

CyberSentinel: AI-Powered Threat Detection & Explainable Security

Durva Patkar¹, Vaishnavi Rawate², Madhuri Patil³, Sumedh Pundkar⁴

¹Student, Dept of Computer Science and technology, Usha mittal institute of technology, Maharashtra, India

²Student, Dept of Computer Science and technology, Usha mittal institute of technology, Maharashtra, India

³Student, Dept of Computer Science and technology, Usha mittal institute of technology, Maharashtra, India

⁴Professor, Dept of Computer Science and technology, Usha mittal institute of technology, Maharashtra, India

Abstract - The mobile devices have become part of everyday life and the newfound use has turned them into easy targets of advanced cyber-attacks, including malicious apps, unprotected Wi-Fi networks, and background connections. The current mobile security tools tend to be based on the concept of static or signature detection and have therefore been left deficient in the protection of zero-day attacks and network-based exploits. The presented paper CyberSentinel, is an AI-driven Android security software that is intended to offer a full-fledged, real-time protection both on the device and network levels. CyberSentinel combines an AI-based malware detection engine, which can analyze the behavior of newly instigated and existing applications, a system of security mode configuration, an application-level network surveillance to identify malicious IP addresses, and a Wi-Fi scanner to assess the safety of the public network. The system also has an LLM-powered chatbot that provides actionable advice and explains security alerts to make them easier to use. Through the integration of intelligent malware, network threat, and user-focused design, CyberSentinel seals the most important vulnerabilities existing in the mobile security market and pushes the Android cybersecurity to a higher level.

Keywords-Android Security, Mobile Malware Detection, AI-based Threat Detection, Neural Networks, Wi-Fi Security Analysis, Large Language Models (LLM), Threat Remediation, Real-time Protection and Permission-based Features.

1. INTRODUCTION

Smartphones are vital in the digital world, and used to communicate with friends and carry out banking transactions, as well as access sensitive information, but they are becoming targets of advanced cyber-attacks without the majority of users understanding the dangers of downloading malicious software, using unprotected Wi-Fi, and unnoticed background applications. Available security solutions primarily identify known malware and cannot detect zero-day and network-based attacks [1]. One of the major issues is that the users install apps without understanding whether they are safe or not, and it can result in the theft of their data or the

communication of harmful messages. Another significant weakness is the presence of public Wi-Fi, which is usually not encrypted or even created by malicious individuals to capture data [5]. The proposed paper will consist of an advanced Android security application that provides both de-vice and network protection. It monitors the new applications and the old applications when a user selects it in new or existing modes and identifies suspicious activity using AI-driven, behavior-based systems [2]. It is also able to notify the user about threats, block access to apps, or recommend deletion. The app also has an LLM-based chatbot, which explains notifications, gives guidelines, and directs users in real time to make the app more usable[9].

This paper is aimed to create an Android application that: Introduce AI-based malware detection to track new and already installed apps and their behavior to detect possible security threats in real-time [1]. Provide three customizable security levels (Low, Balanced, High) so that the users can tailor the degree of monitoring and security to their needs in terms of performance and security. Install an App-Level Network Monitor that keeps checking the app connections with the dynamically created database of legal malicious IP addresses, which gives proactive protection against network-based attacks [3]. Add a Wi-Fi Network Scanner capable of assessing the security of public Wi-Fi networks based on the evaluation of important parameters like the type of network, its security status, and speed and has features of giving troubleshooting recommendations [6][8]. Provides an easy-to-use interface to track the security of the device and to react to the possible threats without impacting on the performance and battery life of the device significantly. Integrate an AI-based chatbot that will respond to the user, give direct explanations of security notifications, respond to questions, and make personalized suggestions to increase their understanding of the security of their device and its control.

This paper adds to the area of mobile security by presenting a system that secures the user from threats coming from both malicious applications and insecure public Wi-Fi networks. It describes an Android

application that uses AI-based malware detection to analyze potential threats in real time, like zero-day attacks, without relying on traditional signature-based detection methods. The system has three adjustable security levels: Low, Balanced, and High, which offer different degrees of protection depending on performance trade-offs. It also includes an App-Level Network Monitor that can find network connections to suspicious IP addresses and a Wi-Fi scanner meant for checking how safe nearby networks are to help users stay away from insecure connections.

2. LITERATURE SURVEY

The paper [1] ReDroidDet (2021) suggests an Android malware detector based on Recurrent Neural Networks (RNN) and based on the use of the static features in the API calls, permissions, system events, and permission utilization rates that have high accuracy (~98.58%), and better than traditional machine learning models like SVM and KNN because it only uses the static analysis and is unable to detect the runtime malware or zero-day malware that conceals malicious activity. Correspondingly, [2] Hybrid Android Malware Detection Using Deep Neural Networks uses deep neural networks (DNNs) that combine several static predictors (permissions, API calls, code patterns) to better predict malware but will not detect the malware as it comes online to activate, and relies primarily on the static analysis. However, [3] Network-Traffic-Based Android Malware Detection is more dynamic behavior-based: it analyzes internet communication patterns, including IP addresses, traffic size and connection frequency, with bagging, random forest and boosting machine learning algorithms yielding more than 95 percent average malicious network activity detection. Lastly, [4] AntMonitor (VpnService-based monitoring system) uses the VpnService API of the Android system to establish a local VPN on the device, monitoring all app network traffic without becoming rooted, allowing the system to provide detailed app-level logging of app network traffic and detect privacy leaks; more than most research-oriented systems, which only raise red flags on suspicious behavior, this allows users to take real-time actions against real-time threats.

George Chatzisoifroniou and Panayiotis Kotzanikolaou (2025) security design of Wi-Fi Easy Connect, in fact, the Device Provisioning Protocol – DPP; which was brought to replace Wi-Fi Protected Setup or WPS [5]. Their paper discussed the authentication and cryptographic mechanism of DPP. Also mentioned possible weaknesses like configurator impersonation, downgrade attack and implementation flaws. DPP has better security than WPS but according to them bad configuration can lead to vulnerability in security [5]. The study by Prateek

Bheevgade et al. (2024) discussed security threats as a supplementary approach with increased usage of public Wi-Fi networks [6]. This study uncovered through survey results and practical testing that packet sniffing, rogue access points as well as propagation of malware and man-in-the-middle attacks are some risks[6]. The authors stressed on user awareness along with protective measures such as secure authentication and VPN usage to mitigate these threats [6].

The article devoted to the accuracy of digital infrastructures and the expert use of AI models. A research by Reinaldo Sanchez-Arias et al. (2023) on access to broadband connectivity in Polk County, Florida, showed that the coverage and speed test in official FCC reports are frequently overemphasized in cases of rural communities [7]. The authors compared this information to crowdsourced data provided by Ookla and M-Lab; hence, their argument was that it is necessary to conduct real-world speed tests to help planners detect areas that are actually underserved. Continuing on this measurement focus, Kyle MacMillan et al. (2023) made a comparative study of the Ookla Speed test and NDT7 of M-Labs [8]. They discovered through laboratory experiments and thousands of tests in homes that the tools typically converge, but Ookla is more likely to record higher speeds in high-latency conditions, and NDT7 is less likely to record peak performance, indicating that the choice of server and software versions are also critical to correct data interpretation [8]. Moving on to the topic of artificial intelligence (AI), Hanxiang Xu et al. (2024) have offered a systematic examination of the means of syncing the Large Language Models (LLMs) to the realm of cybersecurity [9]. They have discussed 127 papers and reported that the functionality of the LLMs could be applied effectively in solving such tasks as malware analysis or vulnerability detection, although the research area is still concerned with data privacy, protection, and a shortage of various training sets [9]. Lastly, one of the studies on the educational technology elaborated the creation of a domain-specific chatbot based on RASA and LSTM networks. This system is intended to support student and user queries regarding exams and courses content, therefore, it is designed to automatize responses using intent recognition, which illustrates how deep learning can greatly ease the burden of administration and offer students access to information in real-time [10].

3. METHODOLOGY

The methodology for developing the AI-based malware detector and application-level network threat detection system on Android devices is comprehensive, covering everything from the conceptual framework to data preparation, AI algorithms, and system architecture. The core idea is to integrate machine learning-based

malware detection with real-time network traffic monitoring and threat mitigation.

3.1. Dataset Overview and Feature Engineering

a. DREBIN Dataset:

This dataset contains approximately 15,000 samples of Android applications, corresponding to older malware, which was prevalent during the time of its collection, to test the perceptions of the static analysis and malware classification.

b. TUANDROMD Dataset:

It consists of around 4,000 Android apps, which contains recent malware trends.

c. COLCOM dataset:

It comprises of around 400 Android application samples (where 1 indicates malware, 0 benign applications), with a temporal range between DREBIN and TUANDROMD, which can be used to test the model performance on new or unseen malware samples.

The feature engineering is concerned with the connection between various malware behavior and system permissions. Raw and engineered permission features are used. Raw features are 14 Android permissions whereas engineered features group related permissions into behaviour patterns related to a malware categories like spyware, ransomware and SMS worms, enhancing the detection of unknown samples.

3.2. System Overview

a. Wi-Fi Threat Detection and Risk Assessment

When the app is opened, it will check that all the necessary permissions (location access) are granted. Upon enabling Wi-Fi, it extracts network information such as SSID, BSSID and type of encryption [5]. It searches the nearby Wi-Fi networks and classifies them as high-risk systems by their encryption strength, signal strength, and consistency of the SSID. The networks are categorized into High Risk, Moderate, and Low Risk networks and the user is notified on the same.

b. Network Center Module

It is the key control unit of the Wi-Fi diagnostics and has three sub-units:

- General Info: SSID, IP address, and encryption protocol information are presented.
- Speed Test: A test that approximates the present

download and upload speed.

- Troubleshoot: It is used to check the connectivity by using ping tests with external servers.

The Network Center also gives the users information on performance as well as possible solutions to connectivity problems.

c. Application-Level Network Scanner

It is used to track app-level network traffic to detect communications with malicious IP addresses. It starts by checking whether the network monitoring is on or off. The system activates and reads the installed applications then retrieves the TCP and UDP connection data of those applications using system files (/proc/net/tcp and /proc/net/udp). The obtained remote IP addresses are matched with a local threat intelligence database. If any malicious IP address is detected, it notifies the user and gives suggested remedial measures. Once this is done, after 5 minutes the system resumes again by repeating the monitoring cycle to provide continuous periodic scanning.

d. AI-Powered Malware Detection Module

It consist of User Interface and a Malware Detection Engine. The user is allowed to either scan all installed apps, or selected apps. The system identifies the chosen applications, gets their permissions, transforms them into feature vectors, and runs the data through a pre-trained neural network to get a maliciousness confidence score. A context-aware score is obtained by adding the model confidence and the app category risk along with sideloading status to decrease the false positives and increase the accuracy of the model using the following equation:

$$0.5 \times \text{Model Confidence} + 0.3 \times \text{Category Risk} + 0.2 \times \text{Sideloading Score}$$

The system will then show the possible malicious apps as well as recommended remedial action.

e. Security Modes

The system has configurable protection modes that can enable users to select:

- Manual Mode: Manual configuration of security settings is done by a user.
- Balanced Mode: Optimizes the operations, depending on the status of devices and their battery.

- Smart Mode: It is an automatic adjustment of the intensity of protection based on system intelligence.

f. Help Bot Module

The system combines an AI chatbot which is an OpenRouter.ai (Mistral) bot that assists the end user to comprehend threats that are detected, clarify system notifications, and recommend suitable security measures after interacting with a natural language [9].

g. Real-Time System Monitoring

This continuously monitors Android package installations and OS-level this will monitor applications that have just been installed or updated. It automatically initiates relevant scans to ensure that there is current security.

h. Threat Remediation

After one threat has been identified, be it an application threat or a network threat, then the system will show detailed threat information and offer the choice of Delete, Ignore, or Quarantine. Once the system has been remedied, it verifies that the remediation has been done successfully and updates the user interface. The proposed workflow will guarantee the suggested system to conduct intelligent and automated detection of malware and network threats with deep learning and real-time monitoring.

malicious activities in Android applications due to their ability to automatically to learn features and make generalizations based on large amounts of data.

The neural network used here is a binary classifier having 28 input features. The architecture has two hidden layer with 64 and 32 neurons respectively, where the ReLU (Rectified Linear Unit) activation function to provide non-linearity. The last layer is the output layer (one neuron), which applies sigmoid activation function, to produce a probability score to determine the application to be benign or malicious. The model was executed in the TensorFlow framework, that automatically uses backpropagation in the training process to update the network weights and enhance the accuracy of classification.

The architecture of the neural network applied in this paper is shown in the following diagram:

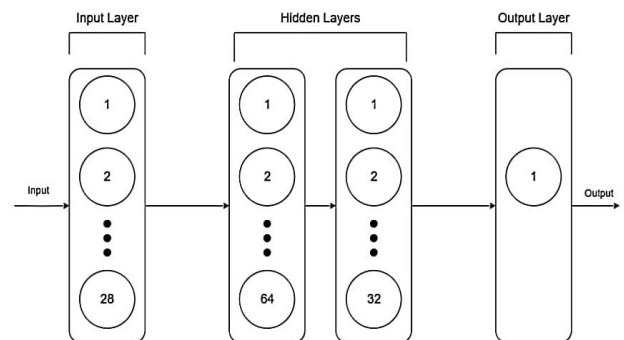


Fig. 2. Neural Network Architecture used for Android Malware Detection

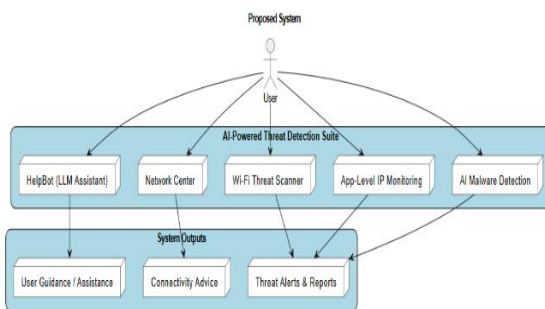


Fig 1. Proposed System for Cyber Sentinel

3.3. Artificial Neural Network for Malware Detection

The ANNs are computational models that are based on the structure and operation of the human brain. They are made up of multiple interrelated layers of simple processing units known as neurons, and which learn patterns from data through weighted connectivity. Image classification, natural language processing and malware detection are the tasks that neural networks are useful for. They are capable of detecting weak and hidden

Algorithm: Android Malware Detection using Artificial Neural Network

Input: Static feature dataset with 28 features per Android application.

Output: Classification label - Benign or Malicious.

Step 1: Collect static features such as permissions and engineered behavioral indicators from Android applications.

Step 2: Preprocess the dataset by normalizing numerical features and encoding categorical values.

Step 3: Split the dataset into training and testing subsets.

Step 4: Initialize the neural network parameters:

Input layer: 28 neurons (one for each feature).

Hidden Layer 1: 64 neurons with ReLU activation, defined as:

$$\text{ReLU}(x) = \max(0, x)$$

Hidden Layer 2: 32 neurons with ReLU activation.

Output layer: 1 neuron with Sigmoid activation, defined as:

$$\sigma(x) = 1 / (1 + e^{-x})$$

Step 5: Perform forward propagation - pass the input

data through the network to compute outputs at each layer using the activation functions.

Step 6: Compute the loss using the Binary Cross-Entropy function to measure the error between predicted and actual labels.

Step 7: Apply back propagation to adjust weights and biases, minimizing the loss using the gradient descent optimization technique.

Step 8: Train the model for multiple epochs until the loss converges and desired accuracy is achieved.

Step 9: Evaluate the trained model using the test dataset to measure its accuracy and performance.

Step 10: For a new application, extract the same 28 features and feed them into the trained model.

Step 11: The model outputs a probability between 0 and 1. If the output ≥ 0.5 , classify the app as Malicious; otherwise, classify it as Benign.

4. IMPLEMENTATION RESULTS

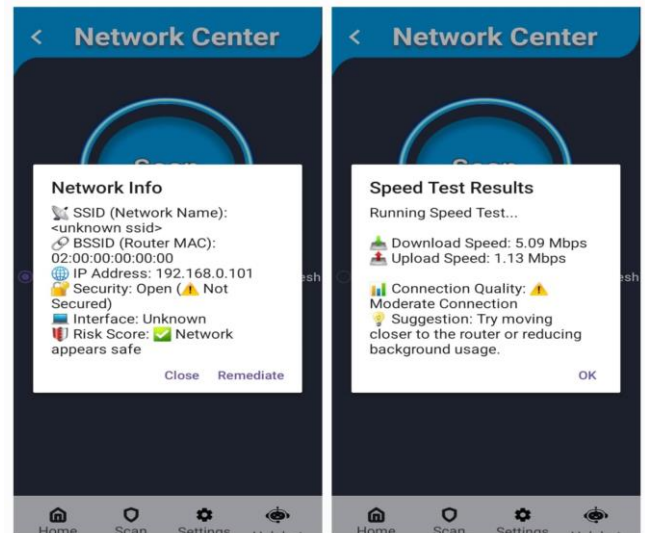
The Cyber Sentinel application Home Screen can be taken as the central dashboard that allows one to have a brief glance at the current security status of the device. It shows network-based threat warnings and information about the connected Wi-Fi network, such as SSID, BSSID, type of encryption as well as the security classification based on internal scan. The bottom navigation bar enables easy access to the most important modules, including Malware Scan, Settings, and HelpBot, which ensures a smooth and intuitive navigation throughout the whole application. The Network Center Screen enables the user to conduct the manual network diagnostics using three modes: General Test, Speed Test, and Troubleshoot.



Fig. 3. Home Screen and Network Center

Network Center Screen: The General Test shows information on SSID, BSSID, IP address, type of encryption and the general security rating. Speed Test is

used to measure real-time upload and download speeds and the Troubleshoot mode is used to test connectivity by pitting an external server (8.8.8.8). This is a modular architecture that allows users to have necessary monitoring and maintenance tools that keep the network



running safely.

Fig. 4. Network Info and Speed Test Results

The Malware Scanning Screen is the main interface of scanning devices. It also gives users two choices of scanning, All Apps, which scans all the installed applications and Select Apps which enables users to scan specific applications. Key Scan button triggers the selected scanning procedure so the interface is easy, straightforward, and user-friendly.



Fig. 5. All App Scan Interface and Display Malicious Apps

Threat Remediation is smart, taking into account risks detected. In the case of bad applications, it displays threat details and has a Delete App button to uninstall it. In the case of browser-based threats such as a phishing page or malicious URL, it does not provide the delete option, rather, it shows a warning. This makes it easy to understand and provides a contextual guide and makes the interface easy to use. Note: Google Play Services is not in fact connecting with the suspicious IP address in Figure. The IP was simply added in our threat intelligence database on a temporary basis to be used only as a test. The HelpBot is a knowledge assistant that is centralized

and can be used to guide on the functionality of the application, security warnings and suggested actions that eventually can result in a more intuitive and user-friendly platform.

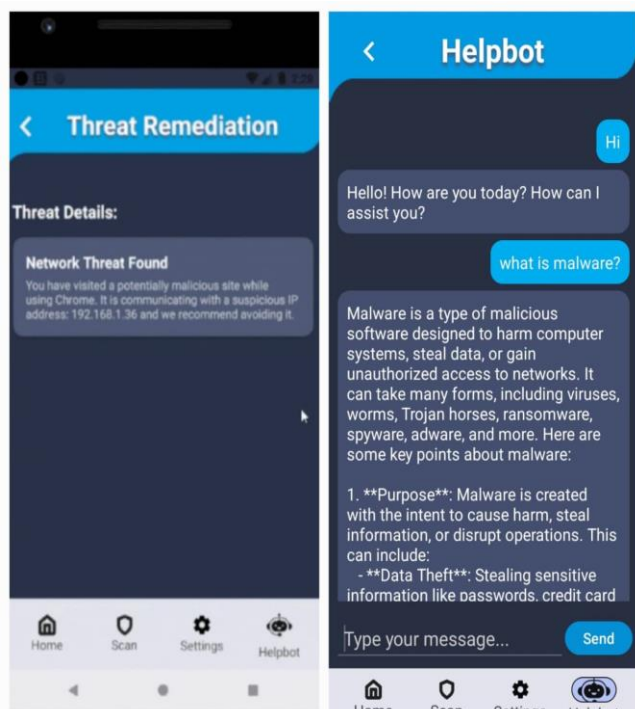


Fig. 6. Threat Remediation and Help bot

5. Experimental Setup

We trained our neural network on a mixed dataset of TUANDROMD which is a representation of newer malware and DREBIN which is a collection of older samples of malware [1][2]. The figure below shows a distribution of malware and benign samples in this dataset with almost equal distribution which practically dispels any class imbalance worries.

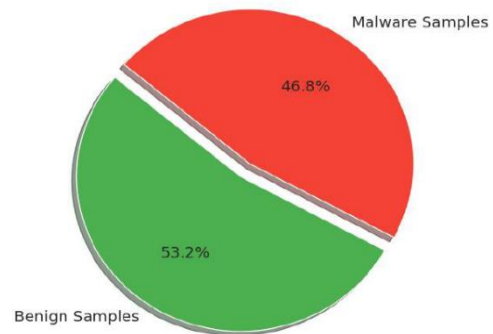


Fig. 7. Distribution of Benign and Malware Samples in our Hybrid Dataset

The training of the model was done in 27 epochs. The figures below show the loss and AUCs during training. We find that the loss changes downward steadily, and almost reaches a point at the end, which means that learning is effective, and the change of the AUC is gradual, and it also approaches a point which means that there is better and consistent performance.

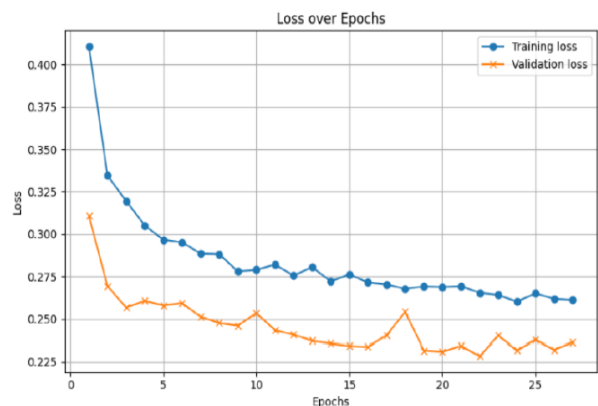


Fig. 8. Loss over Epochs

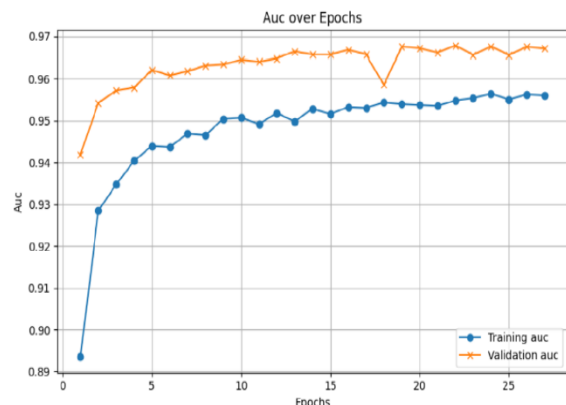


Fig. 9. AUC over Epochs (Area Under the Curve)

The confusion matrix of the test results is as shown in the following figure. Among 2087 samples of goodware, 1973 were correctly identified and 114 mistakenly identified as malware. In case of the 1813 samples of malware, 1538 samples were classified well and 275 had been wrongly classified as good ware.

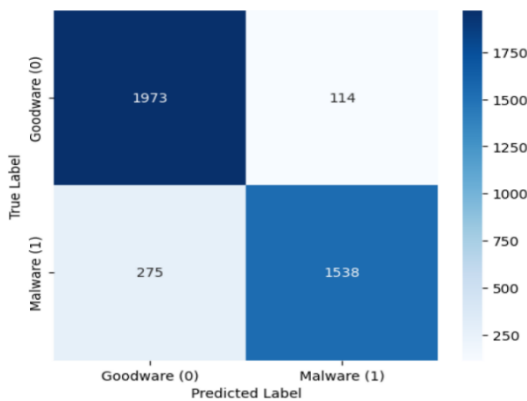


Fig. 10. Confusion Matrix

All in all, the model shows a relatively low false positive and false negative, which indicates a high level of classification. The table below represents the visualization of the classification performance of the malware detection model. It reveals accuracy, recall as well as F1-score of the two classes: Good ware and Malware [1][2]. The model had a high recall with Good ware (0.95), i.e., it recognized the majority of the benign apps, whereas the recall on Malware (0.85) represents the fact that it failed to recognize some samples of malware. The precision is a little better with Malware (0.93) and indicates that the majority of the malware samples that were predicted were malicious. The balance of the overall macro and weighted averages of all metrics is at 0.90 which shows a steady result in both classes.

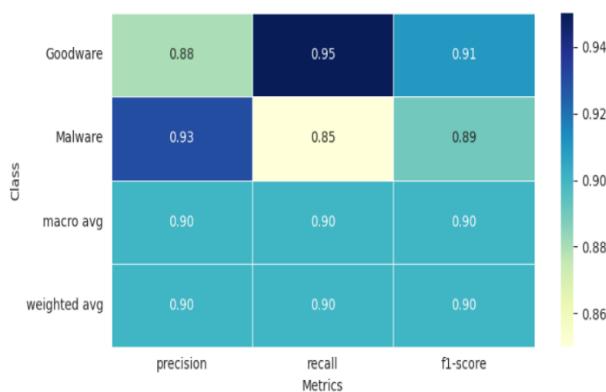


Fig. 11. Classification Report Heatmap

6. CONCLUSION AND FUTURE SCOPE

This introduced design and development of CyberSentinel, a smart Android security application which is a combination of machine learning-based malware detection system and real-time network monitoring and threat remediation system. We started with the description of Android Security Model, the analysis of existing models of permissions, and the survey of popular types of malware. The comprehensive comparative analysis of the commercial mobile security solutions has shown the main gaps existing in the current tools specifically absence of transparency, on-gadget intelligence, and user-oriented design. In order to overcome these problems, we trained a neural network on a variety of datasets (DREBIN, TUANDROMD, COLCOM) and constructed permission-based features in order to provide high generalization quality with unseen malware. We also combined CyberSentinel with a modular Android application which has features like Network Center, HelpBot assistant, and threat remediation. Lots of testing have showed that the system is capable of detecting threats with high accuracy, precision and yet it is lightweight and easy to use. CyberSentinel proves that implementing AI-based detection along with contextual awareness and system-level visibility can help enhance proactive mobile security to a great extent. The project offers a new security solution as well as customizable platform in the future research of adaptive on-device threat defense.

We intend to add application-level network scanner support to Android 10 and beyond devices in the future by investigating other options that meet new security controls, such as VPN-based traffic inspection and Android network monitoring APIs. Moreover, we are also planning to add the Help Bot module, which will have the voice interaction option, so a user will be able to be guided and operate any necessary action, based on a voice command. This will ensure that Cyber Sentinel is more convenient and approachable to the users who might have a more favourable reaction to hands-free use or need any kind of an assistive tool.

7. REFERENCES

- [1] ReDroidDet – Android Malware Detection (2021). https://www.researchgate.net/publication/351682116_ReDroidDet_Android_Malware_Detection_Based_on_Recurrent_Neural_Network
- [2] Hybrid Android Malware Detection (2025). <https://link.springer.com/article/10.1007/s44196-025-00783-x>

- [3] Android Traffic Malware Analysis (2024). <https://www.sciencedirect.com/science/article/pii/S20944792400515X>
- [4] AntMonitor-App-LevelMonitoring (SIGCOMM)(2021). <https://conferences.sigcomm.org/sigcomm/2015/pdf/papers/c2bid/p15.pdf>
- [5] Wi-Fi Easy Connect Security Analysis (2023). https://www.researchgate.net/publication/388801920_Security_analysis_of_the_Wi-Fi_Easy_Connect
- [6] The Rise of Public Wi-Fi and Threats (2023). https://www.researchgate.net/publication/375230379_The_Rise_of_Public_Wi-Fi_and_Threats
- [7] Broadband Connectivity - Speedtests (2023). https://www.researchgate.net/publication/373779469_Understanding_the_State_of_Broadband_Connectivity_An_Analysis_of_Speedtests_and_Emerging_Technologies
- [8] Ookla vs NDT7 Comparison (2023). https://www.researchgate.net/publication/371934889_A_Comparative_Analysis_of_Ookla_Speedtest_and_Measurement_Labs_Network_Diagnostic_Test_NDT7
- [9] LLMs in Cybersecurity (2024). https://www.researchgate.net/publication/383064112_Large_Language_Models_for_Cyber_Security_A_Systematic_Literature_Review
- [10] Domain-Specific Chatbots (2022). https://www.researchgate.net/publication/362797107_Domain-Specific_Chatbot_Development_Using_the_Deep_Learning-Based_RASA_Framework
- [11] K. Jolly, S. Petersen, and B. Connelly, ooklaOpenDataR: Work with 'Speedtest by Ookla' Global Internet Performance Data, 2022. R package version 0.1.0.
- [12] Ranya Sharma, Tarun Mangla, James Saxon, Marc Richardson, Nick Feamster, and Nicole P Marwell. 2022. Benchmarks or Equity? A New Approach to Measuring Internet Performance. A New Approach to Measuring Internet Performance (August 3, 2022) (2022).
- [13] Xinlei Yang, Xianlong Wang, Zhenhua Li, Yunhao Liu, Feng Qian, Liangyi Gong, Rui Miao, and Tianyin Xu. 2021. Fast and Light Bandwidth Testing for Internet Users.. In NSDI.
- [14] Tareq M, Safeia A (2021) Man in the middle attack in wireless-network. <https://www.researchgate.net/publication/356290033> Access on November 2022
- [15] Maimon D, Howell CJ, Jacques S, Perkins RC (2022) Situational awareness and public Wi-Fi users' self-protective behaviors. *Secur J* 35(1):154-174. <https://doi.org/10.1057/s41284-020-00270-2>
- [16] Agarwal A, Kumari V, Sharma Y, Goel L (2021) Ranking based question answering system with a web and mobile application. In: 2021 11th International conference on cloud computing, data science & engineering (Confluence). IEEE, New York, pp 52-58