International Journal of Engineering and Information Systems (IJEAIS)
ISSN: 2643-640X
Vol. 10 Issue 7 July - 2026, Pages: 28-40

Artificial Neural Network-Based Android Malware Detection Using Application Permission Features

Taha Al-Jaro, Muhammad Isleem, Samy S. Abu-Naser

Department of Information Technology,
Faculty of Engineering and Information Technology,
Al-Azhar University, Gaza, Palestine

Abstract*: Android malware continues to challenge mobile security because malicious applications can imitate legitimate software while requesting combinations of permissions that enable surveillance, unauthorized messaging, data access, persistence, or installation of additional packages. This study develops and evaluates a feed-forward artificial neural network for binary Android malware detection using application permission features from the NATICUSdroid dataset. The source file contains 29,332 observations, 86 binary permission variables, and a binary outcome in which 0 denotes benign software and 1 denotes malware. A detailed data audit identified extensive repetition and a smaller number of contradictory labels for identical permission profiles. To reduce information leakage and remove impossible input-output conflicts, identical profiles were consolidated by majority label, tied profiles were removed, and the remaining data were balanced by reproducible undersampling. The final JustNN-compatible dataset contains 5,116 unique permission profiles, equally divided between benign and malicious classes. A stratified 70/30 split produced 3,581 training cases and 1,535 validation cases. The evaluated multilayer perceptron used 86 input neurons, one hidden layer of 16 sigmoid neurons, and one sigmoid output neuron. On the held-out validation set, the network achieved 93.81% accuracy, 93.98% precision, 93.61% malware recall, a 93.79% F1-score, 94.01% specificity, and an ROC-AUC of 0.985. The findings indicate that permission profiles contain strong predictive information and can support lightweight malware screening. Nevertheless, permission-based models should be considered screening mechanisms rather than complete security solutions because permissions do not directly describe runtime behavior, adversaries may manipulate manifest declarations, and performance can decline under temporal or cross-dataset distribution shifts.*

Keywords: Android malware detection; artificial neural networks; application permissions; static analysis; NATICUSdroid; binary classification; cybersecurity.

1. Introduction

Mobile devices have become primary platforms for communication, banking, education, social interaction, and access to organizational systems. This central role has made the Android ecosystem an attractive target for malicious software. Android malware may steal credentials, intercept messages, collect location data, display aggressive advertisements, install additional packages, or maintain unauthorized access to a device. Traditional signature-based detection remains useful for known threats, but it can be weakened by repackaging, code obfuscation, polymorphism, and the rapid production of new variants. Consequently, machine-learning methods have been increasingly investigated as a way to identify statistical patterns that distinguish malicious from benign applications.

Android applications declare permissions in their manifest files to request access to protected system resources and operations. Examples include reading phone state, receiving boot-completion events, accessing storage, sending text messages, and requesting package installation. A single permission is rarely sufficient to determine whether an application is malicious because many legitimate programs require sensitive capabilities. The pattern formed by several permissions, however, can provide a compact representation of an application's intended access requirements. This makes permission-based static analysis computationally inexpensive and suitable for preliminary screening before installation or deeper behavioral inspection.

Artificial neural networks are well suited to this type of pattern-recognition problem. A feed-forward multilayer perceptron can combine many binary indicators, learn nonlinear relationships between them, and output an estimated probability of malware. Unlike a simple rule such as 'classify every app requesting SMS access as malicious,' a neural network can learn conditional combinations. For example, the importance of a messaging permission may depend on whether the same application also requests boot persistence, device identifiers, storage access, or package-installation privileges.

This paper presents a complete, reproducible workflow based on the NATICUSdroid Android Permissions dataset. The work covers data auditing, conflict resolution, class balancing, CSV preparation for JustNN, stratified training and validation, neural-network design, performance evaluation, and interpretation of the most influential permission features. Particular attention is given to duplicate permission profiles. Randomly splitting highly duplicated records can place identical inputs in both training and validation sets, causing optimistic performance estimates. The adopted preprocessing strategy therefore prioritizes methodological validity rather than reporting the highest possible accuracy.

The contribution of this study is threefold. First, it produces a clean and directly importable JustNN dataset with simplified feature identifiers and a separate permission mapping. Second, it evaluates a compact 86-16-1 feed-forward network under a held-out validation protocol. Third, it discusses the practical meaning and limitations of permission-based malware detection, including feature ambiguity, temporal evolution, label noise, and domain shift.

2. Objectives

The general objective of this study is to design and evaluate an artificial neural network that classifies Android applications as benign or malicious using binary application-permission features.

The specific objectives are:

- To obtain and inspect the NATICUSdroid dataset and identify its feature types, target encoding, missing values, class distribution, duplicate records, and contradictory permission profiles.
- To transform the original dataset into a clean CSV format that can be imported directly into JustNN without categorical encoding or numerical normalization problems.
- To prevent data leakage by consolidating identical permission vectors before performing the training-validation split.
- To construct a compact feed-forward neural network with 86 input nodes, one hidden layer, and one binary output node.
- To evaluate the model using accuracy, precision, recall, F1-score, specificity, false-positive rate, false-negative rate, confusion matrix, and ROC-AUC.
- To identify permission variables that have the greatest effect on validation accuracy and interpret their security relevance cautiously.
- To discuss the operational benefits and limitations of permission-based neural malware detection and propose directions for future improvement.

3. Problem Statement

Android users and organizations must evaluate large numbers of applications, while malware authors continuously modify code and packaging to evade known signatures. Dynamic analysis can reveal runtime behavior, but executing applications in instrumented sandboxes requires additional time, infrastructure, and security controls. Static permission analysis is faster and safer because it does not require running the application; however, the relationship between permissions and maliciousness is complex. Sensitive permissions may be legitimate in one context and suspicious in another, while some malware requests only a small set of capabilities or delays harmful behavior.

The technical problem addressed in this research is therefore to learn a nonlinear classification function $f(x)$ that maps an 86-dimensional binary permission vector x to a binary class y, where $y = 0$ represents a benign application and $y = 1$ represents malware. The classifier must achieve high detection sensitivity without producing an excessive number of false alarms. It must also be evaluated on permission profiles that were not copied into the training set. This requirement is important because the raw dataset includes extensive repetition; a naive row-level random split can allow the model to encounter the same permission vector during both learning and validation.

The central research question is: To what extent can a compact feed-forward artificial neural network distinguish Android malware from benign applications using only application-permission features after duplicate and contradictory permission profiles have been addressed? A secondary question concerns which permissions contribute most strongly to the model's decisions and whether the resulting performance is sufficient for a lightweight screening layer.

4. Literature Review

4.1 Static and Dynamic Android Malware Analysis

Android malware-analysis methods are commonly divided into static, dynamic, and hybrid approaches. Static analysis examines artifacts such as the application manifest, permissions, code structure, strings, API calls, certificates, and embedded resources without executing the program. It is generally faster and safer than runtime analysis but may be affected by obfuscation, reflection, dynamic code loading, and incomplete visibility into behavior. Dynamic analysis observes system calls, network traffic, file operations, and other actions while the application executes in a controlled environment. Although dynamic evidence can be behaviorally rich, it is more expensive and may miss logic that is not activated during the observation period. Hybrid systems attempt to combine the coverage of both approaches.

4.2 Permission and Manifest-Based Detection

DREBIN demonstrated that lightweight static features extracted from Android packages could support effective and explainable malware detection directly on resource-constrained devices. Its feature space included requested permissions, hardware

components, application components, filtered intents, restricted API calls, used permissions, suspicious API calls, and network addresses. The study reported detection of approximately 94% of malware with a low false-positive rate, showing that static representations can be highly informative (Arp et al., 2014). DREBIN also emphasized interpretability by presenting the features that contributed to a classification decision.

Permission-based methods are attractive because manifest parsing is fast and does not require code execution. Nevertheless, earlier permission datasets can become outdated as Android evolves, applications adopt new libraries, and developers define custom permissions. Mathur et al. (2021) introduced NATICUSdroid to address this concern by analyzing native and custom permissions from more than 29,000 benign and malicious applications collected between 2010 and 2019. Their best reported Random Forest model achieved 97% accuracy, a 3.32% false-positive rate, and an F-measure of 0.96. The dataset used in the present study is derived from that work.

4.3 API and Behavioral Feature Models

Permissions indicate what an application requests, but API calls provide more direct evidence of how capabilities may be used. Aafer, Du, and Yin (2013) proposed DroidAPIMiner, which extracted API-level features and parameter information to build lightweight Android malware classifiers. DroidMiner extended static analysis by learning fine-grained behavioral patterns from component interactions and program logic (Yang et al., 2014). These studies illustrate the value of richer code-level features, although their extraction is more complex than reading a manifest.

MaMaDroid modeled sequences of abstracted API calls as Markov chains, grouping calls by package or family to increase resilience to changes in the Android framework. It achieved high F-measure on large longitudinal datasets and showed that abstract behavioral structure can remain useful over time (Mariconti et al., 2017; Onwuzurike et al., 2019). The comparison between permission-only and behavior-oriented systems highlights an important design trade-off: simple static indicators provide speed and deployability, while sequence and graph representations may offer greater behavioral specificity.

4.4 Neural Networks for Binary Security Classification

A multilayer perceptron learns a set of weighted transformations from input variables to an output class. For a hidden neuron, the weighted sum $z = w^T x + b$ is passed through an activation function. In the current network, the logistic sigmoid $\sigma(z) = 1/(1 + e^{-z})$ maps the neuron response to the interval from zero to one. Backpropagation adjusts weights to reduce prediction error. Neural networks can model nonlinear interactions among permissions, but they also require careful validation because excessive capacity may memorize training patterns rather than generalize to new applications (Bishop, 2006; Goodfellow et al., 2016).

4.5 Research Gap and Motivation

Many malware studies emphasize algorithm comparison or maximize accuracy on a random split, while less attention is given to duplicate feature vectors and contradictory labels. In a permission dataset, two applications can have the same selected permissions but different labels because permissions alone do not fully describe code behavior. If identical permission profiles occur in both training and validation subsets, measured accuracy may reflect memorization rather than generalization. The present study therefore focuses on a transparent preprocessing and validation workflow suitable for a master's-level neural-network application. Recent cross-domain research has also shown that strong within-dataset results do not guarantee deployment performance; models trained on one permission dataset can lose substantial accuracy when evaluated on another distribution (Islam, 2026). This reinforces the need to report limitations rather than treating a single validation score as universal evidence.

5. Methodology

5.1 Research Design

The study follows a quantitative experimental design. A labeled public dataset is prepared, divided into independent training and validation sets, and used to train a supervised feed-forward neural network. The workflow consists of dataset acquisition, structural inspection, quality auditing, duplicate-profile consolidation, tie removal, class balancing, feature renaming, stratified splitting, model training, and performance evaluation. The final CSV structure is compatible with JustNN, while an equivalent scikit-learn multilayer perceptron is used as a reproducible reference implementation for the numerical results.

5.2 Dataset Description

The NATICUSdroid dataset was obtained from the UCI Machine Learning Repository. UCI describes the dataset as a tabular classification resource containing permission indicators extracted from more than 29,000 benign and malicious Android applications released between 2010 and 2019. The catalog reports 29,333 instances and 86 features with no missing values. The downloaded CSV used in this project contains 29,332 data rows, 86 binary input columns, and one target column named Result. Every permission feature is encoded as 0 when absent and 1 when present. The target is encoded as 0 for benign and 1 for malware.

Property	Value
Dataset	NATICUSdroid (Android Permissions)
Application domain	Android malware detection
Rows in downloaded CSV	29,332
Input variables	86 binary permission indicators
Target variable	Result / Malware
Class encoding	0 = benign; 1 = malware
Missing values	0
Source class counts	14,632 benign; 14,700 malware
Recommended split in UCI metadata	70% training; 30% validation/testing

Table 1. Main characteristics of the source dataset.

5.3 Data Quality Audit and Preprocessing

Initial inspection confirmed that all input and target variables were binary and that no values were missing. Therefore, categorical replacement and min-max normalization were unnecessary: every input already lies within the interval [0, 1]. The principal data-quality issue was duplication. The source file contained 21,841 exact duplicate rows, and grouping by the 86 input features produced 7,395 unique permission profiles. Ninety-six profiles appeared with both benign and malware labels, demonstrating that permissions alone cannot uniquely determine behavior for all applications.

For each identical permission profile, labels were consolidated by majority voting. If more occurrences were labeled malware, the profile received label 1; otherwise it received label 0. Sixteen profiles had an exact tie and were removed because no defensible majority label existed. After consolidation, 7,379 unique profiles remained: 4,821 benign and 2,558 malware. To avoid a model biased toward the majority class and to make accuracy directly interpretable, 2,558 benign profiles were selected using a fixed random seed and combined with all 2,558 malware profiles. The final prepared dataset therefore contained 5,116 unique and balanced permission profiles.

The long Android permission names were replaced by codes F01 through F86 to reduce import and display problems in JustNN. A separate mapping file preserves the original permission names. The output column was renamed Malware and retained as the final column. No continuous scaling was applied because all predictors and the target were already binary.

Audit or transformation	Result
Source rows	29,332
Exact duplicate rows	21,841
Unique permission profiles	7,395
Profiles with contradictory labels	96
Exact-tie profiles removed	16
Profiles after consolidation	7,379
Final balanced profiles	5,116
Final class balance	2,558 benign and 2,558 malware

Table 2. Data-quality audit and preprocessing summary.

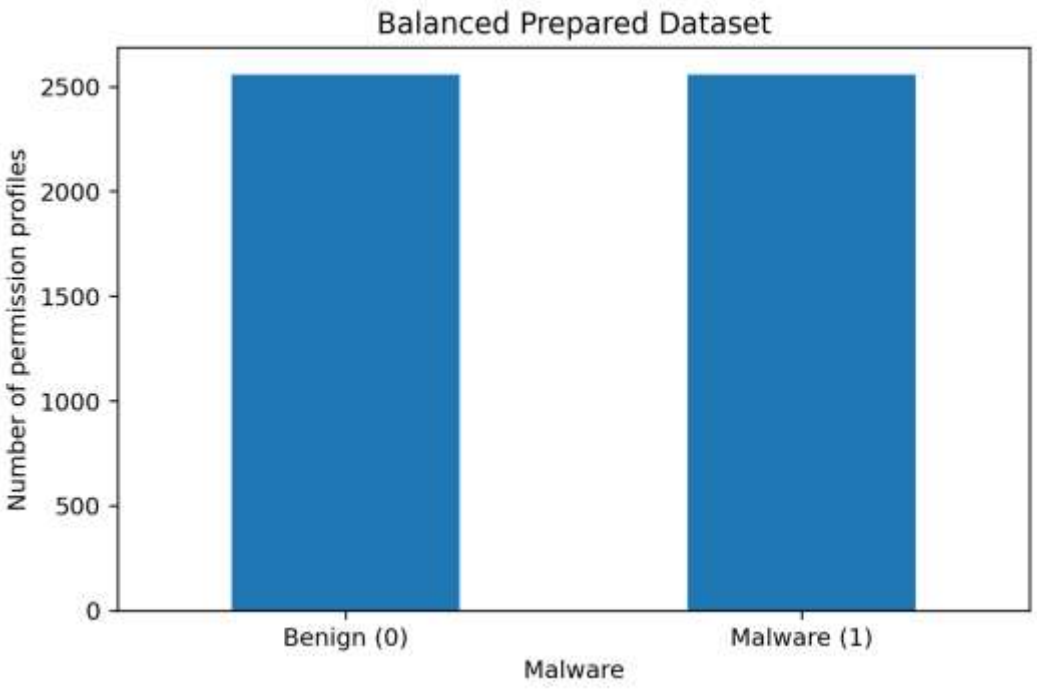

Figure 1. Class distribution after consolidation and balancing.

5.4 Training and Validation Split

The prepared data were divided using a stratified 70/30 split with random seed 42. Stratification preserved equal class proportions in both subsets. The training set contained 3,581 profiles: 1,790 benign and 1,791 malware. The independent validation set contained 1,535 profiles: 768 benign and 767 malware. Consolidation occurred before splitting, ensuring that the same permission vector could not appear in both subsets.

5.5 Neural Network Architecture and Training

The selected network is a feed-forward multilayer perceptron with an 86-16-1 structure. The input layer contains one neuron for each permission feature. A single hidden layer contains 16 neurons using logistic sigmoid activation. The output layer contains one sigmoid neuron whose value is interpreted as the estimated probability of malware. A threshold of 0.50 converts the probability into a binary prediction.

A compact hidden layer was chosen to balance nonlinear modeling capacity against overfitting risk. The reference implementation used the Adam optimizer with an initial learning rate of 0.001, batch size 64, L2 regularization coefficient 0.0001, and a maximum of 500 iterations. Early stopping was activated when the internal validation score failed to improve for 25 consecutive iterations. The network converged after 76 iterations. In JustNN, the same CSV files and 86-16-1 structure can be used, and training should be stopped when validation error no longer decreases.

Parameter	Configuration
Input nodes	86
Hidden layers	1
Hidden nodes	16
Output nodes	1
Hidden activation	Logistic sigmoid
Output activation	Logistic sigmoid
Decision threshold	0.50
Reference optimizer	Adam

Initial learning rate	0.001
Batch size	64
Maximum iterations	500
Early-stopping patience	25 iterations
Observed stopping point	76 iterations

Table 3. Neural-network and training configuration.

5.6 Evaluation Metrics

Performance was evaluated from the validation confusion matrix. True positives are malware profiles correctly classified as malware; true negatives are benign profiles correctly classified as benign; false positives are benign profiles incorrectly flagged as malware; and false negatives are malware profiles missed by the network. Accuracy measures the proportion of all correct predictions. Precision measures the proportion of malware predictions that are correct. Recall or sensitivity measures the proportion of malware cases detected. Specificity measures the proportion of benign cases correctly rejected. The F1-score is the harmonic mean of precision and recall. ROC-AUC summarizes discrimination across decision thresholds. Because false negatives allow malware to pass undetected and false positives burden users or analysts, both error types are reported.

6. Results

6.1 Training Behavior

The reference network stopped after 76 iterations with a final training log-loss of 0.1306. Training accuracy was 95.56%, while held-out validation accuracy was 93.81%. The difference of 1.75 percentage points is modest and does not indicate severe overfitting. The training-loss curve shows rapid improvement during the earlier iterations followed by gradual stabilization.

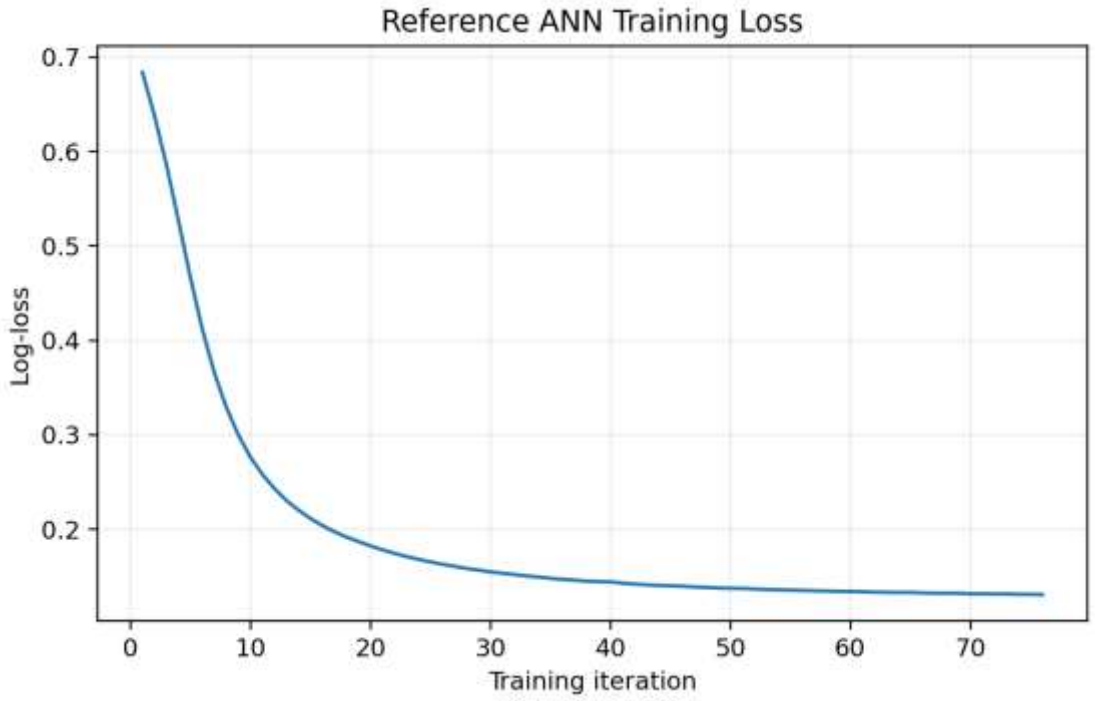

Figure 2. Reference ANN training loss by iteration.

6.2 Validation Performance

On 1,535 independent validation profiles, the model achieved 93.81% accuracy (approximate 95% confidence interval: 92.61% to 95.02%). Malware precision was 93.98%, malware recall was 93.61%, and the F1-score was 93.79%. Specificity for benign applications reached 94.01%. The ROC-AUC of 0.985 indicates strong separation between the two classes over a range of thresholds.

Metric	Validation result
Accuracy	93.81%
Precision – malware	93.98%
Recall / sensitivity – malware	93.61%
F1-score – malware	93.79%
Specificity – benign	94.01%
False-positive rate	5.99%
False-negative rate	6.39%
ROC-AUC	0.985

Table 4. Held-out validation performance.

6.3 Confusion Matrix

The network correctly classified 722 benign profiles and 718 malware profiles. It incorrectly flagged 46 benign profiles as malware and missed 49 malware profiles. The similar numbers of false positives and false negatives indicate a balanced operating point at the 0.50 threshold rather than a model strongly biased toward one class.

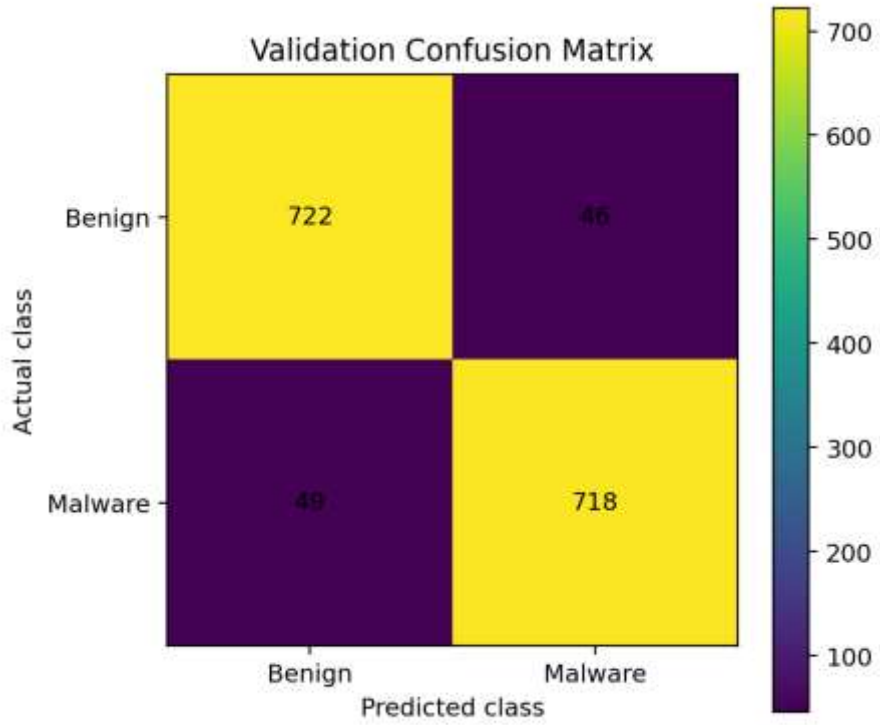

Figure 3. Confusion matrix for the 30% validation set.

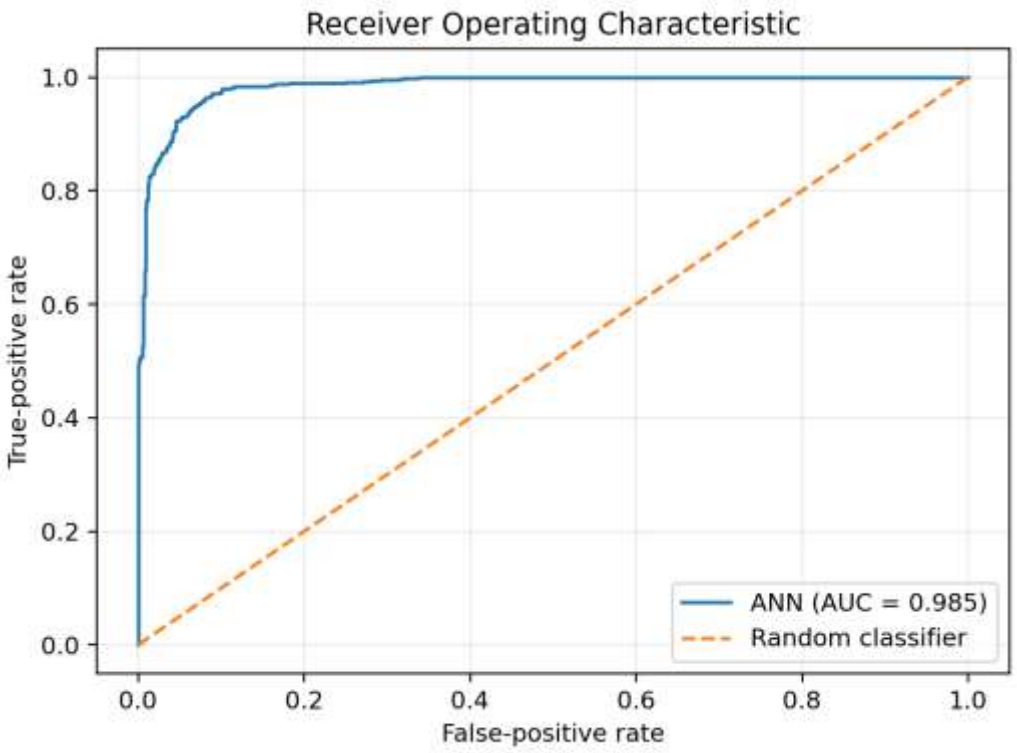

Figure 4. Receiver operating characteristic for the validation set.

6.4 Permission Feature Importance

Permutation importance was calculated by randomly shuffling each permission feature in the validation set and measuring the average decrease in accuracy. READ_PHONE_STATE produced the largest decrease, followed by the Google cloud-messaging receive permission. Permissions related to shortcut installation, Google services, external storage, coarse location, install-referrer binding, billing, boot completion, and Bluetooth also contributed. These values describe the trained model and dataset; they must not be interpreted as proof that requesting a permission causes malicious behavior.

Code	Permission (short name)	Mean accuracy decrease
F22	READ_PHONE_STATE	0.0856
F40	RECEIVE	0.0664
F18	INSTALL_SHORTCUT	0.0111
F10	READ_GSERVICES	0.0093
F06	READ_EXTERNAL_STORAGE	0.0068
F54	ACCESS_COARSE_LOCATION	0.0063
F83	BIND_GET_INSTALL_REFERRER_SERVICE	0.0061
F64	BILLING	0.0050
F66	RECEIVE_BOOT_COMPLETED	0.0044
F69	BLUETOOTH	0.0040

Table 5. Ten highest permutation-importance features.

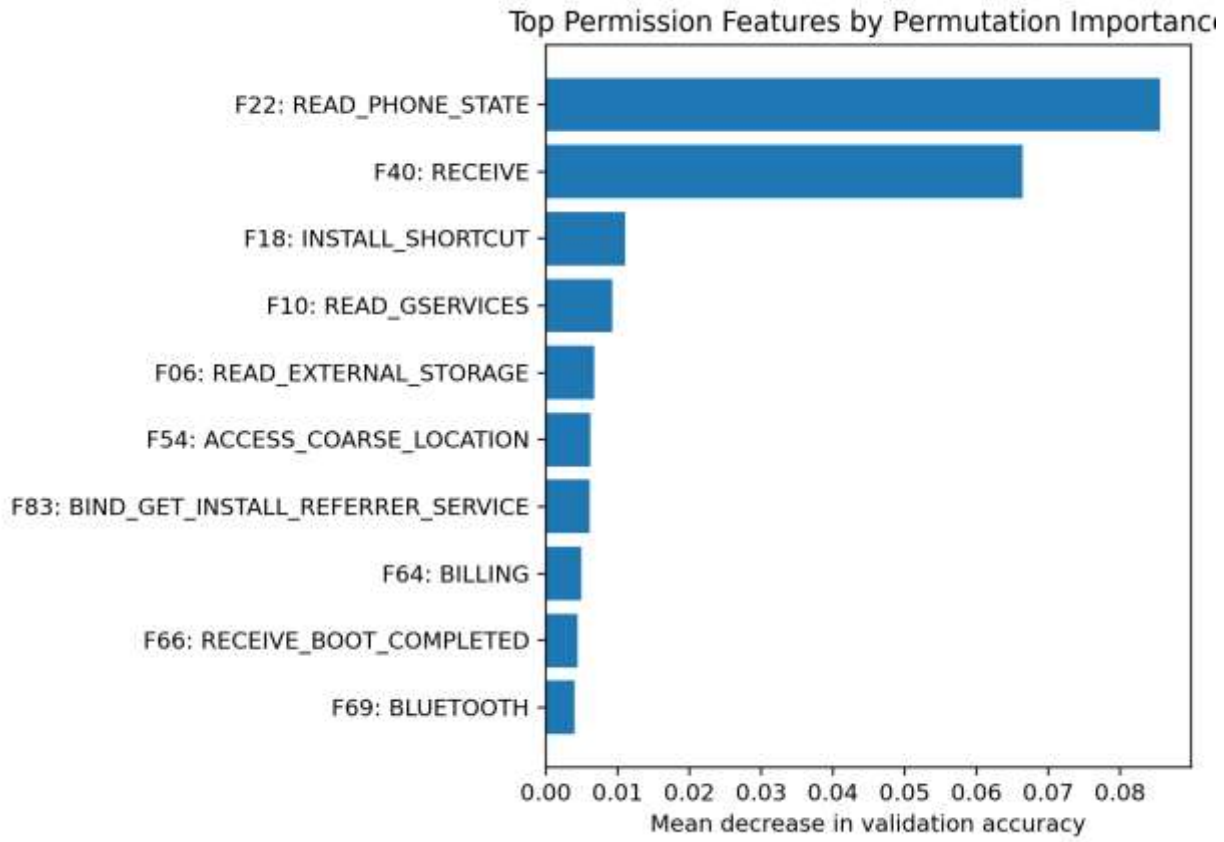

Figure 5. Ten most influential permission features in the validation analysis.

7. Discussion

7.1 Interpretation of Predictive Performance

The results support the central hypothesis that Android permission profiles contain sufficient information for a neural network to distinguish many malicious and benign applications. Validation accuracy above 93% and ROC-AUC near 0.99 demonstrate that the classes are strongly separable in the prepared dataset. The network is also compact: it requires only binary manifest-level inputs and one small hidden layer. Such a model could function as an early screening stage that directs suspicious applications to deeper static or dynamic analysis.

The reported accuracy is lower than the 97% Random Forest accuracy reported in the original NATICUSdroid study. This difference is reasonable for several reasons. The current work uses a neural network rather than the best-performing ensemble in the original study; it removes duplicate-profile leakage; it resolves contradictory input patterns; it balances the classes through undersampling; and it does not perform extensive feature selection or hyperparameter optimization. A lower but more defensible validation result may be preferable to an optimistic score obtained when repeated permission vectors are distributed across both training and validation sets.

7.2 Security Meaning of Important Permissions

READ_PHONE_STATE can expose device and telephony information and has frequently appeared in malware-detection studies because it may support device identification, tracking, or environment checks. RECEIVE_BOOT_COMPLETED allows an application to react after device startup and can support persistence. SEND_SMS can be associated with premium-rate fraud or unauthorized messaging, while REQUEST_INSTALL_PACKAGES may enable the installation of additional applications. READ_EXTERNAL_STORAGE and ACCESS_COARSE_LOCATION provide access to user data and contextual information. However, legitimate communication, navigation, backup, billing, and device-management applications may request the same permissions. The network's strength lies in combining features rather than treating any single permission as a deterministic signature.

7.3 False Positives and False Negatives

The 46 false positives represent benign permission profiles that resembled malware. In practice, these errors could cause unnecessary warnings or analyst workload. The 49 false negatives are more security-critical because malicious profiles were accepted as benign. Threshold adjustment could reduce one error type at the expense of the other. For a consumer application store, a lower threshold may prioritize recall and route more applications to secondary analysis. For an environment in which excessive alarms are disruptive, a higher threshold may be selected. ROC analysis provides a basis for choosing this operating point rather than assuming that 0.50 is universally optimal.

7.4 Methodological Strengths

A major strength of this study is that preprocessing occurred before splitting. Identical permission profiles were not permitted to appear in both training and validation subsets. The class distribution was made exactly balanced, so accuracy was not inflated by majority-class prediction. The random seed, architecture, split files, model artifact, feature mapping, and evaluation metrics are all supplied, which supports reproducibility. The model is also interpretable at the feature level through permutation analysis, although neural-network decisions remain less transparent than linear rules.

7.5 Limitations and Threats to Validity

First, permissions describe declared capabilities rather than observed behavior. An application may request a sensitive permission and never misuse it, or may perform malicious actions through native code, exploited vulnerabilities, dynamic code loading, or APIs not represented by the selected features. Second, consolidating contradictory profiles by majority vote reduces label inconsistency but cannot recover behavioral information that is absent from the inputs. Third, balancing by undersampling discards some benign profiles and may reduce coverage of legitimate application diversity.

Fourth, the dataset spans applications collected between 2010 and 2019. Android permission semantics, application-development practices, malware families, and third-party libraries evolve over time. A model evaluated on a random within-dataset split may perform less effectively on newer applications. Recent cross-domain evaluation has shown substantial performance degradation when permission-based models are transferred between datasets, even when their intra-domain accuracy is high (Islam, 2026). Fifth, the numerical experiment was reproduced with a standard multilayer perceptron implementation. JustNN is used as the required educational interface for importing, dividing, training, and saving the network, but minor differences in weight initialization, optimization, stopping criteria, or row assignment can produce different numerical results.

7.6 Practical Implications

The model is most appropriate as one component of a layered detection architecture. It can rapidly analyze an application's manifest and produce a risk score before installation. High-risk cases can then be examined using code-level API analysis, certificate reputation, network indicators, sandbox execution, or analyst review. The feature-importance results can also guide security policies and educational warnings, but permissions should be interpreted contextually. A banking application legitimately requires network and authentication capabilities, while a simple wallpaper application requesting messaging, boot persistence, phone-state, and installation permissions may deserve additional scrutiny.

8. Conclusion

This research developed a JustNN-compatible artificial neural network workflow for Android malware detection using NATICUSdroid permission features. The source dataset was audited before training, revealing no missing values but substantial duplication and some contradictory labels for identical permission profiles. Majority consolidation, tie removal, and class balancing produced a final dataset of 5,116 unique profiles. A stratified 70/30 split was used to train and validate an 86-16-1 feed-forward network.

The network achieved 93.81% validation accuracy, 93.98% malware precision, 93.61% malware recall, a 93.79% F1-score, and an ROC-AUC of 0.985. These results demonstrate that permission patterns can support accurate and computationally efficient malware screening. READ_PHONE_STATE and cloud-messaging reception were the most influential variables in the evaluated model, but no individual permission should be treated as conclusive evidence of malware.

The study concludes that a compact neural network can provide a useful first-line classifier when its predictions are combined with deeper security analysis. Future work should test the model on temporally newer and externally collected applications, compare multiple network architectures, apply cost-sensitive learning, retain the full benign class through class weighting rather than undersampling, and integrate permissions with API calls, certificate information, intents, and dynamic behavior. Such extensions would improve robustness against adversarial adaptation and dataset shift.

9. JustNN Implementation Procedure

The practical course requirement can be completed using the supplied files as follows. These steps should also be documented with screenshots from the student's Windows installation of JustNN.

1. Open JustNN and create a new grid or project.
2. Import NATICUSdroid_JustNN_Prepared.csv. Select comma as the delimiter and indicate that the first row contains variable names.
3. Define F01 through F86 as input variables and Malware as the output variable. Confirm that 0 means benign and 1 means malware.
4. Use the supplied Training_70 and Validation_30 files to reproduce the research split, or assign 70% of prepared rows to training and 30% to validation without mixing identical profiles.
5. Generate a feed-forward network containing 86 input nodes, one hidden layer with 16 nodes, and one output node.
6. Begin training and monitor both training and validation errors. Stop when validation error no longer improves or begins to rise.
7. Save the trained network or project using the JustNN save command.
8. Export the result table and retain screenshots of the data grid, network structure, training graph, validation results, and saved-network confirmation.

The file ANN_Reference_Model.joblib is supplied only as a reproducibility artifact for the independently trained reference network. It is not a native JustNN network file; the native file must be created and saved from JustNN after importing the prepared CSV.

Appendix A. Selected Feature Code Mapping

The complete 86-feature mapping is supplied in NATICUSdroid_Feature_Mapping.csv. Selected codes referenced in the results are listed below.

Feature code	Original Android permission
F22	android.permission.READ_PHONE_STATE
F40	com.google.android.c2dm.permission.RECEIVE
F18	com.android.launcher.permission.INSTALL_SHORTCUT
F10	com.google.android.providers.gsf.permission.READ_GSERVICES
F06	android.permission.READ_EXTERNAL_STORAGE
F54	android.permission.ACCESS_COARSE_LOCATION
F83	com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE
F64	com.android.vending.BILLING
F66	android.permission.RECEIVE_BOOT_COMPLETED
F69	android.permission.BLUETOOTH
F45	android.permission.SEND_SMS
F47	android.permission.REQUEST_INSTALL_PACKAGES

References

1. Aafer, Y., Du, W., & Yin, H. (2013). DroidAPIMiner: Mining API-level features for robust malware detection in Android. In Security and Privacy in Communication Networks (pp. 86–103). Springer. https://doi.org/10.1007/978-3-319-04283-1_6
2. Arp, D., Spreitzenbarth, M., Hübner, M., Gascon, H., & Rieck, K. (2014). DREBIN: Effective and explainable detection of Android malware in your pocket. Proceedings of the Network and Distributed System Security Symposium (NDSS).
3. Bishop, C. M. (2006). Pattern recognition and machine learning. Springer.
4. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press.
5. Islam, M. R. (2026). Diagnosing and mitigating domain shift in permission-based Android malware detection. arXiv. https://doi.org/10.48550/arXiv.2605.09028
6. Neural Planner Software Ltd. (n.d.). JustNN user guide. JustNN neural network software manual.
7. UCI Machine Learning Repository. (2022). NATICUSdroid (Android Permissions) [Data set]. https://doi.org/10.24432/C5FS64
8. Yang, C., Xu, Z., Gu, G., Yegneswaran, V., & Porras, P. (2014). DroidMiner: Automated mining and characterization of fine-grained malicious behaviors in Android applications. In Computer Security – ESORICS 2014 (pp. 163–182). Springer.
9. Aarra, Lana, et al., "Streamlined Book Rating Prediction with Neural Networks", International Journal of Engineering and Information Systems (IJEAIS), vol. 7, no. 10, pp. 7-13, 2023.
10. Abu Nasser, Mohammed S, et al., "Predictive Modeling of Obesity and Cardiovascular Disease Risk: A Random Forest Approach", International Journal of Academic Information Systems Research (IJAISR), vol. 7, no. 12, pp. 26-38, 2024.
11. Abu-Mehsen, Dana O, et al., "Predicting Audit Risk Using Neural Networks: An In-depth Analysis.", International Journal of Academic Information Systems Research (IJAISR), vol. 7, no. 10, pp. 48-56, 2023.
12. Abunaser, Batool S, et al., "Predictive Modeling of Underweight Malnutrition Using Neural Networks: Insights from Global Nutrition Datasets", pp. 209-222, 2025.
13. Abu-Naser, Samy S, et al., "Predicting Student Performance Using Artificial Neural Network: in the Faculty of Engineering and Information Technology", International Journal of Hybrid Information Technology, vol. 8, no. 2, pp. 221-228, 2015.
14. Abunasser, Basem S, et al., "Abunaser-a novel data augmentation algorithm for datasets with numerical features", Journal of Theoretical and Applied Information Technology, vol. 101, no. 11, 2023.
15. AbuSada, Mohammed Mkhaimar, et al., "Google Stock Price Prediction Using Just Neural Network", International Journal of Academic Engineering Research (IJAER), vol. 7, no. 10, pp. 10-16, 2023.
16. Afana, Mohsen, et al., "Artificial Neural Network for Forecasting Car Mileage per Gallon in the City", International Journal of Advanced Science and Technology, vol. 124, pp. 51-59, 2018.
17. Al Barsh, Yara Ibrahim, et al., "MPG Prediction Using Artificial Neural Network", International Journal of Academic Information Systems Research (IJAISR), vol. 4, no. 11, pp. 7-16, 2020.
18. Alajrami, Eman, et al., "Blood Donation Prediction using Artificial Neural Network", International Journal of Academic Engineering Research (IJAER), vol. 3, no. 10, pp. 1-7, 2019.
19. Al-Araj, Rand Suhail Abu, et al., "Classification of Animal Species Using Neural Network", International Journal of Academic Engineering Research (IJAER), vol. 4, no. 10, pp. 23-31, 2020.
20. Alarayshi, Aya Mazen, et al., "Artificial Neural Network for Global Smoking Trend", International Journal of Academic Information Systems Research (IJAISR), vol. 7, no. 9, pp. 55-61, 2023.
21. Alaydi, Beesan M, et al., "Automatic Cantaloupe Fruit Classification Using Deep Convolutional Neural Networks", International Journal of Academic Information Systems Research (IJAISR), vol. 10, no. 2, pp. 13-18, 2026.
22. Al-Azbaki, Muhammad Ashraf, et al., "Classification of plant Species Using Neural Network", International Journal of Engineering and Information Systems (IJEAIS), vol. 7, no. 10, pp. 28-35, 2023.
23. Al-Baghdadi, Ibrahim Sufyan, et al., "Forecasting COVID-19 cases Using ANN", International Journal of Academic Engineering Research (IJAER), vol. 7, no. 10, pp. 22-31, 2023.
24. Aldaya, Salah-Aldin S, et al., "Diagnosing Sprained Ankles Using Clips", 2025.
25. Alghoul, Ahmed, et al., "Email Classification Using Artificial Neural Network", International Journal of Academic Engineering Research (IJAER), vol. 2, no. 11, pp. 8-14, 2018.
26. Al-Hayik, Saif al-Din Yusuf, et al., "Neural Network-Based Audit Risk Prediction: A Comprehensive Study", International Journal of Academic Engineering Research (IJAER), vol. 7, no. 10, pp. 43-51, 2023.
27. Al-Hayik, Udai Hamed Saeed, et al., "Chances of Survival in the Titanic using ANN", International Journal of Academic Engineering Research (IJAER), vol. 7, no. 10, pp. 17-21, 2023.
28. Al-Hussein, Ahmed Samir Abu, et al., "Analyzing the Relationship between Smoking and Drinking Patterns Using Neural Networks: A Comprehensive Feature-Based Approach", International Journal of Academic Engineering Research (IJAER), vol. 7, no. 9, pp. 18-25, 2023.
29. Ali, Abu Al-Reesh Kamal, et al., "Predictive Modeling of Smoke Potential Using Neural Networks and Environmental Data", International Journal of Engineering and Information Systems (IJEAIS), vol. 7, no. 9, pp. 38-46, 2023.
30. Al-Jalil, Khaled M Abu, et al., "Artificial Neural Network Heart Failure Prediction Using JNN", International Journal of Academic Engineering Research (IJAER), vol. 7, no. 9, pp. 26-34, 2023.
31. Aljerjawi, Nesreen S, et al., "Neural Sorting Networks: Applying Deep Learning to Ranking and Sorting Tasks in NLP", International Journal of Academic Information Systems Research (IJAISR), vol. 9, no. 8, pp. 86-89, 2025.
32. Al-Kahlout, Mutasim Mahmoud, et al., "Neural Network Approach to Predict Forest Fires using Meteorological Data", International Journal of Academic Engineering Research (IJAER), vol. 4, no. 9, pp. 68-72, 2020.
33. Alkronz, Eyad Sameh, et al., "Prediction of Whether Mushroom is Edible or Poisonous Using Back-propagation Neural Network.", International Journal of Academic and Applied Research (IJAAR), vol. 3, no. 2, pp. 1-8, 2019.
34. Allouh, Majd N, et al., "Heart attack analysis & Prediction: A Neural Network Approach with Feature Analysis.", International Journal of Academic Information Systems Research (IJAISR), vol. 7, no. 9, pp. 47-54, 2023.
35. Al-Madhoun, Mohammed Ashraf, et al., "Neural Network-Based Water Quality Prediction", International Journal of Academic Information Systems Research (IJAISR), vol. 7, no. 9, pp. 25-31, 2023.
36. Al-Masawabe, Marah M, et al., "Papaya maturity Classification Using Deep Convolutional Neural Networks", International Journal of Engineering and Information Systems (IJEAIS), vol. 5, no. 12, pp. 60-67, 2021.
37. Al-Massri, Remah, et al., "Classification Prediction of SBRCTs Cancers Using Artificial Neural Network", International Journal of Academic Engineering Research (IJAER), vol. 2, no. 11, pp. 1-7, 2018.
38. Al-Mobayed, Awni Ahmed, et al., "Artificial Neural Network for Predicting Car Performance Using JNN", International Journal of Engineering and Information Systems (IJEAIS), vol. 4, no. 9, pp. 139-145, 2020.
39. Al-Mubayyed, Osama M, et al., "Predicting Overall Car Performance Using Artificial Neural Network", International Journal of Academic and Applied Research (IJAAR), vol. 3, no. 1, pp. 1-5, 2019.
40. Alqedra, Heba IA, et al., "Revolutionizing Lemon Quality Control: A Convolutional Neural Network Approach", International Journal of Academic Information Systems Research (IJAISR), vol. 9, no. 6, pp. 56-63, 2025.
41. Al-Rayes, Marwan RM, et al., "Smoke Detectors Using ANN", International Journal of Academic Engineering Research (IJAER), vol. 7, no. 10, pp. 1-9, 2023.
42. Al-Sharif, Ahmed Muhammad Haider, et al., "Predicting Heart Disease using Neural Networks", International Journal of Academic Information Systems Research (IJAISR), vol. 7, no. 9, pp. 40-46, 2023.
43. Al-Shawwa, Mohammed, et al., "Predicting Birth Weight Using Artificial Neural Network", International Journal of Academic Health and Medical Research (IJAHMR), vol. 3, no. 1, pp. 9-14, 2019.
44. Al-Shawwa, Mohammed, et al., "Predicting Effect of Oxygen Consumption of Thylakoid Membranes (Chloroplasts) from Spinach after Inhibition Using Artificial Neural Network", International Journal of Academic Engineering Research (IJAER), vol. 3, no. 2, pp. 15-20, 2019.
45. Al-Shawwa, Mohammed, et al., "Predicting Temperature and Humidity in the Surrounding Environment Using Artificial Neural Network", International Journal of Academic Pedagogical Research (IJAPR), vol. 2, no. 9, pp. 1-6, 2018.
46. Ashqar, Belal A M, et al., "Identifying Images of Invasive Hydrangea Using Pre-Trained Deep Convolutional Neural Networks", International Journal of Academic Engineering Research (IJAER), vol. 3, no. 3, pp. 28-36, 2019.
47. Attia, Maher Wissam, et al., "Predicting Fire Alarms in Smoke Detection using Neural Networks", International Journal of Academic Information Systems Research (IJAISR), vol. 7, no. 10, pp. 26-33, 2023.
48. Baraka, Raghad Fattouh, et al., "Predicting Books' Rating Using Just Neural Network", International Journal of Engineering and Information Systems (IJEAIS), vol. 7, no. 9, pp. 14-19, 2023.
49. Barhoom, Alaa M, et al., "Predicting Titanic Survivors using Artificial Neural Network", International Journal of Academic Engineering Research (IJAER), vol. 3, no. 9, pp. 8-12, 2019.
50. Dalffa, Mohaned Abu, et al., "Tic-Tac-Toe Learning Using Artificial Neural Networks", International Journal of Engineering and Information Systems (IJEAIS), vol. 3, no. 2, pp. 9-19, 2019.
51. Dawood, Kamel Jamal, et al., "Artificial Neural Network for Mushroom Prediction", International Journal of Academic Information Systems Research (IJAISR), vol. 4, no. 10, pp. 9-17, 2020.
52. Dawoud, Alaa Mohammed, et al., "Predicting Life Expectancy in Diverse Countries Using Neural Networks: Insights and Implications", International Journal of Academic Engineering Research (IJAER), vol. 7, no. 9, pp. 46-54, 2023.
53. Dheir, Ibtesam M, et al., "Classifying Nuts Types Using Convolutional Neural Network", International Journal of Academic Information Systems Research (IJAISR), vol. 3, no. 12, pp. 12-18, 2020.
54. El_Jerjawi, Nesreen Samer, et al., "Diabetes Prediction Using Artificial Neural Network", International Journal of Advanced Science and Technology, vol. 121, pp. 55-64, 2018.
55. El-Khatib, Mohmmad Jamal, et al., "Glass Classification Using Artificial Neural Network", International Journal of Academic Pedagogical Research (IJAPR), vol. 3, no. 2, pp. 25-31, 2019.
56. El-Mahelawi, Jamal Khamis, et al., "Tumor Classification Using Artificial Neural Networks", International Journal of Academic Engineering Research (IJAER), vol. 4, no. 11, pp. 8-15, 2020.
57. Elmahmoum, Abedeleilah SA, et al., "Image-Based Classification of Date Types Using Convolutional Neural Networks", International Journal of Academic Information Systems Research (IJAISR), vol. 9, no. 1, pp. 10-16, 2025.
58. Elzamly, Abdelrafe, et al., "Predicting Critical Cloud Computing Security Issues using Artificial Neural Network (ANNs) Algorithms in Banking Organizations", International Journal of Information Technology and Electrical Engineering, vol. 6, no. 2, p. 40-45, 2017.
59. Hadi, Ahmed Munther Abdel, et al., "Forecasting Stock Prices using Artificial Neural Network", International Journal of Engineering and Information Systems (IJEAIS), vol. 7, no. 10, pp. 42-50, 2023.
60. Harara, Mahmoud, et al., "Unlocking Literary Insights: Predicting Book Ratings with Neural Networks", International Journal of Academic Information Systems Research (IJAISR), vol. 7, no. 10, pp. 22-27, 2023.
61. Harz, Hussam Hatem, et al., "Artificial Neural Network for Predicting Diabetes Using JNN", International Journal of Academic Engineering Research (IJAER), vol. 4, no. 10, pp. 14-22, 2020.
62. Hasan, Walaa, et al., "Artificial Neural Network for Predicting COVID 19 Using JNN", International Journal of Academic Engineering Research (IJAER), vol. 7, no. 10, pp. 41-47, 2023.
63. Hassanein, Raghad Adnan Abu, et al., "Artificial Neural Network for Predicting Workplace Absenteeism", International Journal of Academic Engineering Research (IJAER), vol. 4, no. 9, pp. 62-67, 2020.
64. Heriz, Hazem Hatem, et al., "English Alphabet Prediction Using Artificial Neural Networks", International Journal of Academic Pedagogical Research (IJAPR), vol. 2, no. 11, pp. 8-14, 2018.
65. Jamala, Mohammed N, et al., "Predicting MPG for Automobile Using Artificial Neural Network Analysis", International Journal of Academic Information Systems Research (IJAISR), vol. 2, no. 10, pp. 5-21, 2018.
66. Jarghon, Amira, et al., "Predicting Kidney Stone Presence from Urine Analysis: A Neural Network Approach using JNN", International Journal of Academic Information Systems Research (IJAISR), vol. 7, no. 9, pp. 32-39, 2023.
67. Kashf, Dalia Wajeeh Abu, et al., "Predicting DNA Lung Cancer using Artificial Neural Network", International Journal of Academic Pedagogical Research (IJAPR), vol. 2, no. 10, pp. 6-13, 2018.
68. Khalil Ahmed, J, et al., "Energy Efficiency Prediction using Artificial Neural Network", International Journal of Academic Pedagogical Research (IJAPR) _, vol. 3, no. 9, pp. 1-7, 2019.
69. Kweik, Ola Mohammed Abu, et al., "Artificial Neural Network for Lung Cancer Detection", International Journal of Academic Engineering Research (IJAER), vol. 4, no. 11, pp. 1-7, 2020.
70. Maghari, Alaa Mazen, et al., "Books' Rating Prediction Using Just Neural Network", International Journal of Engineering and Information Systems (IJEAIS), vol. 4, no. 10, pp. 17-22, 2020.
71. Malekpourheydari, SALIMEH, et al., "Wavelet transform and neural network model for streamflow forecasting", J. Theor. Appl. Inf. Technol, vol. 100, 2022.
72. Marouf, Ahmad, et al., "Predicting Antibiotic Susceptibility Using Artificial Neural Network", International Journal of Academic Pedagogical Research (IJAPR), vol. 2, no. 10, pp. 1-5, 2018.
73. Mattar, Mohammad Ayman, et al., "Spotify Status Dataset", International Journal of Engineering and Information Systems (IJEAIS), vol. 7, no. 10, pp. 14-21, 2023.
74. Metwally, Najla'Fathi, et al., "Diagnosis of Hepatitis Virus Using Artificial Neural Network", International Journal of Academic Pedagogical Research (IJAPR), vol. 2, no. 11, pp. 1-7, 2018.
75. Mohammed, Ghaida Riyad, et al., "Predicting the Age of Abalone from Physical Measurements Using Artificial Neural Network", International Journal of Academic and Applied Research (IJAAR), vol. 4, no. 11, pp. 7-12, 2020.
76. Musleh, Musleh M, et al., "Predicting Liver Patients using Artificial Neural Network", International Journal of Academic Information Systems Research (IJAISR), vol. 3, no. 10, pp. 1-11, 2019.
77. Nasser, Ibrahim M, et al., "A Proposed Artificial Neural Network for Predicting Movies Rates Category", International Journal of Academic Engineering Research (IJAER), vol. 3, no. 2, pp. 21-25, 2019.
78. Nasser, Ibrahim M, et al., "Artificial Neural Network for Diagnose Autism Spectrum Disorder", International Journal of Academic Information Systems Research (IJAISR), vol. 3, no. 2, pp. 27-32, 2019.
79. Nasser, Ibrahim M, et al., "Artificial Neural Network for Predicting Animals Category", International Journal of Academic and Applied Research (IJAAR), vol. 3, no. 2, pp. 18-24, 2019.
80. Nasser, Ibrahim M, et al., "Developing Artificial Neural Network for Predicting Mobile Phone Price Range", International Journal of Academic Information Systems Research (IJAISR), vol. 3, no. 2, pp. 1-6, 2019.
81. Nasser, Ibrahim M, et al., "Lung Cancer Detection Using Artificial Neural Network", International Journal of Engineering and Information Systems (IJEAIS), vol. 3, no. 3, pp. 17-23, 2019.
82. Nasser, Ibrahim M, et al., "Predicting Books' Overall Rating Using Artificial Neural Network", International Journal of Academic Engineering Research (IJAER), vol. 3, no. 8, pp. 11-17, 2019.
83. Nasser, Ibrahim M, et al., "Predicting Tumor Category Using Artificial Neural Networks", International Journal of Academic Health and Medical Research (IJAHMR), vol. 3, no. 2, pp. 1-7, 2019.
84. Nasser, Mohammed S Abu, et al., "Leveraging Artificial Neural Networks for Cancer Prediction: A Synthetic Dataset Approach", International Journal of Academic Engineering Research (IJAER), vol. 7, no. 11, pp. 43-51, 2023.
85. Qamar, Sulafa Yhaya Abu, et al., "Predicting the Number of Calories in a Dish Using Just Neural Network", International Journal of Academic Information Systems Research (IJAISR), vol. 7, no. 10, pp. 1-9, 2023.
86. Qarmout, Hussein Khaled, et al., "Alzheimer: A Neural Network Approach with Feature Analysis.", International Journal of Academic Information Systems Research (IJAISR), vol. 7, no. 10, pp. 10-18, 2023.
87. Qwaider, Sabreen R, et al., "Artificial Neural Network Prediction of the Academic Warning of Students in the Faculty of Engineering and Information Technology in Al-Azhar University-Gaza", International Journal of Academic Information Systems Research (IJAISR), vol. 4, no. 8, pp. 16-22, 2020.
88. Sabah, Ahmed S, et al., "Pistachio Variety Classification using Convolutional Neural Networks", International Journal of Academic Information Systems Research (IJAISR), vol. 8, no. 4, pp. 113-119, 2024.
89. Sadek, Ramzi M, et al., "Parkinson's Disease Prediction Using Artificial Neural Network", International Journal of Academic Health and Medical Research (IJAHMR), vol. 3, no. 1, pp. 1-8, 2019.
90. Safiah, Saja Kh Abu, et al., "Climate Change temperature Prediction Using Just Neural Network", International Journal of Academic Engineering Research (IJAER), vol. 7, no. 9, pp. 35-45, 2023.
91. Salah, Mohammed, et al., "Predicting Medical Expenses Using Artificial Neural Network", International Journal of Engineering and Information Systems (IJEAIS), vol. 2, no. 20, pp. 11-17, 2018.
92. Samhan, Lamis F, et al., "Classification of Alzheimer's Disease Using Convolutional Neural Networks", International Journal of Academic Information Systems Research (IJAISR), vol. 6, no. 3, pp. 18-23, 2022.
93. Sultan, Alaa Yousef Abu, et al., "Neural Style Transfer Using Convolutional Neural Networks: An Experimental Study", International Journal of Academic Information Systems Research (IJAISR), vol. 10, no. 2, pp. 50-57, 2026.
94. Sultan, Anas Bachir Abu, et al., "Predictive Modeling of Breast Cancer Diagnosis Using Neural Networks: A Kaggle Dataset Analysis", International Journal of Academic Engineering Research (IJAER), vol. 7, no. 9, pp. 1-9, 2023.
95. Younis, Ibrahim, et al., "Prediction Heart Attack using Artificial Neural Networks (ANN)", International Journal of Engineering and Information Systems (IJEAIS), vol. 7, no. 10, pp. 36-41, 2023.
96. Zaiter, Ahmed Salama Abu, et al., "Web page phishing detection Using Neural Network", International Journal of Engineering and Information Systems (IJEAIS), vol. 7, no. 9, pp. 1-13, 2023.