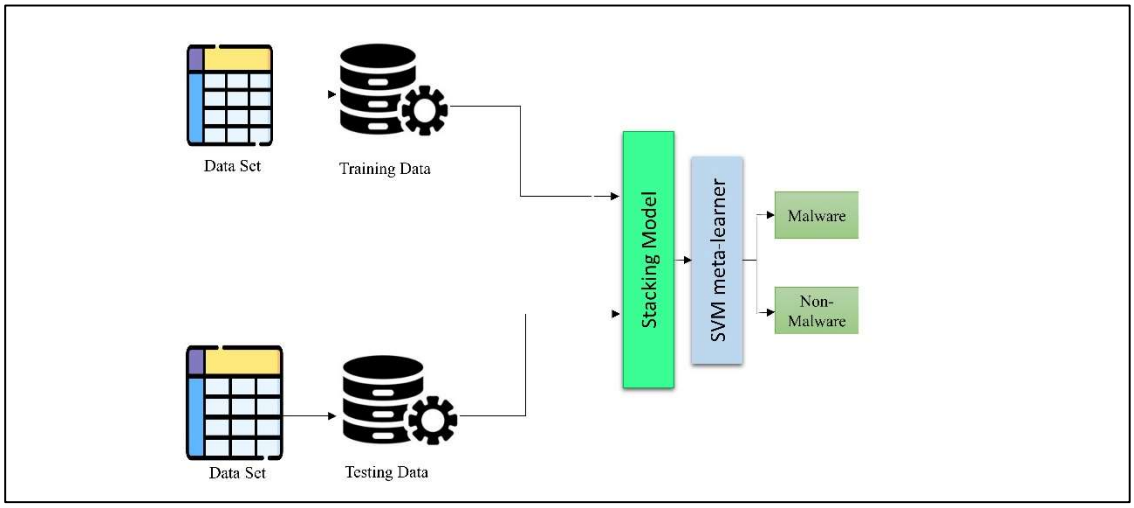


Figure 2. Structure of the proposed stacking ensemble model

3.6 The Proposed Ensemble Stacking Model

The ensemble techniques increase model accuracy by combining many models instead of using only one model. The combined models considerably increase the accuracy of the output. An ensemble technique known as stacking ensembles is a novel approach that combines predictions of many current feasible models. This model integrates predictions of several distinct trained models [18]. The performance of sentiment analysis was much improved by the use of ensemble feature selection models such as SGTB and RF [19, 20]. Majority voting extract more significant characteristics from text input using many ML models. Light weight features a memory state that works well for retaining malicious information from the text and comprehending the phrase as a whole. Thus, using SVM as a meta-learner for Android malware detection from social media, this paper proposed an efficient stacking ensemble approach that relies upon the finest classical machine learning combinations. As Figure 2 illustrates, this study's model is constructed in many phases.

- Section 3.5.1 discusses the ML models, and each model remains inactive except for the output layers.
- The training stack combines each pretrained model's output prediction from the training set. The meta-learner (in our example, SVMs) is then trained and optimized by stacking. A grid search is used to optimize the SGTBs as a meta-learner.
- The test stack is the combination of each pretrained model's output predictions from the testing set. The meta-learner (SVMs) is then examined by testing stacking accuracy, precision, F1-score, and recall.

Two groups of datasets were created: 80% and 20% for training and testing, respectively. The current training optimization set was applied to the models. Unigram and BOW were used for feature extraction and to create feature measures for machine learning algorithms. As shown in [Table 2], the total amount of each ML light weight parameter was used with TF-IDF, and BOW word embedding for both datasets.

Table 2. Hyperparameter values when using stacking ensemble ML with TF-IDF and BOW.

Algorithms	Parameters	TF-IDF		BOW	
		SCSP Dataset's values	**IPC Dataset's values**	**SCSP Dataset's values**	**IPC Dataset's values**
Stacking ensemble ML	Num-filters	[128, 256, 128]	[256, 256, 128]	[256, 256, 512]	[128, 128, 500]
	Pool-Size	[2, 4, 5]	[3, 5, 4]	[2, 4]	[2, 4]
	Kernel-size	[4, 3, 5]	[4, 3, 5]	[5, 4, 5]	[5, 6, 5]
	Dense-Unit	200	400	400	600
	Learning-rate	0.008	0.00155	0.0014	0.0016

3.7 Performance Metrics

A variety of assessment techniques were applied to measure the proposed approach's efficiency, including accuracy, precision, recall, F1-score, and precision-recall curve. The definitions for each are as follows:

The fraction of accurate predictions compared with the entire amount of mass is used to measure accuracy.

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

The fraction of correctly recognized positive messages among each positive message is used to compute precision.

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

The fraction of correctly recognized positive messages among each message is used to compute the recall.

$$\text{Rcall} = \frac{\text{TP}}{(\text{TP} + \text{FN})}$$

The calculated average of the precision and recall is called the F1-score.

$$\text{F1} - \text{score} = \frac{2.\,\text{precision}.\,\text{recall}}{\text{precision} + \text{recall}}$$

Where TP is the proportion of successfully constructed positively expected sentences, FP denotes inaccurately constructed negative sentences, TN denotes correctly constructed positively expected negative terms, and FN denotes correctly created positively predicted phrases.

3. Results and Discussion

The following section provides the results of the proposed stacking model of feature extraction compared to those of the existing ML models. Additionally, it shows the suggested model's outcomes from two Kaggle databases named "Malignant Comment Classification (MCC)" and "Instagram Posts with #Cybersecurity (IPC)". The findings are presented as a precision-recall (PR) curve with four performance metrics.

3.1 The Outcomes of the MCC and IPC Datasets Models

The performance outcomes of the three methods used with the MCC and IPC datasets are shown in this section. The first method uses machine learning models, such as RF, DT, and NB, with TF-IDF and Ngram. The second method uses TF-IDF, BOW and unigram word embedding to apply the ensemble ML model. The suggested model is the third strategy. The performance metric values of both method outcomes are presented in [Table 3].

In terms of machine learning models, RF with unigrams performed best (86.92% accuracy, 87.52% precision, 86.98% recall, and 86.21% f1-score), whereas DT with bigrams performed worst (67.85% accuracy, 50.50% precision, 69.11% recall, and 57.35% f1-score). Comparing the unigram method with RF, it enhanced the accuracy to 6.6%, the precision to 5.97%, the recall to 6.12%, and the f1-score to 6.24%. With 88.55% accuracy, 88.92% precision, 88.61% recall, and 88.52% f1-score. A comparison of the proposed stacking ensemble model with Unigram revealed that the model's performance increased by 0.71% in terms of accuracy, 0.50% in terms of precision, 1% in terms of recall, and 0.69% in terms of the f1-score. Furthermore, for the suggested model for the MCC dataset, the PR curves with the AUC values for both models are shown in [Figure 3]. The precision-recall curve AUC value of 94.15 was the highest for the suggested model using Unigram word embedding. When comparing the areas under the curve (AUCs) of the suggested models with those of the ML models, the ML models using BI-grams had the worst results, at 83.45, 78.24, and 79.26. When using the BOW model, our model achieves the second-best AUC score (93.42). Compared with existing models, the suggested model with TF-IDF embedding performed the best, and all the performance metrics with AUCs were consistent. For IPC dataset, in terms of machine learning models, RF with Bigram performed best (87.25% accuracy, 88.63% precision, recall with 87.10%, and 84.95% F1-scores), whereas NB including Bigram performed worst (70.82% accuracy, 68.61% precision, 65.25% recall, and 62.46% F1-score). Comparing the proposed stacking ensemble model with TF-IDF to SGTB with Bigram, the performance of the proposed model increased by 2.79%, 0.81%, 1.15%, and 0.93%, respectively, for all four performance metrics.

Table 3. MCC and IPC datasets outcomes with both models.

Techniques	ML algorithms	Methods for Feature Extraction	Performance on MCC dataset				Performance on IPC dataset			
			Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
	RF	Unigram	86.92	87.52	86.98	86.21	80.38	81.55	79.65	78.45

Classical ML techniques		Bigram	70.56	48.65	69.25	58.26	87.25	88.63	87.10	84.95
	DT	Unigram	80.84	81.69	80.45	78.89	82.54	80.69	81.45	79.89
		Bigram	67.85	50.50	69.11	57.35	79.85	78.40	77.21	78.38
	NB	Unigram	86.78	84.35	83.57	82.56	75.75	74.35	73.57	72.26
		Bigram	77.85	75.71	79.85	75.26	70.82	68.61	65.25	62.46
	SGTB	Unigram	89.82	88.56	88.65	88.69	92.59	91.76	91.45	92.65
		Bigram	93.52	92.95	93.10	92.45	92.10	92.83	91.80	90.35
The proposed stacking ensemble model	Stacking	TF-IDF	98.23	93.45	94.10	93.46	96.54	95.64	93.82	95.28
	Stacking	BOW	93.30	92.21	93.22	91.95	93.98	93.53	92.52	93.25

Furthermore, for the suggested model for the IPC dataset, the PR curves with the AUC values for both models are shown in [Figure 4]. The precision-recall curve AUC value of 93.65 was the highest for the proposed model using TF-IDF word embedding. When comparing the areas under the curve (AUCs) of the suggested models to those of RF, DT, and NB with those of the machine learning models, the ML models using BI-grams had the worst results, at 80.25, 79.44, and 76.46, respectively. When using the BOW method, our suggested model achieves the second-best AUC (91.52).

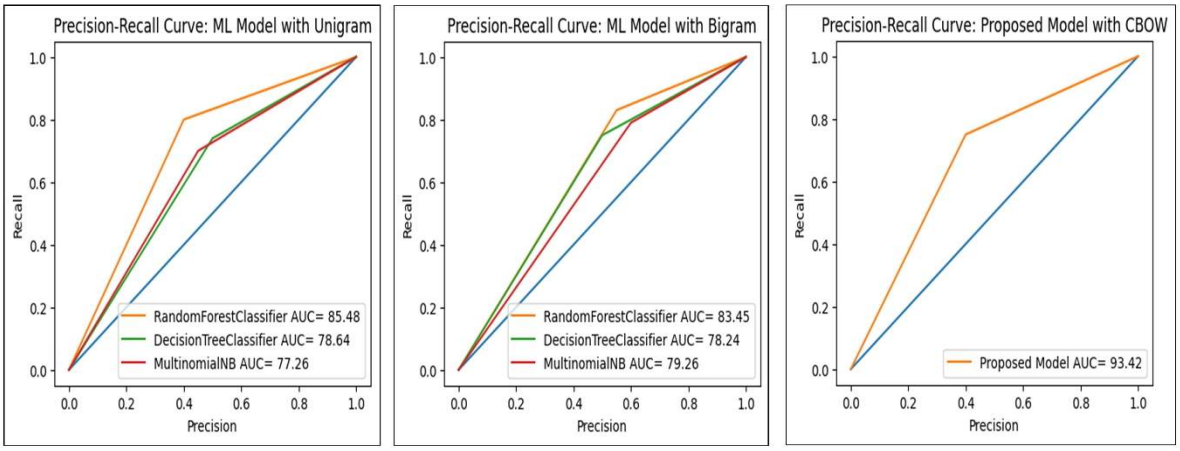
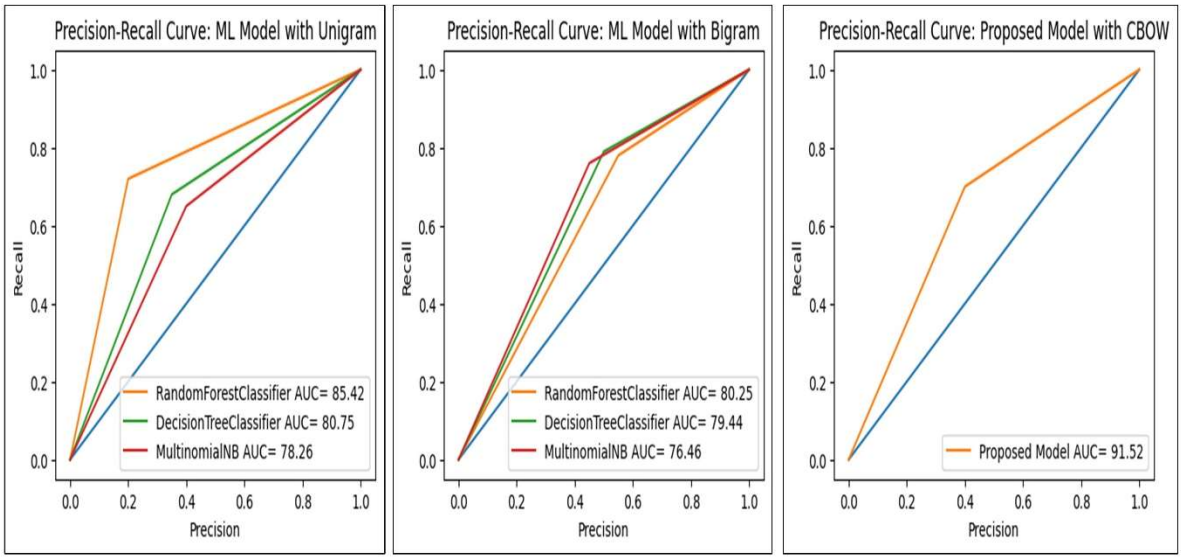


Figure 3. Precision–recall curves with AUC values for the MCC datasets.



4(b)

Figure 4. Precision–recall curves with AUC values for the IPC datasets.

A significant aspect of this study is the combination of NLP tools with an ensemble of ML algorithms. Studies ([12], [13], and [14]) have demonstrated that cyberattack classification tasks are collaborative methods. [Figure 5 and Figure 6] show both datasets' top-performing models. For the ML model, the RF performed the best for both datasets. Despite all the enhanced ML algorithms, the proposed feature extraction model performed best. The performance of the suggested ensemble stacking model is related to these models. The suggested models performed best with TF-IDF compared to the existing models for each dataset. With the MCC dataset, the highest-performing TF-IDF model achieved an accuracy of 98.23%, 93.45% precision, 93.95% recall, and an F1 score of 93.14%. However, DTs with two grams performed the worst (67.85% accuracy, 50.50% precision, 69.11% recall, and 57.35% F1 score). For the IPC dataset, the proposed model performed best, with a TF-IDF accuracy of 96.54%, a precision of 94.64%, a recall of 93.95%, and an F1 score of 94.28%. DTs with two grams performed the worst (79.85% accuracy, 78.40% precision, 77.21% recall, and 78.38% F1 score).

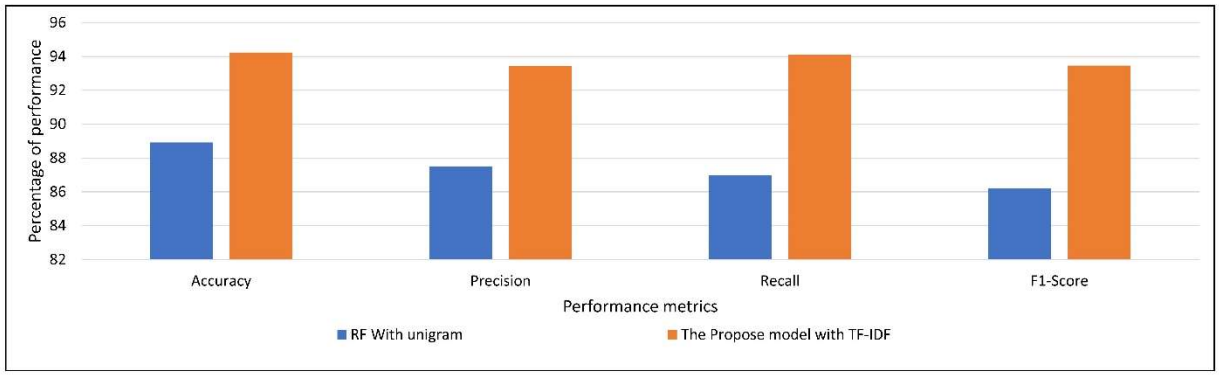


Figure 5. The highest-performing model with the MCC dataset

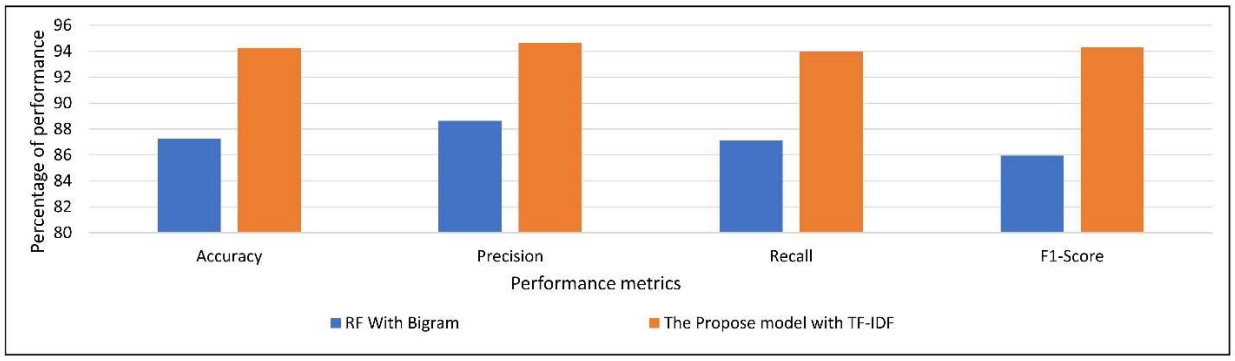


Figure 6. The highest-performing model with the IPC dataset

When we compared the proposed model to existing models, the suggested model performed better than the others, as we found. Compared with the authors who utilized the MCC dataset in [21], MNB, LSVC, LR, and KNN achieved accuracies of 91.26%, 91.42%, 83.39%, and 90.39%, respectively. In their study [18], for the MCC dataset, the accuracy of the proposed stacked model reached 90.97, and the F1 score for the same algorithm reached 91.02. Nevertheless, the suggested method in the study of [5] completes the detecting task whether or not the text contains a URL. It identifies malicious tweets that cause phishing attacks with an accuracy of 74.96%. In their study [22], LSTM was achieved with 92.7% accuracy. In the study of [23], an accuracy of 93% was achieved by CNN-LSTM for the Twitter dataset. The results of [24] for the weighted soft voting classifier were 85.54% for accuracy and 90.12% for the F1-score. In [29], the authors used the combination of five ML and four DL CNN, LSTM, BI-LSTM, and GRU models to achieve average accuracies of 91.61%, 90.77% precision, 90.88% recall and 91.66% F1-score, respectively. However, this proposed model obtained the best result compared to the existing models with different approaches. A comparison of the available literature for each social media dataset with that of the proposed model is presented in [Table 4].

Table 4. The suggested models are compared with existing research.

Publications	**Models**	**Datasets**	Result
[21]	MNB, LSVC, LR and KNN	MCC	91.26%, 91.42%, 83.39% and 90.39% accuracy
[18]	BiLSTM, Conv1DLSTM, LSTM-CNN	MCC	90.97% for accuracy 91.02% for F1-score
[5]	LR, SVM and RF	UCI Machine Learning phishing dataset	74.96% for accuracy
[22]	LSTM	Spam comments and fraudulent email	92.7% for accuracy

[23]	CNN-LSTM	Twitter_Data, Apple-twitter-sentiment-texts, FinalSentimentdata2, Tweets	93% for accuracy
[24]	weighted soft voting classifier	D1 and D2	85.45% for Accuracy and 90.12% of f1-score
[25]	CNN, LSTM, GRU, BiLSTM	suspicious tweets dataset	91.61% for accuracy, 90.77% for precision, by recall 90.88%, and F1-score is 91.66%.
Suggested Model	Stacking Ensemble Model relying on ML	MCC	98.23% accuracy, 93.45% precision, 93.95% recall and the f1-score is 93.14%
		IPC	96.54% for accuracy, 95.64% for precision, 93.82% for recall, and 95.28% for f1-score.

4. Conclusion

Social media have recently been used as attack vectors to spread harmful data to several people. This work offered possible security measures: a method for evaluating social network communications on cross platforms using a stacking ensemble model relying on ensemble machine learning with NLP tools to identify and categorize each element as malware or no-malware. Furthermore, the accuracy of the proposed ensemble models significantly improved. This proposed model combines a four ML model using a SVM meta-learner to improve the model's ability to detect Android malware on cross platforms. After extracting features within the texts for the ML models, the TF-IDF and BoW feature extraction strategies were utilized. The 300 dimensions of the TF-IDF and BOW methods were used for extracting features for the ML model. Moreover, the ensemble stacking of the 300-dimensional word embedding of the TF-IDF and BOW models was applied to extract features for ML models. Compared to different ML models, our ensemble model for feature extrection performs best. The best accuracy for “Malignant Comment Classification” (MCC) and “Instagram posts with #cybersecurity” (IPC), respectively, is achieved by the suggested model with TF-IDF word embedding, which scores 98.23% and 96.54%, respectively. In further research, we will concentrate on improving the efficacy of malware identification assignments by including larger and real-time datasets or additional natural language processing (NLP) methods, such as contextual analysis or the use of GloVe for vectorization. Another avenue for future research may include exploring advanced deep-learning architectures that are specifically designed for text classification tasks. The reason for these improvements is that they could increase the quality of the output obtained through this study. Moreover, the

addition of more features in addition to text, such as user profiles or network information, may improve the reliability of detection. Despite some minor shortcomings, the current study confirms that the use of ensemble learning together with NLP and feature extraction for Android malware detection on social media is promising. However, it also provides scope for further investigation into advanced ML architectures with real-time datasets to develop stronger and more reliable systems for detecting Android malware on social networks.

5. References:

[1] Llorián DM, García CG, Bustelo BCPG, Lovelle JMC. BILROST: Handling actuators of the internet of things through tweets on twitter using a domain-specific language. IJIMAI. 2021;6(6):133-144. DOI: 10.9781/ijimai.2021.01.004

[2] Internet Crime Complaint Center (U.S.), United States, Federal Bureau of Investigation. 2022 internet Crime Report. 2022; 1-32. DOI:https://www.ic3.gov/Media/PDF/AnnualReport/2022_IC3Report.pdf

[3] Singh A, Kaur M. Cuckoo inspired stacking ensemble framework for content-based cybercrime detection in online social networks. Trans Emerg Telecommun Technol. 2021;32(8). DOI: https://doi.org/10.1002/ett.4074

[4] Dong X, Yu Z, Cao W, Shi Y, Ma Q. A survey on ensemble learning. Front Comput Sci. 2020;14(2):241-258. DOI: https://doi.org/10.1007/s11704-019-8208-z

[5] Djaballah KA, Boukhalfa K, Ghalem Z, Boukerma O. A new approach for the detection and analysis of phishing in social networks: the case of Twitter. In: 2020 Seventh International Conference on Social Networks Analysis, Management and Security (SNAMS). IEEE; 2020. p. 1-8. DOI: https://doi.org/10.1109/SNAMS51200.2020.9312674

[6] Mbungang BN, Ali Wacka JB, Tchakounte F, et al. Detecting Android Malware with Convolutional Neural Networks and Hilbert Space-Filling Curves. SN Comput Sci. 2024;5:810. DOI: https://doi.org/10.1007/s42979-024-03123-6

[7] Reddy, V., Kolli, N., & Balakrishnan, N. (2021). Malware detection and classification using community detection and social network analysis. *Journal of Computer Virology and Hacking Techniques, 17*(4), 333–346. https://doi.org/10.1007/s11416-021-00387-x

[8] Ahmadi, M., Ulyanov, D., Semenov, S., Trofimov, M., & Giacinto, G. (2016). Novel feature extraction, selection and fusion for effective malware family classification. In *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy* (pp. 183–194). https://doi.org/10.1145/2857705.2857713

[9] Hakak, S., Alazab, M., Khan, S., Gadekallu, T. R., Maddikunta, P. K., & Khan, W. Z. (2021). An ensemble machine learning approach through effective feature extraction to classify fake news. *Future Generation Computer Systems, 117*, 47–58. https://doi.org/10.1016/j.future.2020.11.022

[10] Zelinka, I., & Amer, E. (2019). An ensemble-based malware detection model using minimum feature set. In *Mendel* (Vol. 25, No. 2, pp. 1–10). https://doi.org/10.13164/mendel.2019.2.001

[11] Das, S., Saha, S., Priyoti, A. T., Roy, E. K., Sheldon, F. T., Haque, A., & Shiva, S. (2021). Network intrusion detection and comparative analysis using ensemble machine learning and feature selection. *IEEE Transactions on Network and Service Management, 19*(4), 4821–4833. https://doi.org/10.1109/TNSM.2021.3130911

[12] Islam, R., Sayed, M. I., Saha, S., Hossain, M. J., & Masud, M. A. (2023). Android malware classification using optimum feature selection and ensemble machine learning. *Internet of Things and Cyber-Physical Systems, 3*, 100–111. https://doi.org/10.1016/j.iotcps.2022.100111

[13] Zhao, C., Xin, Y., Li, X., Yang, Y., & Chen, Y. (2020). A heterogeneous ensemble learning framework for spam detection in social networks with imbalanced data. *Applied Sciences, 10*(3), 936. https://doi.org/10.3390/app10030936

[14] Garg, P., & Singh, S. N. (2021). Analysis of ensemble learning models for identifying spam over social networks using recursive feature elimination. In *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)* (pp. 713–718). IEEE. https://doi.org/10.1109/Confluence51648.2021.9377053

[15] Ahuja R, Chug A, Kohli S, Gupta S, Ahuja P. The impact of feature extraction on the sentiment analysis. Procedia Comput Sci. 2019;152:341-348. DOI:https://doi.org/10.1016/j.procs. 2019.05.020

[16] Wang B, Wang A, Chen F, Wang Y, Kuo CCJ. Evaluating word embedding models: Methods and experimental results. APSIPA Trans Signal Inf Process. 2019;8 DOI: https://doi.org/10.1017/atsip. 2019.12

[17] Fayed HA, Atiya AF. Speed up grid-search for parameter selection of support vector machines. Appl Soft Comput. 2019;80:202-210. DOI:https://doi.org/10.1016/j.asoc.2019.04.033

[18] Muneer A, Alwadain A, Ragab MG, Alqushaibi A. Cyberbullying detection on social media using stacking ensemble learning and enhanced BERT. Information. 2023;14(8):467. DOI:https://doi.org/10.3390/info14080467

[19] Devi, K. K., & Kumar, G. A. S. (2022). Stochastic Gradient Boosting Model for Twitter Spam Detection. *Computer Systems Science & Engineering, 41*(2), 849-859. https://doi.org/10.32604/csse.2022.020836

[20] Saraswathi, V., Adaikkammai, A., Jebamani, A., Devi, D., & Radhika, R. (2023). Ensemble Learning Models for Detecting Spam Over Social Networks Using RFE. In *International Conference on Advances in Artificial Intelligence and Machine Learning in Big Data*

Processing (pp. 150-164). Cham: Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-43247-7_12

[21] Bharadwaj VY, Likhitha V, Vardhini V, Asritha AU, Dhyani S, Kanth ML. Automated cyberbullying activity detection using machine learning algorithm. E3S Web Conf. 2023;430:01039. DOI: https://doi.org/10.1051/e3sconf/202343001039

[22] Baccouche A, Ahmed S, Sierra-Sosa D, Elmaghraby A. Malicious text identification: deep learning from public comments and emails. Information. 2020;11(6):312. DOI:https://doi.org/10.3390/info11060312

[23] Bello A, Ng SC, Leung MF. A BERT framework to sentiment analysis of tweets. Sensors. 2023;23(2):506. DOI: https://doi.org/10.3390/s23020506

[24] Balasubramanian S, Ganesan P, Rajasekaran J. Weighted ensemble classifier for malicious link detection using natural language processing. Int J Pervasive Comput Commun. 2023. DOI: https://doi.org/10.1108/IJPCC-02-2023-0017

[25] Amer A, Siddiqui T, Athamena B. Detecting cybercrime: An evaluation of machine learning and deep learning using natural language processing techniques on the social network. Res Square. 2022. DOI: https://doi.org/10.21203/rs.3.rs-2184218/v1