International Journal of Safety and Security Engineering
Vol. 14, No. 1, February, 2024, pp. 259-266
Journal homepage: http://iieta.org/journals/ijsse

Android Malware Classification Using Gain Ratio and Ensembled Machine Learning

Dwinanda Bagoes Ansori, Joko Slamet, Muhammad Zakky Ghufron, Muhammad Aidiel Rachman Putra, Tohari Ahmad*

Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya 60111, Indonesia

Corresponding Author Email: tohari@its.ac.id

https://doi.org/10.18280/ijsse.140126

Received: 15 December 2023
Revised: 22 January 2024
Accepted: 29 January 2024
Available online: 29 February 2024

Keywords:
Android malware, ensemble machine learning, gain ratio, information security, network infrastructure, national security, network security, Android security, malware detection

ABSTRACT

Recently, the number of Android users has significantly increased, which has made Android a target for attackers to launch their malicious activities. Malware or malicious code is often embedded in Android apps to gain access to the user's device and retrieve personal data. Researchers have explored various approaches to mitigate the spread of Android malware. Besides, the Android malware dataset has huge dimensions with hundreds of features. Choosing the proper feature selection method is one of the challenges for producing a reliable detection model. This paper proposes an approach to detecting Android malware and classifying it into five categories using gain ratio feature selection and an ensemble machine learning algorithm. Features are reduced based on their importance value through the gain ratio calculation method. Then, features that are considered necessary are included in a classification process that combines many models. Experiment using the CICMalDroid2020 (Canadian Institute for Cybersecurity Malware of Android 2020) dataset shows that the proposed approach can improve detection performance. Gain ratio feature selection improves the detection accuracy in several machine learning classification algorithms, 2.59% in Naïve Bayes, 0.90% in *k*-Nearest Neighbor, and 2.29% in Support Vector Machine. Thus, the ensembled machine learning models of Random Forest, Extra Tree, and *k*-Nearest Neighbors achieved the highest performance, with an accuracy of 94.57% and a precision score of 94.71%.

1. INTRODUCTION

Mobile phones have grown in recent years due to their user-friendly design and multifunctional capabilities, making them an essential asset in people's daily lives [1]. One of the mobile phone operating systems is Android, which was released in 2008 [2]. Android has become the most used operating system on mobile devices, currently holding 70.5% of the total market share by the third quarter of 2023 [3]. The large number of users makes the Android operating system the target of cyber attacks. In the middle of 2020, 10.6 million Android malware were found, and this is expected to increase because of various cases of cybercriminals on mobile devices [4]. Besides, malware is constantly evolving, making it very challenging to detect. The rapid evolution of malware poses a significant threat to individual, commercial, and digital security [5].

Hackers employ reverse engineering techniques to modify and repackage harmless applications by incorporating their malicious code [6]. Malware or malicious code is often embedded in Android apps to gain user device access and retrieve personal data [7]. Thousands of malwares can infiltrate Google Play, the most trusted Android software download and installation service provider [8]. Besides, downloading applications from unknown sources increases the risk of virus and malware infiltration. Children are at a higher risk of being tricked since many harmful programs appear to be safe applications with positive reviews [9]. Thus, using trusted sources to download the application, frequently updating software, and ensuring that security is enabled can enhance the security of Android devices to avoid malware [10].

The rise in malware in Android apps presents a significant challenge. The way to stop the spread of malware is to identify and categorize its types. Previous researchers have introduced various ways to identify and categorize Android malware. Machine learning is the most popular technique for identifying and categorizing Android malware [2, 11, 12]. Prior research has proven that machine learning and deep learning are reliable enough to classify Android malware into five categories [13].

This research focused on feature selection and ensembled classification with five machine learning algorithms: Random Forest (RF), Extra Tree (ET), Naive Bayes (NB), *k*-Nearest Neighbors (*k*-NN), and Support Vector Machine (SVM). The dataset used is CICMalDroid2020, which consists of more than 11 thousand data and 471 features. After preprocessing, the model classified five malware categories: Adware, Banking Malware, SMS Malware, Riskware, and Benign. Therefore, the experiment results show that using different feature selection and classification techniques on large datasets significantly affects the detection performance.

The contribution of this study on the Android malware classification was presented below:

(1) This research focused on classifying five malware categories with ensemble machine learning classification and gain ratio feature selection on the Android malware dataset, CICMalDroid2020.

(2) After preprocessing, the data was trained using different machine learning models: Random Forest (RF), Extra Tree (ET), Naive Bayes (NB), *k*-Nearest Neighbors (*k*-NN), and Support Vector Machine (SVM). The outputs of these models were then combined using the ensembled method.

(3) Analyze the impact of using the gain ratio and ensembled method.

However, previous researchers have proposed numerous malware detection and classification models. Still, studies on Android malware detection focusing on ensembled classification and feature selection using the gain ratio are hard to find. This research conducts a study to see the impact of the gain ratio in each machine-learning technique. Besides, this research also provides the impact of ensemble classification on Android malware detection models.

The paper was divided into the following sections: Section 2 presents relevant research, Section 3 describes the feature selection procedure and the proposed ensemble machine learning model, Section 4 details the experiment's findings, and Section 5 outlines the conclusions.

2. RELATED WORKS

Previous research has focused on developing methods for Android malware analysis. This section discussed previous research that relates to this work. Selvaganapathy et al. [14] conducted a literature review study and summarized the possible attacks and defenses, including methods and future challenges for building effective Android malware detection and classification. Android-based malware classification can be divided into three categories: statistical analysis, dynamic analysis, and hybrid analysis [15].

2.1 Statistical analysis

Statistical analysis detects malicious software by analyzing the application manifest without executing the application [16, 17]. Raghuvanshi et al. [18] machine learning approaches on CICAndMal2017 datasets to identify secure applications or malware. This study got a higher accuracy of 96.27% with the Random Forest Algorithm. On the other hand, Amenova et al. [19] used deep learning algorithms to detect the Android malware. Convolutional neural network (CNN) effectively extract features from the input data, and incorporating supplementary LSTM layers enhances the accuracy of predictions. Experiments using the CICMalDroid2020 dataset show reliable prediction results with 94% accuracy with only a 3% false positive rate.

2.2 Dynamic analysis

Dynamic analysis examines the malware by collecting memory, process, and traffic by running the application through a sandbox or designated environment [16, 17]. Islam et al. [20] presented a dynamic analysis technique. This study showed the impact of outlier handling when used in complex malware dataset. The final prediction combines all trained model outputs, including RF, *k*-NN, MLP, DT, SVM, and LR, whose R2 score was more than 0.85.

2.3 Hybrid analysis

A type of analysis blends static and dynamic elements is known as a hybrid analysis. Taheri et al. [21] presented a two-layer Android malware analysis: Static Binary Classification (SBC) and Dynamic Malware Classification (DMC). The first layer used Permission and Intent features to identify malware, while the second layer used API calls to classify the malware sample from the first layer into four categories and 39 families with a Random Forest Algorithm. Using this method, the model can get a recall value of 61.2%, which has increased by 35.7% compared to previous research.

2.4 Feature selection

In machine and deep learning, implementing feature selection affects performance improvement. Many researchers have proposed numerous feature selection techniques to address this issue. Chakravarty et al. [22] compared three feature selection methods: Gain ratio, Information Gain, and Relief. The proposed method uses feature selection on four different classification algorithms, and the results showed that the gain ratio obtained higher performance in most classification algorithms and achieved 94.47% accuracy. Therefore, the author used the gain ratio in this study because this method gave reliable performance in previous studies.

2.5 Ensemble methods

Ensemble methods are commonly implemented to enhance the precision of malware identification and categorization. This technique combines multiple models and determines the weight of each model's output. This method is often called a voting classifier. Islam et al. [20] presented an effective ensemble machine learning. The study showed that the weighted voting ensemble model performs better than the individual model. However, this research did not implement the gain ratio as the feature selection method. Besides, our research focused on analyzing the impact of gain ratio and ensemble learning.

3. PROPOSED APPROACH

This section discussed the proposed approach for classifying Android malware. Figure 1 illustrates the experimental design of this research. There are three main stages in this study: Data Preprocessing, which consists of Data Scaling and Feature Selection, Classification, and Ensemble voting. The feature selection process eliminated 60% of data columns using the gain ratio technique. This classification resulted in predictions of 5 malware categories: Adware, Banking Malware, SMS Malware, Riskware, and Benign.

3.1 Data preprocessing

The dataset typically includes values with dissimilar units in each column and irrelevant features. These factors can adversely affect the performance of the machine learning model. Thus, data preprocessing is needed to improve the classification performance results. The author used Standard Scaling and Gain Ratio to preprocess the data in this paper.

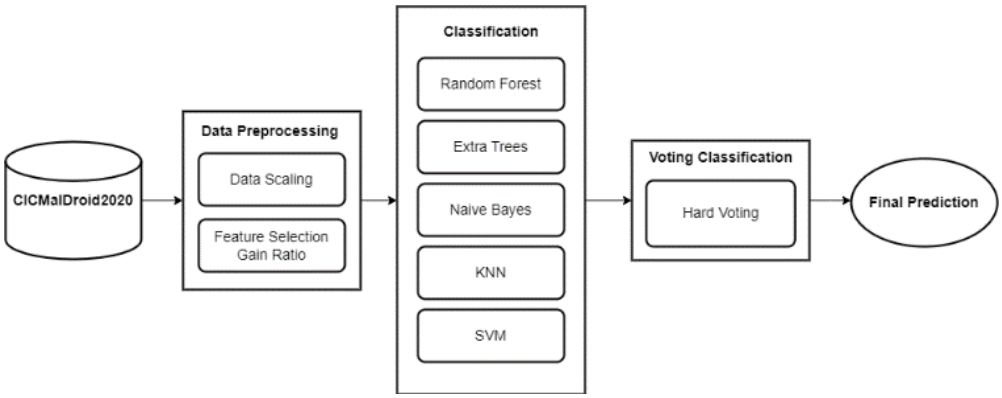

Figure 1. Proposed approach for Android malware classification

3.1.1 Data scaling

Data scaling was one of the most crucial steps in data preprocessing before building a machine learning model. One of the data scaling techniques is standardization, which aims to bind the values between [0, 1] or [-1, 1]. The standardization method used in this research is standard scaler.

A standard scaler is a linear scaler that is very useful for accelerating algorithms using gradient descent [23]. The goal of the standard scaler method is to change features, so it has a mean of zero and a standard deviation of one, as in Eq. (1).

$$x_{scaled} = \frac{x - \mu_x}{\sigma_x} \tag{1}$$

where, μ was the mean and σ was the standard deviation. The formula Eq. (1) is a way to standardize x. Standardization (or z-score normalization) is a common technique in statistics and data analysis. It transforms the values of a variable so that they have a mean of 0 and a standard deviation of 1. The process involves subtracting the mean (μ_x) from each data point (x) to center the distribution around 0. Then, the result is divided by the standard deviation (σ_x) to scale the values, ensuring that they have a consistent unit of measurement.

3.1.2 Gain ratio

This phase focused on reducing the features of the training data. The feature selection process involves identifying and removing less significant features from a dataset to decrease the complexity of machine learning and increase the model accuracy. Previous research on malware detection has proved that the gain ratio performs better than other feature selection methods [22]. Thus, this research also uses the gain ratio method in the Android malware detection model as a feature selection method. The gain ratio is a method that attempts to reduce the bias of information gain by normalizing Information Gain with Information Entropy [24]. For X and Y, Information Gain can be calculated as:

$$IG(X;Y) = Entropy(X) - Entropy(X|Y) \tag{2}$$

$$IG(X;Y) = -\sum_{x\in X} p(x)\log_2\big(p(x)\big) + \sum_{y\in Y} p(y)\sum_{x\in X} p(x|y)\log_2\big(p(x|y)\big) \tag{3}$$

where, $p(x)$ and $p(y)$ represent the probability of x and y class, while $p(x|y)$ is the probability of data x belongs to the class y.

The gain ratio of X compared to Y equals the information gain ratio to the information entropy, which is expressed in Eq. (4) [24]. The gain ratio is defined as the ratio between the mutual information of two random variables and the entropy of one of them. Therefore, the gain ratio (X; Y) falls from 0 to 1. A value of 1 denotes that X leads to Y completely, while 0 signifies complete independence between X and Y [25]. Figure 2 shows ten features with the highest gain ratio score.

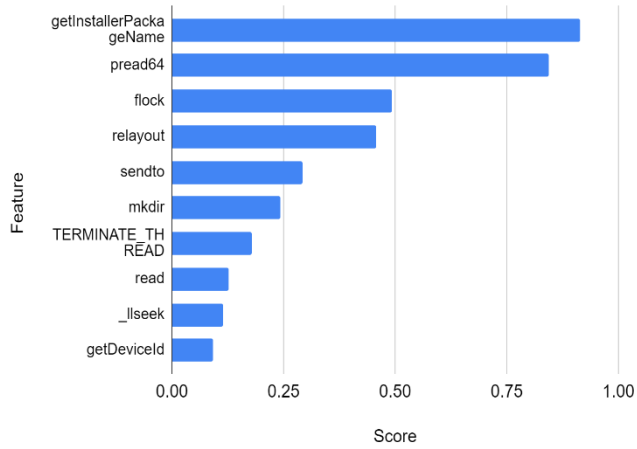

Figure 2. Feature selection using gain ratio

$$Gain\ Ratio(X;Y) = \frac{IG(X;Y)}{Entropy(X)} \tag{4}$$

After feature selection, the dataset was divided into 80% training data (Xtrain, Ytrain) and 20% testing data (Xtest, Ytest).

3.2 Machine learning and ensemble classification

This research aimed to classify five malware categories using five machine learning methods, including Random Forest (RF), Extra Tree (ET), Naïve Bayes (NB), k-Nearest Neighbor (k-NN), and Support Vector Machine (SVM) shown in Figure 1. Random Forest (RF) was an ensemble classifier technique that generates multiple decision trees by randomly selecting subsets of training samples and features [26]. Like Random Forest, the Extra Tree method combined the results of multiple decision trees for classification predictions [27]. Thus, k-Nearest Neighbors (k-NN) classified a new data point by identifying its k-Nearest Neighbors and assigning it to the majority class of those neighbors [28]. On the other hand, SVM aimed to find the maximum margin hyperplane. This decision boundary best separates different classes in the training data [29]. The last is Naive Bayes, which utilizes

probability theory and operates on the assumption that the features considered are independent [30]. RF and ET are samples for a tree-based algorithm, while NB, *k*-NN, and SVM are non-tree-based algorithms. This research uses five different machine learning models to analyze the impact of the gain ratio feature selection in each algorithm. For implementing the machine learning algorithm, this experiment utilizes the Scikit Learn library, and the hyperparameters follow the default values of the library. Thus, the last process combined all the machine learning detection results with an ensemble classification approach.

Ensemble classification was a methodology combining multiple machine learning models rather than relying on a single algorithm. Ensemble algorithms fall under supervised learning, as they can be trained on labeled data and used for making predictions. Combining multiple models in an ensemble represents a collective hypothesis that aims to provide a more robust and accurate prediction than individual models acting alone [31, 32]. This study implemented the ensemble method with a hard voting approach. The majority of the chosen class from the classification determines the result of the hard voting classification. For example, in a scenario where RF, SVM, and *k*-NN predict Riskware, while NB and ET predict Adware, the hard voting result is Riskware as the majority output. This method was applied to 3 and 5 machine learning model combinations, shown in Table 1.

Table 1. Ensembled classification combination

3 Models Combination		
1	***2***	***3***
RF	ET	*k*-NN
RF	ET	SVM
RF	ET	NB
RF	*k*-NN	SVM
RF	*k*-NN	NB
RF	SVM	NB
ET	*k*-NN	SVM
ET	*k*-NN	NB
k-NN	SVM	NB

4. EXPERIMENT AND RESULTS

This section explained the experimental results and analyzed the impact of using a gain ratio and voting classifier for predicting five malware categories. This experiment used Google Colab and was implemented using Python for the proposed model, where the ensemble mechanism used in the proposed mode is shown in Figure 3. To conduct this experiment, the CICMalDroid2020 dataset will be split into two parts with an 80% ratio for training data and a 20% ratio for testing data. Thus, the performance of classification models is analyzed using standard evaluation metrics, such as Accuracy, Precision, Recall, and F1-Score.

4.1 Dataset

CICMalDroid2020 was a public dataset collected from 17,341 Android samples from numerous sources such as VirusTotal, AMD, and MalDozer. There are three big groups of datasets: Statistical information, Dynamic observed behaviors, and network traffic [23, 33]. The dataset consists of 471 features and 11,598 data, divided into five categories: 1) Adware with 1,253 data; 2) Banking Malware with 2,100 data; 3) SMS Malware with 3,904 data; 4) Riskware with 2,546 data; and 5) Benign with 1,795 data [23]. The distribution of each category and data example can be seen in Figures 4 and 5, respectively.

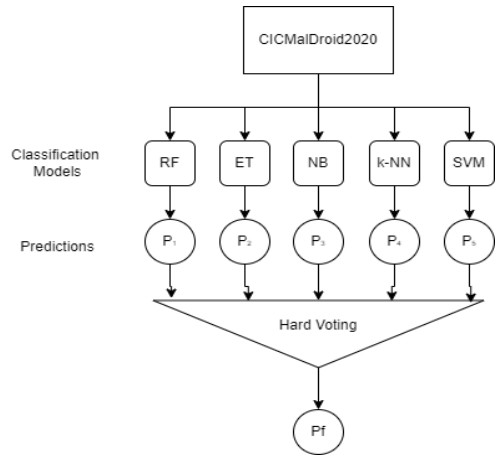

Figure 3. Proposed ensembled classification diagram

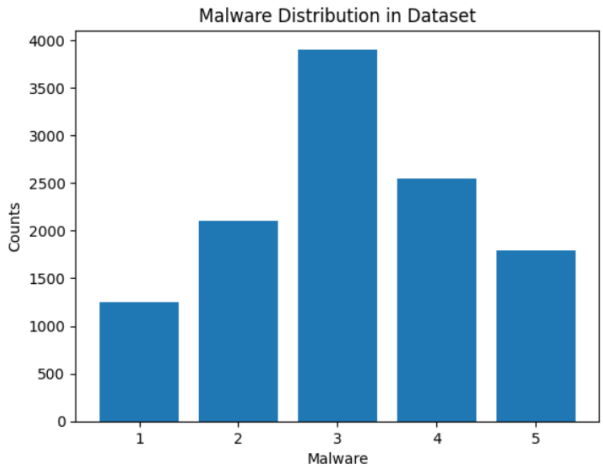

Figure 4. Dataset categories distribution

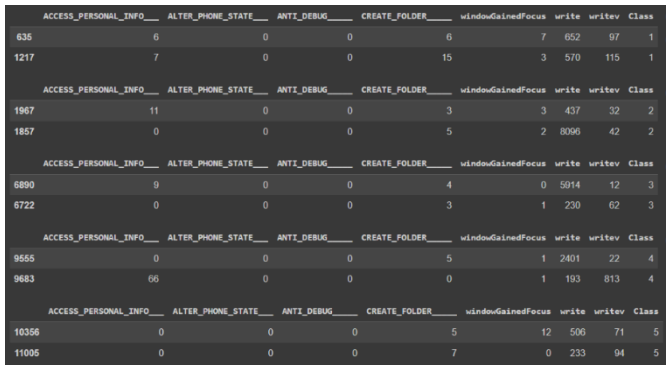

Figure 5. Sample of CICMalDroid2020 dataset

4.2 Performance results on single machine learning with gain ratio

The large number of unimportant features causes increased data redundancy and increases the probability of overfitting. Therefore, feature selection is highly recommended as it significantly affects model training. According to Mahdavifar et al. [23], a zero-gain ratio score means a feature did not

influence identifying malware class. In this experiment, 64% of features with a zero-score of gain ratio were removed, meaning that 171 out of 470 features were effective for the machine learning model. The feature selection results with the ten highest gain ratio scores are shown in Figure 2. Feature selection using Gain Ratio showed better performance with single machine learning models. Figures 6-9 show that Random Forest (RF) and Extra Tree (ET) had the highest performance rate. However, the impact of using gain ratio can be seen very clearly in Naïve Bayes (NB), *k*-Nearest Neighbor (*k*-NN), and Support Vector Machine (SVM). For instance, SVM had 79.05% accuracy without a gain ratio, whereas with a gain ratio applied, it could reach 81.34% accuracy (see Figure 6). It had a 2.29% increase in accuracy, compared to ET and RF, which have slightly different accuracy with gain ratio applied.

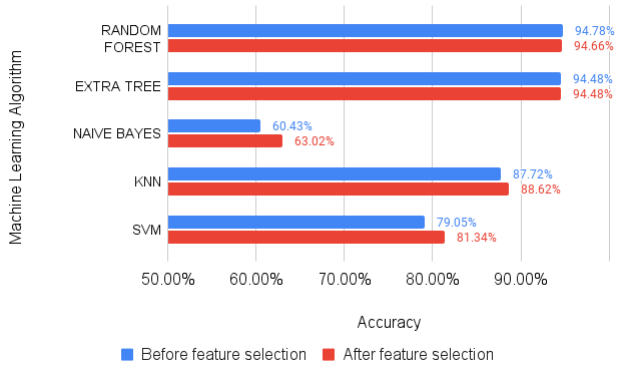

Figure 6. Model accuracy before and after gain ratio

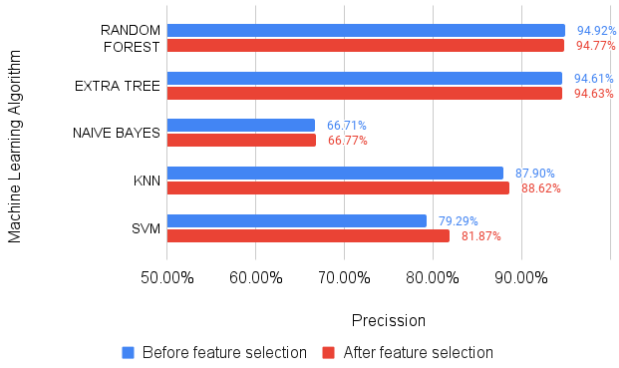

Figure 7. Model precision before and after gain ratio

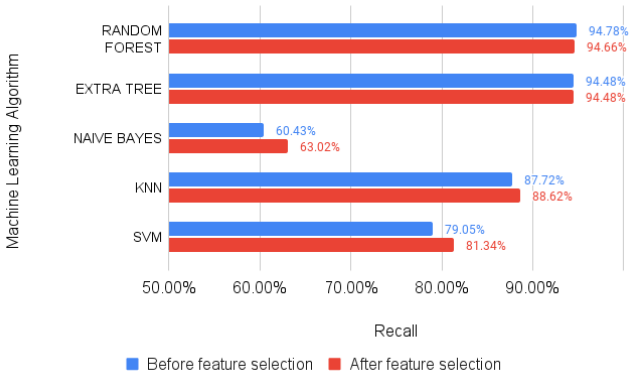

Figure 8. Model recall before and after gain ratio

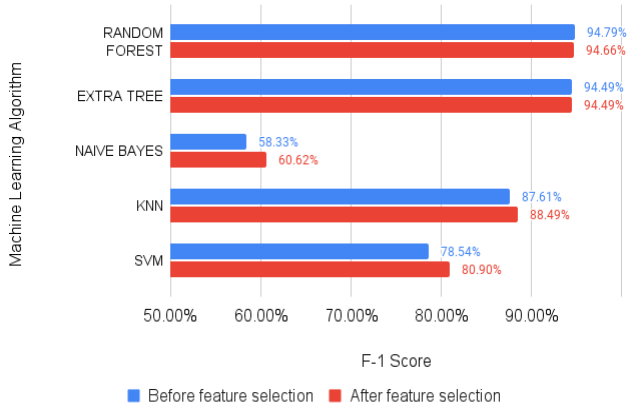

Figure 9. Model f1-score before and after gain ratio

4.3 Performance results on ensemble machine learning

This section discussed the ensemble hard voting classification. Based on Table 2, gain ratio outperformed the ensemble classification. Among the results in single method classification, RF, ET, and *k*-NN, which had the highest performance results, showed the highest accuracy at 94.57%. However, another RF and ET-based voting also resulted in competitive performance compared to other combinations, namely over 94.00% in accuracy, precision, recall, and F1-Score. In addition, the five-combination voting classification got an average of 92.50% precision. The results between the single and ensemble methods are different because the machine learning models with underperformed scores, such as Naive Bayes, have influenced other methods that lead to decreased performance.

Thus, RF, ET, and *k*-NN algorithms show the best detection accuracy of each label. Based on Figure 10, the accuracy of the Adware, Banking, SMS malware, Riskware, and Benign labels is 97.54%, 97.63%, 98.92%, 97.24%, and 97.80%, respectively. Based on the accuracy results of each label, the accuracy of the SMS malware label has the highest value. The model can have the best performance detecting SMS malware because the data from the SMS malware label has the largest amount compared to other labels.

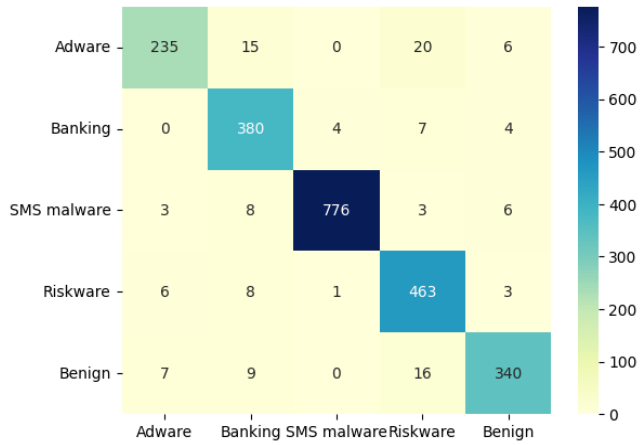

Figure 10. Confusion matrix of RF, ET, k-NN with gain ratio

Table 2. Ensemble voting classification result

Models	Without Gain Ratio				With Gain Ratio			
	Accuracy (%)	*Precision (%)*	*Recall (%)*	*F1-Score (%)*	*Accuracy (%)*	*Precision (%)*	*Recall (%)*	*F1-Score (%)*
RF, ET, *k*-NN	94.22	94.42	94.22	94.22	94.57	94.71	94.57	94.58
RF, ET, SVM	94.40	94.57	94.40	94.41	94.44	94.60	94.44	94.45
RF, ET, NB	94.05	94.33	94.05	94.08	94.35	94.51	94.35	94.37
RF, *k*-NN, SVM	91.34	91.69	91.34	91.37	91.72	91.97	91.72	91.74
RF, *k*-NN, NB	90.13	91.36	90.13	90.30	91.16	91.72	91.16	91.23
RF, SVM, NB	83.23	85.22	83.23	83.25	83.66	84.85	83.66	83.47
ET, *k*-NN, SVM	91.12	91.50	91.12	91.18	91.81	92.06	91.81	91.83
ET, *k*-NN, NB	89.91	91.09	89.91	90.06	91.38	91.88	91.38	91.42
k-NN, SVM, NB	80.39	82.52	80.39	80.38	82.24	83.40	82.24	82.02
RF, ET, *k*-NN, SVM, NB	92.16	92.50	92.16	92.17	92.28	92.40	92.28	92.27

4.4 Comparative analysis

This research compared the proposed method with the model by Nguyen et al. [13] shown in Table 3. The proposed method using gain ratio and ensemble machine learning classification still performs below Nguyen et al. [13]. The result proves that gain ratio feature selection has small impact on the performance compared to Extremely Randomized Trees [13]. The gain ratio calculates the probability of each attribute in the dataset, and this method is unsuitable for a dataset with a huge number of attributes. Besides, the Extremely Randomized Trees method uses a sampling method from the entire dataset while constructing the trees. Different subsets of the data may introduce different biases in the results obtained. Hence, Extra Trees prevents data bias by sampling the entire dataset, so this method is more suitable for use on datasets with many attributes.

Table 3. Model comparison

Models	*Accuracy (%)*	*Precision (%)*	*Recall (%)*	*F1-Score (%)*
Nguyen et al. [13]	97.07	95,50	96.90	95.90
Proposed Model	94.57	94.71	94.57	94.58

5. CONCLUSIONS

Android malware detection is crucial in increasing Android security. Thus, the research proposed a method that combines the gain ratio and ensemble machine learning with five machine learning models: RF, ET, *k*-NN, SVM, and NB. The goal is to detect five Android malware classes: Adware, Banking Malware, SMS Malware, Riskware, and Benign. The experiment result shows that the gain ratio has increased the accuracy of the NB method by 2.59%, *k*-NN by 0.90%, and SVM by 2.29%. RF and ET performed slightly differently after the gain ratio because these machine learning algorithms have a decision tree base and the gain ratio method in default. The combination of RF, ET, and *k*-NN with an ensemble voting classifier achieved the highest performance with an accuracy of 94.57% and a 94.71% precision score. The accuracy score was slightly lower than the highest accuracy in the single RF method, which reached 94.66%. Besides, the ensemble voting classifier has better precision than the single RF method, with 94.66%.

The experiment uses the default values set by the scikit learn library, which means no special phase of hyperparameter tuning. Consequently, the performance does not surpass previous research with similar models or datasets, which can reach 97.07% in RF and 97.67% in ET. Future research analyzes combining machine or deep learning methods to increase ensembled classification performance. Thus, the data preprocessing stage, hyperparameter tuning, and outlier handling need to be provided. However, the application store can use this proposed method for threat mitigation to reduce the likelihood of malicious applications reaching users through official channels.

ACKNOWLEDGMENT

The authors gratefully acknowledge financial support from the Institut Teknologi Sepuluh Nopember for this work, under project scheme of the Publication Writing and IPR Incentive Program (PPHKI) 2024.

REFERENCES

[1] Peng, S., Cao, L., Zhou, Y., Xie, J., Yin, P., Mo, J. (2020). Challenges and trends of android malware detection in the era of deep learning. In 2020 IEEE 8th International Conference on Smart City and Informatization (iSCI). Guangzhou, China, pp. 37-43. https://doi.org/10.1109/isci50694.2020.00014

[2] Liu, K., Xu, S., Xu, G., Zhang, M., Sun, D., Liu, H. (2020). A review of android malware detection approaches based on machine learning. IEEE Access, 8: 124579-124607. https://doi.org/10.1109/access.2020.3006143

[3] Statista Inc. Global market share held by mobile operating system from 2009 to 2023, by quarter. https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/, accessed on Nov. 2, 2023

[4] Hadiprakoso, R.B., Kabetta, H., Buana, I.K.S. (2020). Hybrid-based malware analysis for effective and efficiency android malware detection. In 2020 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS), pp. 8-12. https://doi.org/10.1109/icimcis51567.2020.9354315

[5] Pachhala, N., Jothilakshmi, S., Battula, B.P. (2023). Enhanced malware family classification via image-based analysis utilizing a balance-augmented VGG16 model. Traitement Du Signal, 40(5): 2169-2178. https://doi.org/10.18280/ts.400534

[6] Elsersy, W.F., Feizollah, A., Anuar, N.B. (2022). The rise of obfuscated Android malware and impacts on

detection methods. PeerJ Computer Science, 8: e907. https://doi.org/10.7717/peerj-cs.907

[7] Pan, Y., Ge, X., Fang, C., Fan, Y. (2020). A systematic literature review of Android malware detection using static analysis. IEEE Access: Practical Innovations, Open Solutions, 8: 116363-116379. https://doi.org/10.1109/access.2020.3002842

[8] Cao, M., Ahmed, K., Rubin, J. (2022). Rotten apples spoil the bunch: An anatomy of google play malware. In 2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE), pp. 1919-1931. https://doi.org/10.1145/3510003.3510161

[9] Osman, M.Z., Abidin, A.F.Z., Romli, R.N., Darmawan, M.F. (2021). Pixel-based feature for Android malware family classification using machine learning algorithms. In 2021 International Conference on Software Engineering & Computer Systems and 4th International Conference on Computational Science and Information Management (ICSECS-ICOCSIM), Pekan, Malaysia, pp. 552-555. https://doi.org/10.1109/icsecs52883.2021.00107

[10] Riadi, I., Sunardi, Aprilliansyah, D. (2023). Analysis of Anubis Trojan attack on Android banking application using mobile security labware. International Journal of Safety and Security Engineering, 13(1): 31-38. https://doi.org/10.18280/ijsse.130104

[11] Yuksel, A.K., Ar, Y. (2023). A machine learning approach to malware detection using application programming interface calls (MDAPI). Traitement Du Signal, 40(4): 1511-1520. https://doi.org/10.18280/ts.400419

[12] Rasheed, M.M., Faieq, A.K., Hashim, A.A. (2020). Android botnet detection using machine learning. Ingénierie des Systèmes d'Information, 25(1): 127-130. http://dx.doi.org/10.18280/isi.250117

[13] Nguyen, C.D., Khoa, N.H., Doan, D, Cam, N.T. (2023). Android malware category and family classification using static analysis. In 2023 International Conference on Information Networking (ICOIN), Bangkok, Thailand, pp. 162-167. https://doi.org/10.1109/ICOIN56518.2023.10049039

[14] Selvaganapathy, S., Sadasivam, S., Ravi, V. (2021). A review on Android malware: Attacks, countermeasures and challenges ahead. Journal of Cyber Security and Mobility, 10(1): 177-230. https://doi.org/10.13052/jcsm2245-1439.1017

[15] Amer, E. (2021). Permission-based approach for Android malware analysis through ensemble-based voting model. In 2021 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC), Cairo, Egypt, pp. 135-139. https://doi.org/10.1109/miucc52538.2021.9447675

[16] Musikawan, P., Kongsorot, Y., You, I., So-In, C. (2023). An enhanced deep learning neural network for the detection and identification of Android malware. IEEE Internet of Things Journal, 10(10): 8560-8577. https://doi.org/10.1109/jiot.2022.3194881

[17] Zelinka, I., Amer, E. (2019). An ensemble-based malware detection model using minimum feature set. Mendel, 25(2): 1-10. https://doi.org/10.13164/mendel.2019.2.001

[18] Raghuvanshi, P., Singh, J.P. (2022). Android malware detection using machine learning techniques. In 2022 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, pp. 1117-1121. https://doi.org/10.1109/CSCI58124.2022.00200

[19] Amenova, S., Turan, C., Zharkynbek, D. (2022). Android malware classification by CNN-LSTM. In 2022 International Conference on Smart Information Systems and Technologies (SIST), Nur-Sultan, Kazakhstan, pp. 1-4. https://doi.org/10.1109/sist54437.2022.9945816

[20] 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.2023.03.001

[21] Taheri, L., Kadir, A.F.A., Lashkari, A.H. (2019). Extensible Android malware detection and family classification using network-flows and API-calls. In 2019 International Carnahan Conference on Security Technology (ICCST), Chennai, India, pp. 1-8. https://doi.org/10.1109/ccst.2019.8888430

[22] Chakravarty, S. (2020). Feature selection and evaluation of permission-based Android malware detection. In 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI) (48184), Tirunelveli, India, pp. 795-799. https://doi.org/10.1109/ICOEI48184.2020.9142929

[23] Mahdavifar, S., Abdul Kadir, A.F., Fatemi, R., Alhadidi, D., Ghorbani, A.A. (2020). Dynamic Android malware category classification using semi-supervised deep learning. In 2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, International Conferences on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), Calgary, AB, Canada, pp. 515-522. https://doi.org/10.1109/dasc-picom-cbdcom-cyberscitech49142.2020.00094

[24] Tateno, S., Ichiyama, M., Yahiro, K., Matsuyama, H., Oshima, E. (2012). Development of corrosion rates estimation method for CUI using information gain ratio. In 2012 12th International Conference on Control, Automation and Systems, Jeju, Korea (South), pp. 803-807.

[25] Feng, J., Niu, X., Zhang, J., Wang, J.H. (2022). Gene selection and classification of scRNA-seq data combining information gain ratio and genetic algorithm with dynamic crossover. Wireless Communications and Mobile Computing, 2022: 9639304. https://doi.org/10.1155/2022/9639304

[26] Chanal, D., Steiner, N.Y., Chamagne, D., Pera, M.C. (2021). Impact of standardization applied to the diagnosis of LT-PEMFC by Fuzzy C-Means clustering. In 2021 IEEE Vehicle Power and Propulsion Conference (VPPC), Gijon, Spain, pp. 1-6. https://doi.org/10.1109/vppc53923.2021.9699234

[27] Majidi, S.H., Hadayeghparast, S., Karimipour, H. (2022). FDI attack detection using extra trees algorithm and deep learning algorithm-autoencoder in smart grid. International Journal of Critical Infrastructure Protection, 37(100508): 100508. https://doi.org/10.1016/j.ijcip.2022.100508

[28] Dong, X., Liang, Y., Miyamoto, S., Yamaguchi, S. (2023). Ensemble learning based software defect prediction. Journal of Engineering Research, 11(4): 377-391. https://doi.org/10.1016/j.jer.2023.10.038

[29] Blanco, V., Japón, A., Puerto, J. (2022). A mathematical programming approach to SVM-based classification with label noise. Computers & Industrial Engineering, 172(108611): 108611. https://doi.org/10.1016/j.cie.2022.108611
[30] Vu, D.H. (2022). Privacy-preserving Naive Bayes classification in semi-fully distributed data model. Computers & Security, 115(102630): 102630. https://doi.org/10.1016/j.cose.2022.102630
[31] Ganaie, M.A., Hu, M., Malik, A.K., Tanveer, M., Suganthan, P.N. (2022). Ensemble deep learning: A review. Engineering Applications of Artificial Intelligence, 115(105151): 105151. https://doi.org/10.1016/j.engappai.2022.105151
[32] Luo, S., Gu, Y., Yao, X., Fan, W. (2021). Research on text sentiment analysis based on neural network and ensemble learning. Revue d Intelligence Artificielle, 35(1): 63-70. https://doi.org/10.18280/ria.350107
[33] Mahdavifar, S., Alhadidi, D., Ghorbani, A.A. (2022). Effective and efficient hybrid Android malware classification using pseudo-label stacked auto-encoder. Journal of Network and Systems Management, 30: 1-34. https://doi.org/10.1007/s10922-021-09634-4

NOMENCLATURE

x	Spesific data/value from feature
y	Specific data/value from target feature
μ	Mean
σ	Standard deviation
X	Dataset feature
Y	Dataset target feature
p	Probability
gain ratio	Ratio between the mutual information of two random variables and the entropy of one of them
Entropy	Uncertainty or randomness in the distribution of class labels in a data set.
IG	Measures the effectiveness of certain attributes in reducing uncertainty (entropy) in a dataset
$\sum_{x \in X} x$	Sum of x as an element of X