International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

Classification of Malicious Android Applications Using Naive Bayes and Support Vector Machine Algorithms

Abdullah Batuhan Yilmaz[1], Yavuz Selim Taspinar[2], Murat Koklu[3*]

Submitted: 29/04/2022 Accepted : 23/05/2022

Abstract: As the use of smart devices increases, the number of malicious software is also increasing day by day. Android is the most used operating system in smart devices. That's why there is a lot of malware targeting this platform. By examining the permission properties of malicious software, it can be determined whether it is malicious or not. However, this is a complex problem. In order to solve this problem, in this study, classification processes have been carried out to determine whether the software is harmful with machine learning methods. For this purpose, a dataset containing 2854 malicious software and 2870 harmless software was created. In the dataset, there are 116 permission features for each software and a class feature that indicates whether it is malicious or not. Using these features, Support Vector Machine (SVM) and Naïve Bayes (NB) models were trained. The 10-fold cross validation method was used in training and testing processes. Accuracy, F-1 Score, precision, recall and specificity metrics were used to analyze the performances of the models. ROC curve and AUC values were used to analyze the learning and prediction levels of the models. As a result of the tests of the models, 90.9% classification success was obtained from the SVM model and 92.4% from the NB model.

Keywords: *Classification, machine learning, malware detection, naïve bayes, support vector machine*

1. Introduction

Today, technology is developing very rapidly and this development is directly related to many areas of life. In parallel with technological developments, developments in mobile applications continue rapidly. The most important reason for these developments is that almost all transactions that can be carried out via computers can now be carried out with mobile devices. At the beginning of the preference of mobile devices are the dimensions and the advantages of the dimensions at the point of transportation. It has become very popular with attackers with the increase in processing capacities and the increasing interest in devices. Attackers always prefer to attack areas of intense interest to people. If a person or institution is not targeted in the cyber world, the aim is to deliver the pest to the most people. When the mobile operating systems used worldwide are examined, the most used operating systems are Android and IOS. While these two operating systems account for 98.3% of the total usage in the world, other operating systems have a usage rate of only 1.7%. The Android operating system, which has a usage rate of 70 percent, comes first, and IOS comes in the second place with a rate of 28.3 percent [1]. In areas where antivirus software is insufficient, machine learning algorithms have become very important in order to ensure that mobile platforms, which can run increasingly complex software, can become more protected against malicious software. Since the security systems to be developed can update themselves dynamically, they increase the security in the cyber world by producing more successful results against the attackers and the pests developed by the attackers [2]. Xu et al. (2016) suggested in their study that existing malware detection methods are configured according to permissions and API requests, and that although these methods capture interactions between mobile applications and Android systems, the communication occurring within the application boundaries is not taken into account, therefore it is insufficient to detect malicious software that does not need suspicious resources. For this reason, a detection model has been revealed by working on 5,264 malware and 12,026 harmless software. They claimed that the proposed system achieved a 10% higher success rate compared to permission-based systems. They achieved an error rate of 0.67% and a success rate of 97.4%. As a result of the manual analysis of the applications that gave false results, they determined that there were 43 new malware from the harmless dataset. However, they found that the number of faulty applications was reduced to seven, and the developed system detected 1,708 more advanced malware than the signature-based benchmark system, and it could not detect 220 malware, which is malicious software [3]. Arslan et al. carried out a study on the formation of security models with permission-based systems of mobile applications. The risk assessment was carried out on the identification of data permissions of malicious software and on requesting additional permissions. They used static analysis and code analysis for these operations. They conducted a study on the creation of a risk value for applications that are considered harmful with the data obtained [4]. Bikmaz [5] worked on measuring security awareness of students in mobile applications, ensuring that they are aware of malware and software used for security purposes on smart phones, and taking warnings into account during

[1] Graduate School of Natural and Applied Science, Selcuk University, Konya, Türkiye, ORCID ID : 0000-0002-4400-712X
[2] Doganhisar Vocational School, Selcuk University, Türkiye ORCID ID : 0000-0002-7278-4241
[3] Department of Computer Engineering, Selcuk University, Türkiye ORCID ID : 0000-0002-2737-2360
** Corresponding Author Email: mkoklu@selcuk.edu.tr*

the downloading processes of applications. Feizollah et al. [6] in their study, they argued that Android systems are used by a very large audience, and malware targeting these systems is increasing at the same rate, and this is a natural consequence of Android's facilitating the inclusion of third-party applications. In the examinations made on a data set containing 7406 applications, 1846 harmless and 5560 malicious, they performed a malware detection study, which achieved a success rate of 91% over permissions. Jörgensson (2018) collected survey data in order to learn about user behavior due to the fact that malicious software is constantly increasing, and recently, unusual types of malware have begun to appear, and that prevention systems are insufficient. With this data, he made a study on the examination of malicious software that emerged in the previous years and after. Utku et al. argued that the usage rates of mobile devices are increasing gradually, the increase in pests such as increased use, and the increasing pests endanger personal data. In order to increase the adequacy of existing security measures in preventing malicious software, they have worked on the detection of Android-based malware [7]. Odusami et al. stated that one of the biggest security problems for smartphones is malware. They conducted a study in which machine learning gave high detection results for the detection of malware [8]. In her study, Barbieru conducted a study aiming to establish a platform for the analysis of cyber security solutions, the diversity of smart mobile devices, and the analysis of mobile malware for the protection of devices due to the increase in cyber attacks affecting organizations worldwide [9]. Sapalo Sicato et al. studied the development and security of smart home technologies, malware attacks, potential risks on smart home systems, and defense mechanisms [10]. Vasan et al. conducted a study to detect the types of malware groups and to detect malware using deep learning architecture, where the types and complexity of malicious software increase [11]. Tahtaci and Canbay worked on the scanning of malicious software by using machine learning algorithms and the creation of models using the properties of financial files [12]. Altan [13] has successfully detected attacks on IoT-based systems with deep neural networks and autoencoders. Altan et al. proposed a clustering-based method to detect problems in wireless sensor networks and to ensure network security [14].
The procedures performed in this study are as follows:

- A dataset containing 2854 malicious software and 2870 harmless software was created.
- Training of SVM and NB models was carried out using the data in the dataset.
- Performance evaluations of the obtained classification models were carried out.

The rest of the work is planned as follows. In section 2, the dataset, machine learning algorithms and materials used in performance evaluation are explained. In section 3, the experimental results obtained as a result of the training and testing of the models are given. In section 4, the results and discussions of the study are given.

2. Material and Methods

2.1. Data Acquisition and Android Malware Dataset (AMD)

The files in the dataset were provided by us for this study from the antivirus company named Zeman. The created dataset contains data of 2854 malicious applications and 2870 non-malicious applications. In this dataset, malicious and non-malicious android applications have also been confirmed from websites such as Virustotal and VirusShare [15, 16]. This data is not downloadable by anyone, in order to prevent the malware's developers from achieving their goals. For this reason, a special authorization has been obtained for malicious software by communicating with the owners of this website via e-mail, explaining the work to be done with the corporate e-mail account, and specifying the purpose for which the malicious software will be used. The stages of obtaining the dataset are shown in Figure 2.

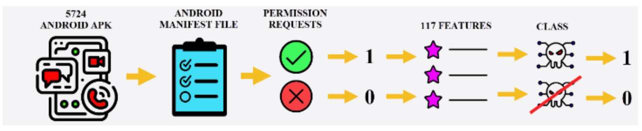


Fig. 1. Data acquisition

All data in the dataset is binary. A value of 0 for 117 features indicates that the application does not request permission for the current feature, and a value of 1 indicates that it requests permission. For the class property, which indicates whether the application is harmful, a value of 0 indicates that the application is harmless, and a value of 1 indicates that it is harmful. Table 1 shows the permission properties of the applications in the dataset.

Table 1. Permission features of android APK

INTERNET	USE_CREDENTIALS	KEEP_SCREEN_ON	WRITE_EXTERNAL_STORAGE
MOUNT_FORMAT_FILESYSTEMS	BROADCAST_SMS	READ_OWNER_DATA	READ_PHONE_STATE
READ_CONTACTS	FLASHLIGHTGHARDWARE_C.	WRITE_CALL_LOG	RECEIVE_BOOT_COMPLETED
RECEIVE_SMS	CHANGE_WIFI_STATE	READ_DATAGPERSONAL_INFOL	ACCESS_WIFI_STATE
READ_SMS	WRITE_MEDIA_STORAGE	DELETE_PACKAGES	SEND_SMS
CALL_PHONE	BIND_WALLPAPERTNTB	CD_MESSAGE	SYSTEM_ALERT_WINDOW
MOUNT_UNMOUNT_FILESYSTEMS	READ_SYNC_SETTINGS	WRITE_SYNC_SETTINGS	GET_TASKS
PROCESS_OUTGOING_CALLS	CHANGE_CONFIGURATION	RECEIVE_USER_PRESENT	BIND_DEVICE_ADMIN
BIND_WALLPAPER	READ_PROFILE	WRITE_PROFILE	ACCESS_NETWORK_STATE
READ_EXTERNAL_STORAGE	RECEIVE_MMS	NETWORK	READ_SETTINGS
WRITE_SETTINGS	SET_ACTIVITY_WATCHER	ACCESS_DOWNLOAD_MANAGER	ACCESS_COARSE_LOCATION
ACCESS_FINE_LOCATION	ACCESS_ALL_DOWNLOADS	WRITE_INTERNAL_STORAGE	CAMERA
FLASHLIGHT	BAIDU_LOCATION_SERVICE	WRITE_CONTACTS	VIBRATE
WAKE_LOCK	BATTERY_STATUS	BROADCAST_WAP_PUSHF	DISABLE_KEYGUARD
GET_ACCOUNTS	EXPAND_STATUS_BAR	WRITE_APN_SETTINGS	CHANGE_NETWORK_STATE
WRITE_SMS	CONNECTIVITY_INTERNAL	HARDWARE_TEST	SET_WALLPAPER
SET_WALLPAPER_HINTS	READ_CALL_LOG	LOCATION	MODIFY_AUDIO_SETTINGS
RECORD_AUDIO	ACCESS_MOCK_LOCATION	CHANGE_WIFI_MULTICAST_STATE	READ_CALENDAR
BROADCAST_STICKY	SIGNAL_PERSISTENT_PROCESSES	UA_DATAL	BIND_VPN_SERVICE
MAPS_RECEIVEL	INTERNETGCMG	BIND_PRINT_SERVICE	BIND_NOTIFICATION_LISTENER_SERVICE
BATTERY_STATS	UA_DATA	MODIFY_PHONE_STATE	BLUETOOTH
RESTART_PACKAGES	ACCESS_COARSE_UPDATES	READ_MESSAGING_EXTENSION_DATA	NFC
BIND_ACCESSIBILITY_SERVICE	SEND_MESSAGES	BIND_REMOTEVIEWS	GET_PACKAGE_SIZE
KILL_BACKGROUND_PROCESSES	INTERACT_ACROSS_USERS_FULL	INSTALL_PACKAGES	BLUETOOTH_ADMIN
RAISED_THREAD_PRIORITY	AUTHENTICATE_ACCOUNTS	SET_DEBUG_APP	ACCESS_SUPERUSER
BIND_INPUT_METHOD	NETWORK_STATE	CLEAR_APP_CACHE	BROADCAST_WAP_PUSH
REORDER_TASKS	GET_CONTACTS	SEND_RESPOND_VIA_MESSAGE	RECEIVE_WAP_PUSH
PERSISTENT_ACTIVITY	WAKE_LOCKC	READ_LOGS	CD_MESSAGEL
ACCESS_LOCATION_EXTRA_COM.	MANAGE_ACCOUNTS	WRITE_SECURE_SETTINGS	VIBRATEGCMGS
GET_ACCOUNTSK			

2.2. Naive Bayes (NB)

Naive Bayes algorithm is based on Bayes theorem. Bayes' theorem expresses the conditional computation formula discovered by Thomas Bayes. This algorithm has very high success rates on results with two or more levels. The need for training data is considerably less than with logistic regression when the independence assumption is valid. Despite the assumption of independence, the variables are generally not independent from the real world [17]. The classifier is used more widely than complex classification methods because it gives very good results [18]. The Naïve Bayes algorithm, which is a subclass of machine learning, performs operations for the classes to be classified and the data that it knows to which classes the sample data belong to. During this work, while the android software, which is harmful and harmless, represents two classes in our article, the classifier that learns over the data set will decide whether the new future software is harmful or not.

In the operations for this classifier;

- P(A|B); The probability of occurrence of state A when situation B occurs,
- P(B|A); The probability that condition B will occur when condition A occurs,
- P(A); The probability of occurrence of A,
- P(B); The probability of occurrence of B.

It is known that the Naïve Bayes algorithm produces very good results with a small number of learning data. In addition to this advantage, it does not produce results during the study for a situation that is not included in the learning data. For this reason, it is important to see possible scenarios in the training data. Naïve Bayes algorithm focuses on the different effects of different attributes on classification in order to detect whether Android software is malicious or not [19].

2.3. Support Vector Machine (SVM)

One of the classification algorithms, Support Vector Machine, aims to find the line at the best point separating the two classes. Introduced by Vapnik, SVM has become one of the most popular machine learning techniques over time. SVM is a very powerful method for estimating values that are not included in the training set. With this feature, it can be successfully applied in complex classification applications [20].

2.4. k-Fold Cross validation

k-fold cross validation is the method used for segmentation of the data set for training the model by performing the evaluation processes of the models in classification processes [21]. Cross-validation is a way of evaluating learning algorithms by segmenting data and comparing them in addition to these evaluations [22, 23]. In cross validation, the training and validation sets are successively crossed over so that each data point has a chance to be validated. The basic form of cross validation is k-fold cross validation [24]. Due to its versatility and functionality in data mining applications, k-fold cross validation is one of the most used methods for model selection and error estimation of classifiers [25]. Figure 2 shows the general representation of the k-fold cross validation method used in the study.

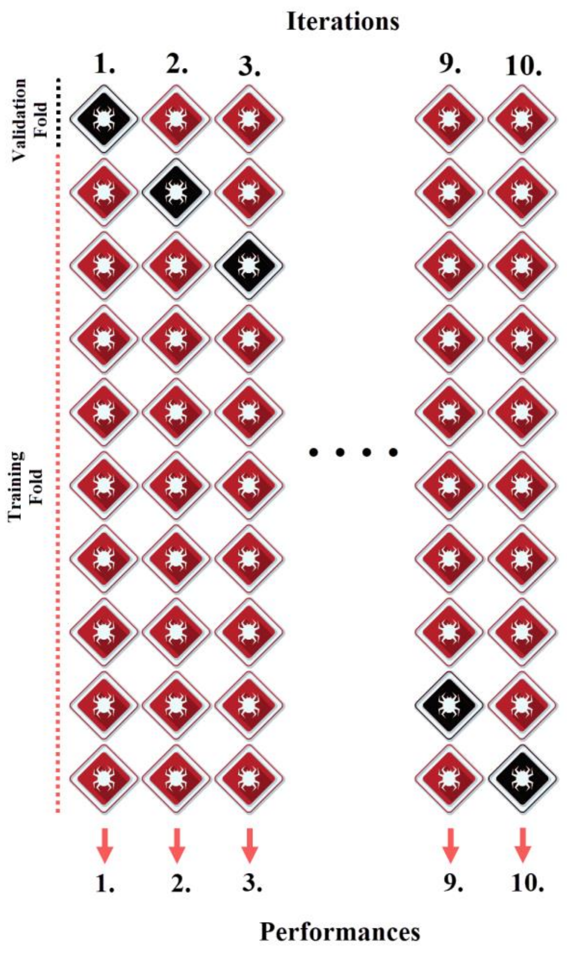


Fig. 2. k-fold cross validation in this study

A cross validation with a k value of 10 is shown in Figure 2. In this method, the dataset is divided into k parts. In each iteration, k-1 pieces are used for training the models, while one piece is used for testing. As a result of k number of iterations, k number of results are obtained. The average performance metrics of the models can be calculated by taking the arithmetic average of the results obtained [26].

2.5. Confusion Matrix

These are the tables used to evaluate the performance of classification models. Thanks to the data in these tables, the performance metrics of the models can be calculated [27, 28]. There are four different values obtained from the models on the confusion matrix. These values are TP, TN, FP and FN [29, 30]. In the study, these values are expressed as follows.

- TP: Classification of malware as malicious
- TN: Classification of harmless software as harmless
- FP: Classification of harmless software as malicious
- FN: Classification of malware as harmless

2.6. Performance Metrics

Performance metrics are used to evaluate and compare the performance of machine learning models [31-33]. Accuracy, precision, recall and F-1 Score metrics were used in the study. The formulas of these metrics are shown in Table 2.

Table 2. Performance metrics formulas used in the study

Accuracy (ACC)	(TP+TN) / (TP+TN+FP+FN)
F1-Score (FSC)	(2*TP) / (2*TP+FP+FN)
Precision (PSC)	TP / (TP+FP)
Recall (RCL)	TP / (TP+FN)
Specificity (SPC)	TN / (TN+FP)

ROC (Receiver Operating Characteristic) curve and AUC (Area under the ROC Curve) values are used to evaluate machine learning models. While the ROC curve shows the predictive capacity of the model, the AUC value can provide information about the learning level of the model [34, 35].

3. Experimental Results

By using 116 features in Andorid Malware Dataset, classifications were carried out to detect whether applications are malware or non-malware. Experiments were made on a computer with Intel® i5® 10200H CPU @2.40 Ghz and NVIDIA GTX1650Ti GPU and 24 GB RAM. Classification models were determined as k=10 in the cross validation method for SVM and NB training. The training of the models was carried out using all the features in the dataset, with 116 inputs and 1 class feature. The flow chart of the processes from data collection to performance evaluation of models is shown in Figure 3.

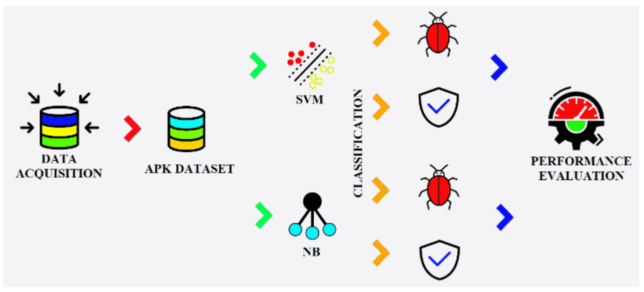
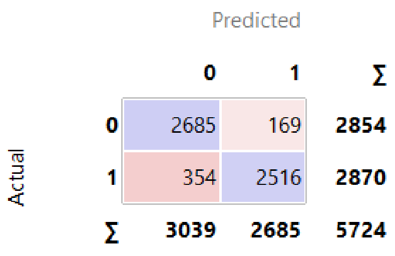


Fig. 3. Flow chart of experiments

In the training of the SVM model, the kernel function was determined as RBF (Radial Basis Function), numerical tolerance 0.0010, iteration 100. As a result of the tests of the trained SVM model, the confusion matrix in Figure 4 was obtained.

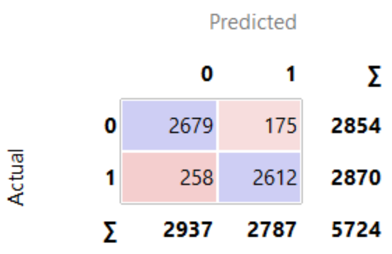


Predicted

Actual	0	1	∑
0	2685	169	2854
1	354	2516	2870
∑	3039	2685	5724

Fig. 4. Confusion matrix of SVM

When Figure 4 is examined, it is seen that 2685 of the non-malware (0) applications are classified correctly. It is seen that 2516 of the applications with Malware (1) are classified correctly. 354 of the malware applications are classified as non-malware. 169 of the non-malware applications were classified as malware. Using these values, the performance metrics of the SVM model were calculated. Performance metrics of the SVM model are shown in Table 3.



Predicted

Actual	0	1	∑
0	2679	175	2854
1	258	2612	2870
∑	2937	2787	5724

Fig. 5. Confusion matrix of NB

When Figure 5 is examined, it is seen that 2679 of the non-malware (0) applications are classified correctly. It is seen that 2512 of the applications with malware (1) are classified correctly. 258 of the malware applications are classified as non-malware. 175 of the non-malware applications were classified as malware. Using these values, the performance metrics of the NB model were calculated. The performance metrics of the NB model are shown in Table 3.

Table 3. Performance metrics of SVM and NB models (%)

	ACC	FSC	PSC	RCL	SPC	AUC
SVM	90.9	90.9	91.0	90.9	90.9	96.9
NB	92.4	92.4	92.5	92.4	92.4	97.7

According to Table 3, it is seen that the classification success of the NB model is higher than the SVM model. Parallel to the Accuracy metric, higher success was achieved in other performance metrics than the NB model. When the AUC values of the models are compared, it is seen that the AUC value of the NB model is higher. ROC curves and AUC values are important criteria in examining the learning capacities of models. Figure 6 shows the ROC curve of the SVM model, and Figure 7 shows the ROC curve of the NB model.

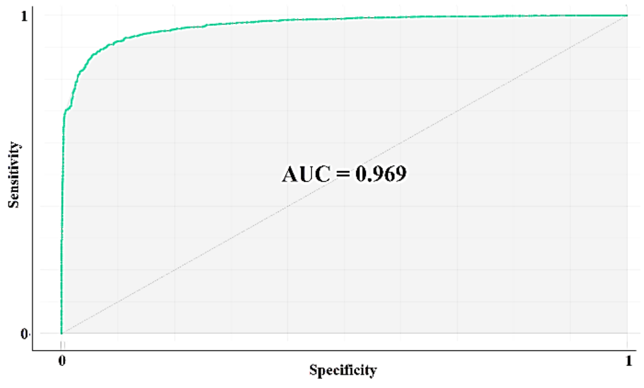


Fig. 6. ROC curve of SVM

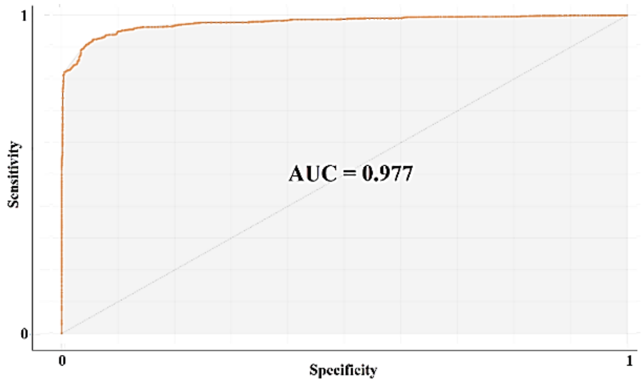


Fig. 7. ROC curve of NB

According to the ROC curve and AUC values in Figure 6 and Figure 7, it can be observed that the NB model can classify more successfully because better learning occurs. The AUC value can take a value in the range of 0-1 and it can be said that the learning capacity of the models increases as it approaches the value of 1.

4. Conclusions

In this study, using the data of 2854 malicious applications and 2870 non-malicious applications used in the Android operating system, classification processes were carried out with machine learning methods. SVM and NB methods were used to classify the data. A 10-fold cross validation was used to carry out detailed performance evaluations. As a result of the tests performed, confusion matrices were used to compare the performances of the classification models. Performance metrics of both models were calculated using the data on the confusion matrix. As a result of the classifications, 90.9% classification success was obtained from the SVM model and 92.4% from the NB model. The F-1 Score value of the SVM model is 90.9%, the precision value is 91%, the recall value is 90.9%, and the specificity value is 90.9%. The F-1 Score value of the NB model is 92.4%, the precision value is 92.5%, the recall value is 92.4%, and the specificity value is 92.4%. The performances of the models were evaluated with ROC curves and AUC. When all metrics are examined, it can be said that the NB model distinguishes between malware and non-malware more successfully. It is possible to make more successful antivirus software by using the recommended models in malware detection. In future studies, the dataset will be expanded and malware detection will be made with different machine learning methods. It is planned to design an interface so that the models to be created can be used by users.

Author contributions

Abdullah Batuhan Yilmaz: Conceptualization, Methodology, Software, Field study **Yavuz Selim Taspinar:** Data curation, Writing-Original draft preparation, Software, Validation., Field study **Murat Koklu:** Visualization, Investigation, Writing-Reviewing and Editing.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] Aytekin, A., et al. *Mobil cihazları etkileyen zararlı yazılımlar ve korunma yöntemleri*. 2019. International Social Research and Behavioral Sciences Symposium.

[2] Liu, K., et al., A review of android malware detection approaches based on machine learning. IEEE Access, 2020. 8: p. 124579-124607.

[3] Xu, K., Y. Li, and R.H. Deng, *Iccdetector: Icc-based malware detection on android.* IEEE Transactions on Information Forensics and Security, 2016. 11(6): p. 1252-1264.

[4] Arslan, R.S., İ.A. Doğru, and N. Barışçı, *Android Mobil Uygulamalar için İzin Karşılaştırma Tabanlı Kötücül Yazılım Tespiti.* Politeknik Dergisi, 2017. 20(1): p. 175-189.

[5] Bıkmaz, Z., Sağlık Yönetimi Bölümü Öğrencilerinin Mobil Güvenlik Farkındalığı ve Dijital Veri Güvenliği Farkındalıklarının Belirlenmesi. Uluslararası Yönetim Bilişim Sistemleri ve Bilgisayar Bilimleri Dergisi, 2017. 1(1): p. 22-30.

[6] Feizollah, A., et al., Androdialysis: Analysis of android intent effectiveness in malware detection. computers & security, 2017. 65: p. 121-134.

[7] Utku, A., I.A. Dogru, and M.A. Akcayol. Permission based android malware detection with multilayer perceptron. in 2018 26th Signal Processing and Communications Applications Conference (SIU). 2018. IEEE.

[8] Odusami, M., et al. Android malware detection: A survey. in International Conference on Applied Informatics. 2018. Springer.

[9] Barbieru, D., *Platforma Software Integrata Pentru Analiza Malware a Terminalelor Mobile.* Buletinul Universității Naționale de Apărare „Carol I", 2019. 6(3): p. 37-46.

[10] Sapalo Sicato, J.C., et al., *VPNFilter malware analysis on cyber threat in smart home network.* Applied Sciences, 2019. 9(13): p. 2763.

[11] Vasan, D., et al., IMCFN: Image-based malware classification using fine-tuned convolutional neural network architecture. Computer Networks, 2020. 171: p. 107138.

[12] Tahtacı, B. and B. Canbay. Android Malware Detection Using Machine Learning. in 2020 Innovations in Intelligent Systems and Applications Conference (ASYU). 2020. IEEE.

[13] Altan, G., SecureDeepNet-IoT: A deep learning application for invasion detection in industrial Internet of Things sensing systems. Transactions on Emerging Telecommunications Technologies, 2021. 32(4): p. e4228.

[14] Turgut, İ.A. and G. Altan, A fully distributed energy-aware multi-level clustering and routing for WSN-based IoT. 2021.

[15] *Virus Share*. 2022 [cited 2022 10 March]; Available from: https://virusshare.com.

[16] 16. *Virus Total*. 2022 [cited 2022 10 March]; Available from: https://virustotal.com.

[17] Alanazi, A., *Using machine learning for healthcare challenges and opportunities.* Informatics in Medicine Unlocked, 2022: p. 100924.

[18] Vembandasamy, K., R. Sasipriya, and E. Deepa, *Heart diseases detection using Naive Bayes algorithm.* International Journal of Innovative Science, Engineering & Technology, 2015. 2(9): p. 441-444.

[19] Shang, F., et al., Android malware detection method based on naive Bayes and permission correlation algorithm. Cluster Computing, 2018. 21(1): p. 955-966.

[20] Ali, W., Hybrid intelligent android malware detection using evolving support vector machine based on genetic algorithm and particle swarm optimization. IJCSNS, 2019. 19(9): p. 15.

[21] Koklu, M. and Y.S. Taspinar, Determining the Extinguishing Status of Fuel Flames With Sound Wave by Machine Learning Methods. IEEE Access, 2021. 9: p. 86207-86216.

[22] Al-Doori, S.K.S., Y.S. Taspinar, and M. Koklu, *Distracted Driving Detection with Machine Learning Methods by CNN Based Feature Extraction.* International Journal of Applied Mathematics Electronics and Computers, 2021. 9(4): p. 116-121.

[23] Bicakci, M., et al., *Metabolic imaging based sub-classification of lung cancer.* IEEE Access, 2020. 8: p. 218470-218476.

[24] Refaeilzadeh, P., L. Tang, and H. Liu, *Cross-validation.* Encyclopedia of database systems, 2009. 5: p. 532-538.

[25] He, J. and X. Fan, Evaluating the performance of the k-fold cross-validation approach for model selection in growth mixture modeling. Structural Equation Modeling: A Multidisciplinary Journal, 2019. 26(1): p. 66-79.

[26] Koklu, M., et al., Classification of Date Fruits into Genetic Varieties Using Image Analysis. Mathematical Problems in Engineering, 2021. 2021.

[27] Singh, D., et al., Classification and Analysis of Pistachio Species with Pre-Trained Deep Learning Models. Electronics, 2022. 11(7): p. 981.

[28] Taspinar, Y.S. and M. Selek, *Object Recognition with Hybrid Deep Learning Methods and Testing on Embedded Systems.* International

Journal of Intelligent Systems and Applications in Engineering, 2020. 8(2): p. 71-77.

[29] Koklu, M., I. Cinar, and Y.S. Taspinar, *Classification of rice varieties with deep learning methods.* Computers and Electronics in Agriculture, 2021. 187: p. 106285.

[30] Ropelewska, E., et al., Discrimination of onion subjected to drought and normal watering mode based on fluorescence spectroscopic data. Computers and Electronics in Agriculture, 2022. 196: p. 106916.

[31] Koklu, M., I. Cinar, and Y.S. Taspinar, *CNN-based bi-directional and directional long-short term memory network for determination of face mask.* Biomedical Signal Processing and Control, 2022. 71: p. 103216.

[32] Altan, G., *Breast cancer diagnosis using deep belief networks on ROI images.* Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi, 2022. 28(2): p. 286-291.

[33] Altan, G., *DeepOCT: An explainable deep learning architecture to analyze macular edema on OCT images.* Engineering Science and Technology, an International Journal, 2022. 34: p. 101091.

[34] Taspinar, Y.S., I. Cinar, and M. Koklu, *Classification by a stacking model using CNN features for COVID-19 infection diagnosis.* Journal of X-ray science and technology, 2022(30): p. 1-16.

[35] Ropelewska, E., K. Sabanci, and M.F. Aslan, Preservation effects evaluated using innovative models developed by machine learning on cucumber flesh. European Food Research and Technology, 2022: p. 1-9.