



# INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

## Malware Detection Web App Based On Hybrid Analysis And Deep Learning Techniques

Lubdha Pashte, Siddhi Patade, Anjali Pathak, Prof. Poonam Vengulekar

Department of Computer Science and Technology

Usha Mittal Institute of Technology SNTD Women University,

Juhu Tara Road, Sir Vithaldas Thackersey, Santacruz(West) Mumbai, Maharashtra -400049.

### Abstract

The Metamal-D-Alert project is an advanced malware detection system designed to address the rising threats of malware, which often lead to severe data breaches, financial losses, and operational disruptions. This initiative employs sophisticated machine learning techniques, specifically Convolutional Neural Networks (CNNs) and Bidirectional Gated Recurrent Units (BiGRU), to identify abnormal patterns indicative of malware. Using the custom MetaMal-D-Alert dataset for Windows environments, the system effectively detects both known and zero-day malware attacks. Implemented as a scalable web application with Docker containers, MetaMal-D-Alert ensures seamless integration and deployment across various platforms. Enhanced by API analysis and Natural Language Processing (NLP), it offers comprehensive dynamic and hybrid threat analysis. Experimental results show that Metamal-D-Alert significantly outperforms traditional signature-based methods, providing superior accuracy and efficiency in identifying malicious activities.

**Keywords:** Malware detection, Neural Networks, API calls, dynamic analysis, Hybrid analysis API, Cybersecurity, Machine Learning, NLP, Deep Learning.

### Introduction

Malware continues to be a significant and evolving threat to computer systems and networks, leading to data breaches, financial fraud, and service disruptions.

With the number of devices connected to IP networks projected to reach 29.3 billion by 2025 and over 29 billion IoT devices by 2030, cybercriminals are increasingly driven to create sophisticated malware. These malicious programs disrupt operations, steal sensitive information, cause damage, and exploit vulnerabilities, affecting millions of devices with worms, ransomware, Trojans, and other malware forms. Traditional signature-based malware detection systems, which rely on known malware signatures, are losing effectiveness as they struggle to identify new, obfuscated malware variants. Consequently, static analysis techniques such as examining printable strings, opcode sequences, and API calls are proving inadequate against the complexities of modern malware challenges.

Motivated by the limitations of traditional malware detection methods, this project introduces a comprehensive malware detection web application leveraging cutting-edge technologies. Central to our approach is a user-friendly web interface that allows easy file submission, with backend processing handled securely within Docker containers. The system employs advanced techniques such as API analysis and Natural Language Processing (NLP) to analyze file behaviour and context, significantly enhancing the detection accuracy of various malware variants. Future updates will expand support to more operating systems and file formats and incorporate emerging technologies to stay ahead of evolving cyber threats. This project

represents a significant step forward in creating a more secure digital environment by improving malware detection accuracy and adaptability.

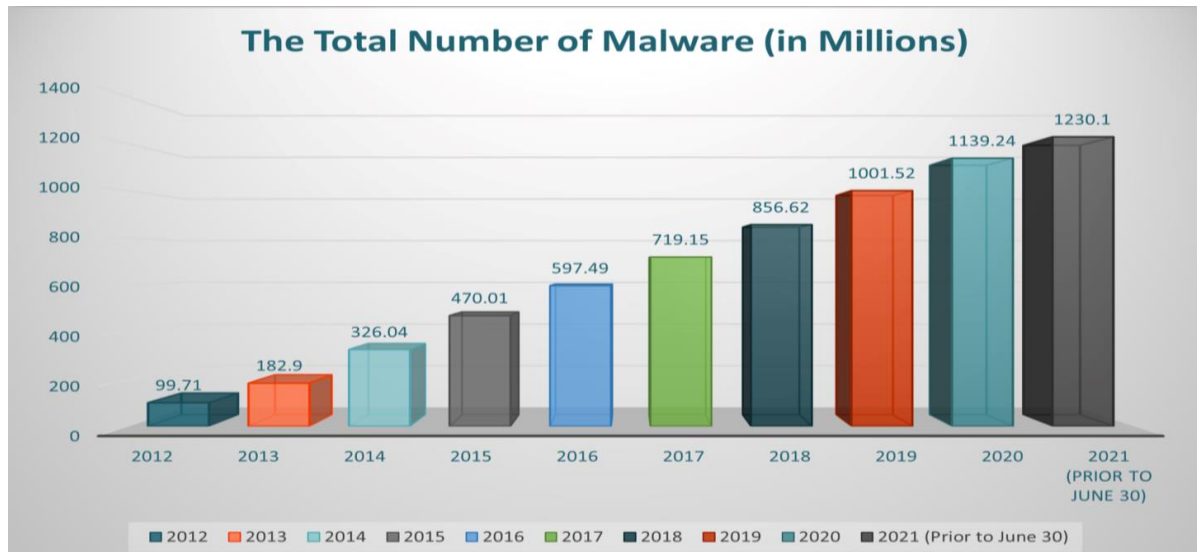


Figure 1. Number of malware attacks report till year 2021.

## Problem Statement

Existing malware detection systems face significant challenges, including accuracy issues, particularly for Windows OS. Static analysis, which examines code without execution, often fails against obfuscated and polymorphic malware that alter their appearance to evade detection. Dynamic analysis, which observes malware behaviour in a controlled environment, can be bypassed by malware designed to detect and evade these environments. Additionally, many current systems are outdated and unable to accurately identify new and sophisticated threats and accuracy limitations. Our project directly addresses these limitations by developing a cutting-edge malware detection system tailored for Windows OS, leveraging Docker containers for deployment. This approach offers better isolation and performance, overcoming the drawbacks of traditional VM-based analysis. The system features a user-friendly web interface for easy file uploads, making it accessible for users with varying technical expertise.

Additionally, our solution incorporates advanced APIs and Natural Language Processing (NLP) for thorough

file analysis, identifying both known and emerging malware threats. This comprehensive approach ensures proactive defence against evolving cyber threats, enhancing the overall cybersecurity posture. By providing timely detection and mitigation, our system significantly reduces the risk and impact of malware infections, safeguarding digital assets more effectively than existing solutions.

**Related work:** This section presents the gist of some seminal and some recent papers which work around malware detection using machine learning algorithms.

a. Static and Dynamic-based malware detection techniques

Recent research has delved into various malware detection strategies, focusing on static, dynamic, hybrid, and memory analysis to understand and preemptively address malware operations effectively. Static analysis has traditionally been favoured for its straightforward approach to examining malware structures without executing them, which allows for the identification of unique malware signatures. However, its efficacy is limited when dealing with obfuscated malware, which conceals its true nature through sophisticated techniques like packing, making static

analysis insufficient for detecting such advanced threats. To circumvent these shortcomings, dynamic analysis has been utilized, providing the ability to execute malware and scrutinize its behaviours in real-time, thereby enhancing the detection of obfuscated and newly emerging malware variants.

In the realm of dynamic analysis, significant advancements have been made as evidenced by studies such as the one described in the paper ‘Automated Behaviour-based Malware Detection Framework Based on Natural Language Processing and Deep Learning Technique’. This research introduces MalBehavD-V1, a dataset of dynamic Windows API calls, and MalDetConV, a framework that integrates a

text processing-based encoder with a hybrid CNN-BiGRU architecture to boost detection capabilities. The framework has proven highly effective in identifying both known and zero-day malware threats across various datasets, marking a substantial improvement over traditional method. Another noteworthy study, ‘Machine Learning Based Solution for the Detection of Malicious JPEG Images’, presents MalJPEG, a machine learning approach tailored to detect malicious JPEG files. Despite using a minimal feature set, MalJPEG demonstrates superior efficacy and speed compared to traditional feature extraction methods, outperforming even leading antivirus solutions in detecting sophisticated cyber threats, thereby showcasing the potential of specialized machine learning models in enhancing cybersecurity measures.

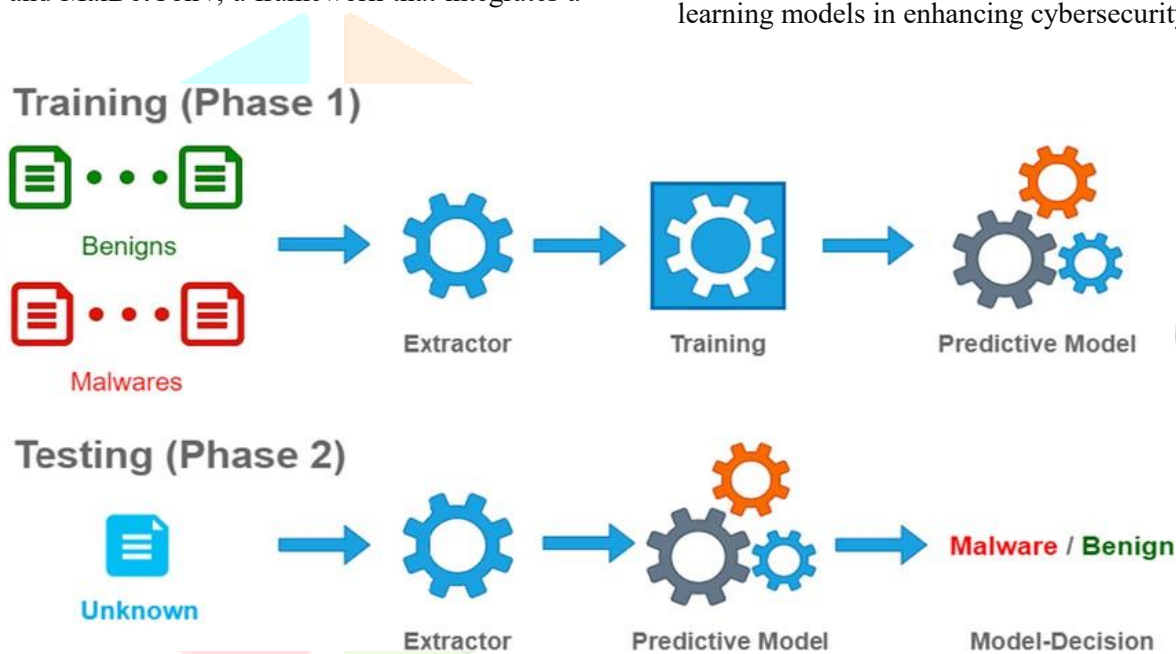


Figure 2. Phases of development of malware detection system

### Proposed work

Our proposed work aims to develop a comprehensive and user-friendly web application using the MERN stack (Express.js, React, and Node.js) for dynamic malware analysis. By integrating this application with our backend using Axios for API calls, we ensure seamless communication and data handling. This initial stage of implementation provides users with an intuitive interface for uploading files and accessing analysis results.

Our backend system for dynamic malware analysis combines natural language processing (NLP), convolutional neural networks (CNNs), and bidirectional gated recurrent units (BiGRUs) with the help falcon sandbox. The goal is to generate dynamic reports based on API call sequences from both benign and malicious programs, thus enhancing malware detection efficacy. These dynamic/behavioural reports originate from analyzing API call sequences within the sandbox environment, which are represented as text and encoded via a word embedding model to create dense vector

representations for each API call. These vectors are then fed into a CNN-BiGRU hybrid feature extractor for in-depth analysis.

The architecture of our proposed framework, illustrated in Figures 3 and 4, consists of six core components/modules: the executable files collection module, behaviour monitoring module, pre-processing module, embedding module, hybrid automatic feature extraction module, and classification module.

1. Executable Files Collection Module: This module gathers executable files for analysis.
2. Behaviour Monitoring Module: It monitors the behaviour of these files within the sandbox environment.
3. Pre-processing Module: This module prepares the API call sequences for further analysis.
4. Embedding Module: It converts these API call sequences into dense vector representations using word embedding techniques.
5. Hybrid Automatic Feature Extraction Module: This module extracts features from the vector

representations using a combination of CNN and BiGRU models.

6. Classification Module: Based on the extracted features, this module classifies executable files as benign or malicious.

These modules collectively empower the web application to detect malicious executable files by scrutinizing their dynamic behaviour and API call sequences. Capturing and analyzing API call sequences from benign and malicious executable files during dynamic analysis relies heavily on the sandbox environment as an analysis testbed. Automated sandboxes are crucial for monitoring malware behaviour during runtime, thereby protecting the host system or production environment from potential damage.

By integrating advanced technologies and robust architecture, our proposed work significantly improves malware detection accuracy and offers a proactive approach to identifying and mitigating cybersecurity threats.

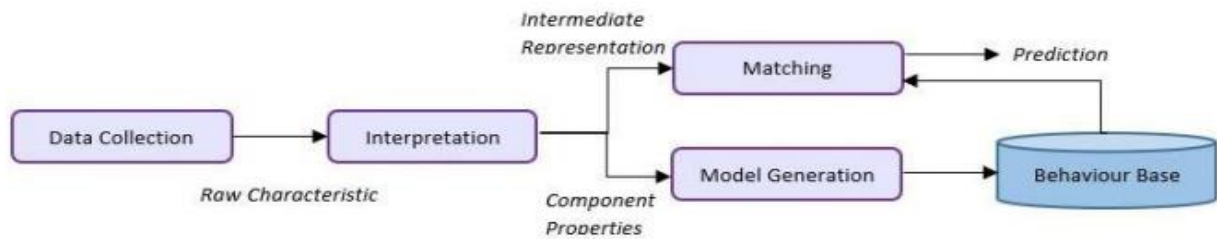
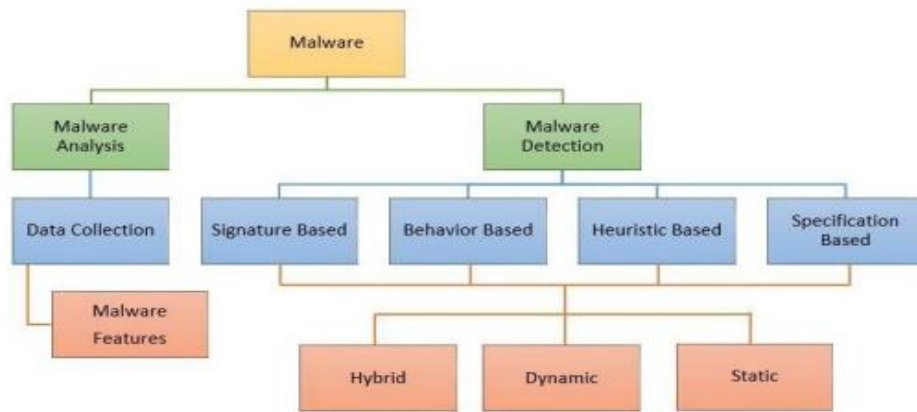


Figure 3. Initial approach and Malware detection model.

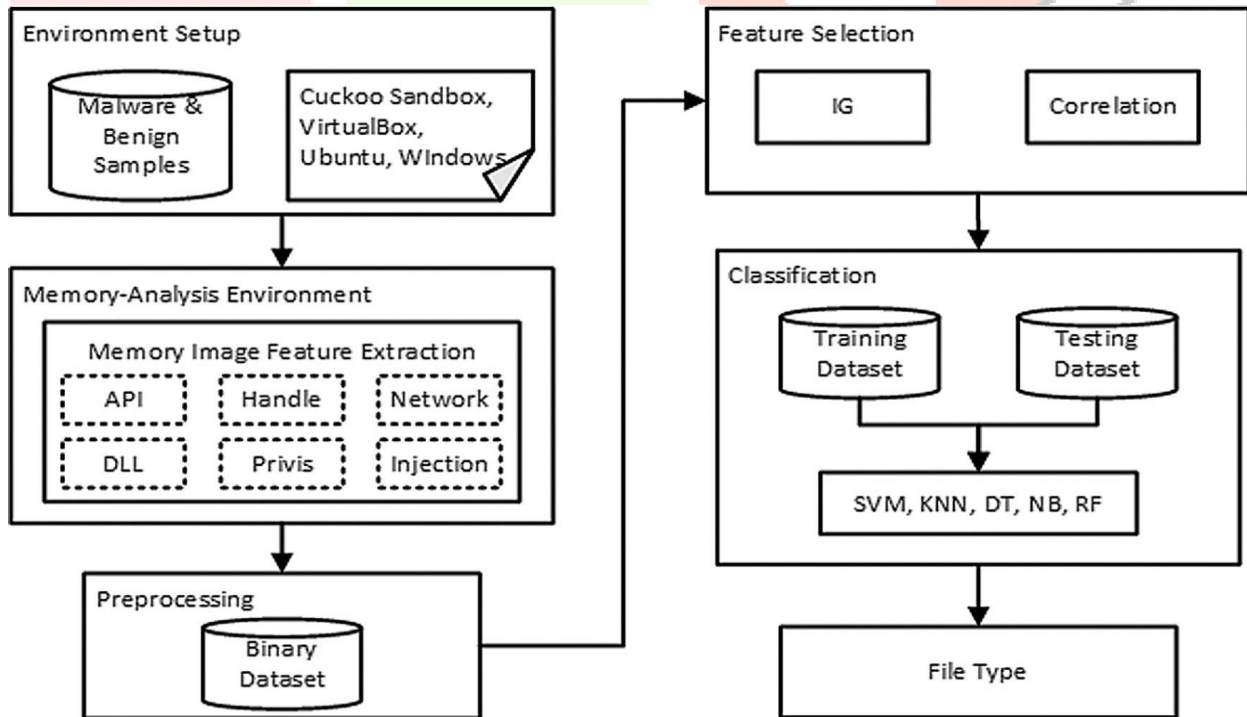


Figure 3.1 - Machine learning and NLP-based malware detection model.

## METAMAL-D-ALERT

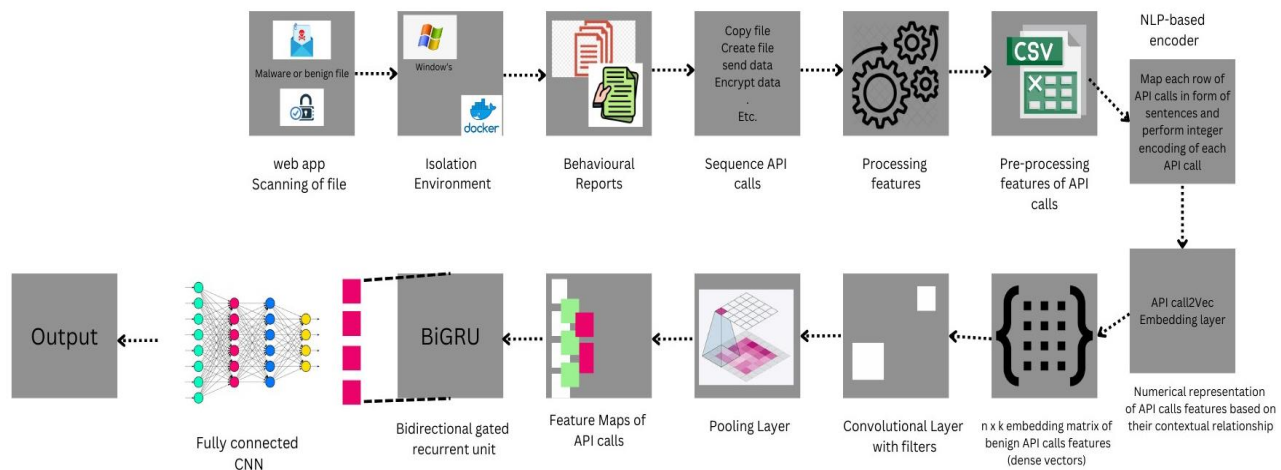


Figure 4. Architecture of the proposed system.

### Methodology

This paper presents a comprehensive methodology for developing a user-friendly web application for dynamic malware detection, leveraging the MERN stack (Express.js, React, Node.js) and integrating it with advanced backend technologies. The methodology encompasses both frontend and backend development, along with Docker-based deployment on a Windows host machine.

#### Web Application Development

- 1. Frontend Development:** The frontend of the web application is developed using React, providing an intuitive and responsive interface for users to upload files for malware analysis. The interface is designed to facilitate easy navigation and file management.
- 2. Backend Integration:** The backend, developed using Node.js and Express.js, handles file uploads, processing, and communication with the analysis modules. Axios is employed for making API calls to ensure seamless integration between the frontend and backend.

### Backend Analysis System

- 1. Dynamic Malware Analysis:** The backend system integrates natural language processing (NLP), convolutional neural networks (CNNs), and bidirectional gated recurrent units (BiGRUs). This setup is inspired by the functionality of Falcon Sandbox, aiming to generate dynamic reports based on API call sequences from both benign and malicious programs.
- 2. Data Collection and Processing:**
  - **Executable Files Collection Module:** Gathers executable files for analysis.
  - **Behaviour Monitoring Module:** Monitors the behaviour of these files within the sandbox environment.
  - **Pre-processing Module:** Prepares the API call sequences for further analysis.
  - **Embedding Module:** Converts these sequences into dense vector representations using word embedding techniques.

### 3. Feature Extraction and Classification:

- Hybrid Automatic Feature Extraction Module:  
Utilizes CNN and BiGRU models to extract features from the vector representations.

- Classification Module: Classifies executable files as benign or malicious based on the extracted features.

```
Epoch 1/11
33/33 [=====] - 48s 1s/step - loss: 0.6910 - accuracy: 0.5034 - val_loss: 0.6858 - val_accuracy: 0.6809 - lr: 1.0000e-04
Epoch 2/11
33/33 [=====] - 36s 1s/step - loss: 0.6709 - accuracy: 0.7500 - val_loss: 0.6308 - val_accuracy: 0.7821 - lr: 1.0000e-04
Epoch 3/11
33/33 [=====] - 37s 1s/step - loss: 0.4932 - accuracy: 0.8551 - val_loss: 0.2596 - val_accuracy: 0.9144 - lr: 1.0000e-04
Epoch 4/11
33/33 [=====] - 48s 1s/step - loss: 0.2238 - accuracy: 0.9163 - val_loss: 0.1963 - val_accuracy: 0.9183 - lr: 1.0000e-04
Epoch 5/11
33/33 [=====] - 52s 2s/step - loss: 0.1843 - accuracy: 0.9348 - val_loss: 0.1843 - val_accuracy: 0.9202 - lr: 1.0000e-04
Epoch 6/11
33/33 [=====] - 40s 1s/step - loss: 0.1530 - accuracy: 0.9484 - val_loss: 0.1867 - val_accuracy: 0.9202 - lr: 1.0000e-04
Epoch 7/11
33/33 [=====] - 41s 1s/step - loss: 0.1464 - accuracy: 0.9470 - val_loss: 0.1540 - val_accuracy: 0.9377 - lr: 1.0000e-04
Epoch 8/11
33/33 [=====] - 40s 1s/step - loss: 0.1394 - accuracy: 0.9499 - val_loss: 0.1790 - val_accuracy: 0.9377 - lr: 1.0000e-04
Epoch 9/11
33/33 [=====] - 39s 1s/step - loss: 0.1222 - accuracy: 0.9587 - val_loss: 0.1880 - val_accuracy: 0.9358 - lr: 1.0000e-04
Epoch 10/11
33/33 [=====] - 40s 1s/step - loss: 0.1376 - accuracy: 0.9533 - val_loss: 0.1448 - val_accuracy: 0.9436 - lr: 1.0000e-04
Epoch 11/11
33/33 [=====] - 49s 1s/step - loss: 0.1042 - accuracy: 0.9635 - val_loss: 0.1574 - val_accuracy: 0.9436 - lr: 1.0000e-04
```

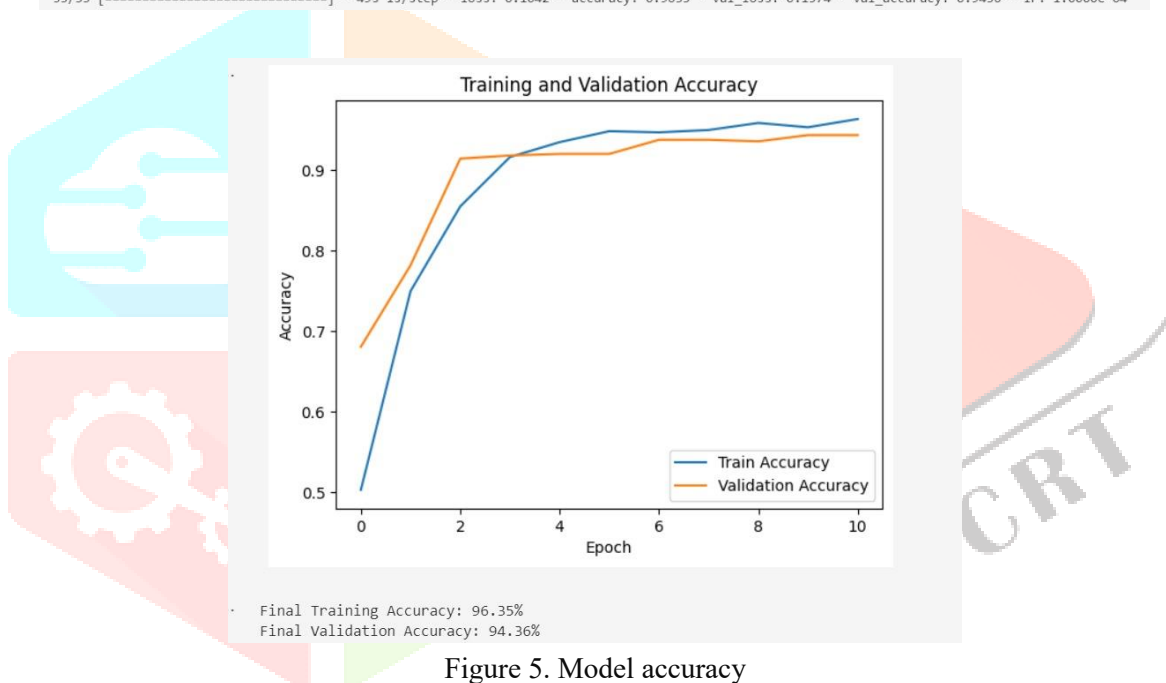


Figure 5. Model accuracy

### Docker-Based Deployment

1. Docker Setup: The initial step involves installing and configuring Docker Desktop on the Windows host machine. Docker Desktop provides a user-friendly interface for managing containers and images, making it suitable for this project.

2. Container Orchestration: Docker Compose is employed to manage the lifecycle of Docker

containers. It allows defining multi-container applications in a single YAML file and launching them with a single command, simplifying the deployment process.

3. Testing and Deployment: Rigorous local testing is conducted to ensure all components function harmoniously. Once validated, the Docker setup is deployed with confidence, ensuring that the system performs as expected in production environments.

n	Training Accuracy (%)	Testing Accuracy (%)
20	99.17%	93.77%
40	99.44%	94.03%
60	99.72%	95.19%
80	99.56 %	95.45%
100	99.72%	96.10%

Table 1. Detection accuracy achieved by MetaMal-D-Alert dataset of API call sequence(n).

4. Monitoring and Maintenance: Continuous monitoring of Docker container performance is essential. Adjustments to resource allocation and configurations are made as needed. Regular updates to Docker images and containers are performed to incorporate security patches and new features, maintaining system security and effectiveness.

By integrating these components, the web application effectively captures and analyzes API call sequences from benign and malicious executable files. The sandbox environment serves as an analysis testbed, ensuring safe monitoring of malware behaviours during runtime. This approach offers significant improvements in malware detection accuracy, enabling proactive identification and mitigation of cybersecurity threats.

**Comprehensive Dynamic Malware Detection**

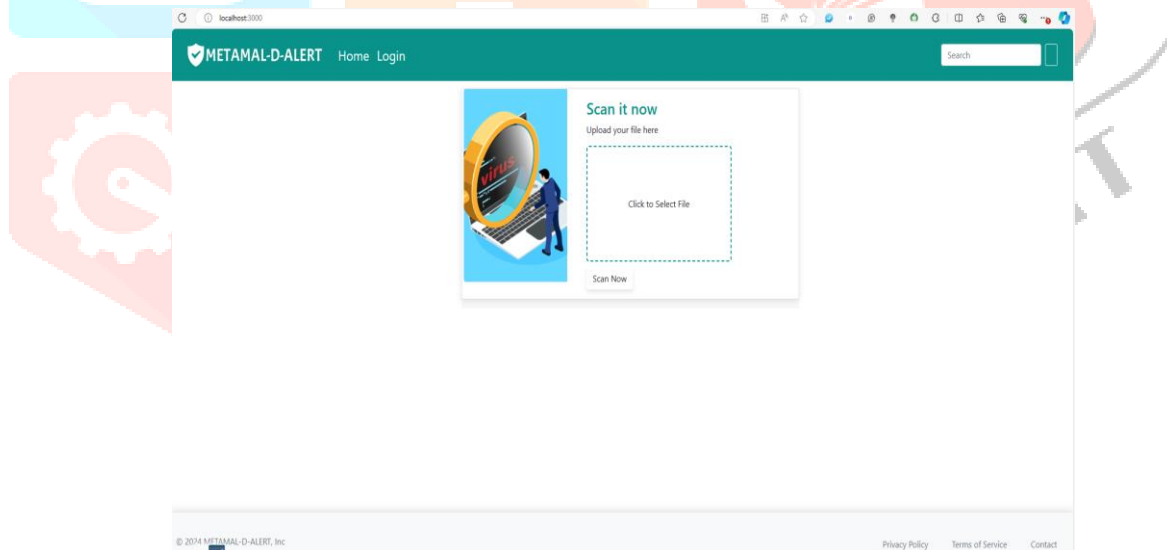


Figure 6. Image depicting our web application front page.

**6. Results and Analysis**

The malware detection web application effectively displays the file name, results, and suggestion after scanning. Utilizing advanced technologies such as deep learning, CNNs, and BiGRUs with the help of a felcon sandbox, the system provides detailed behavioural analysis reports that highlight suspicious

patterns and anomalies. This integration significantly improves detection accuracy and efficiency, outperforming traditional methods. The Docker-based deployment model ensures the system's scalability and maintainability. The user-friendly interface built with the MERN stack allows for seamless file uploads, responsive analysis, and enhancing the overall user experience.



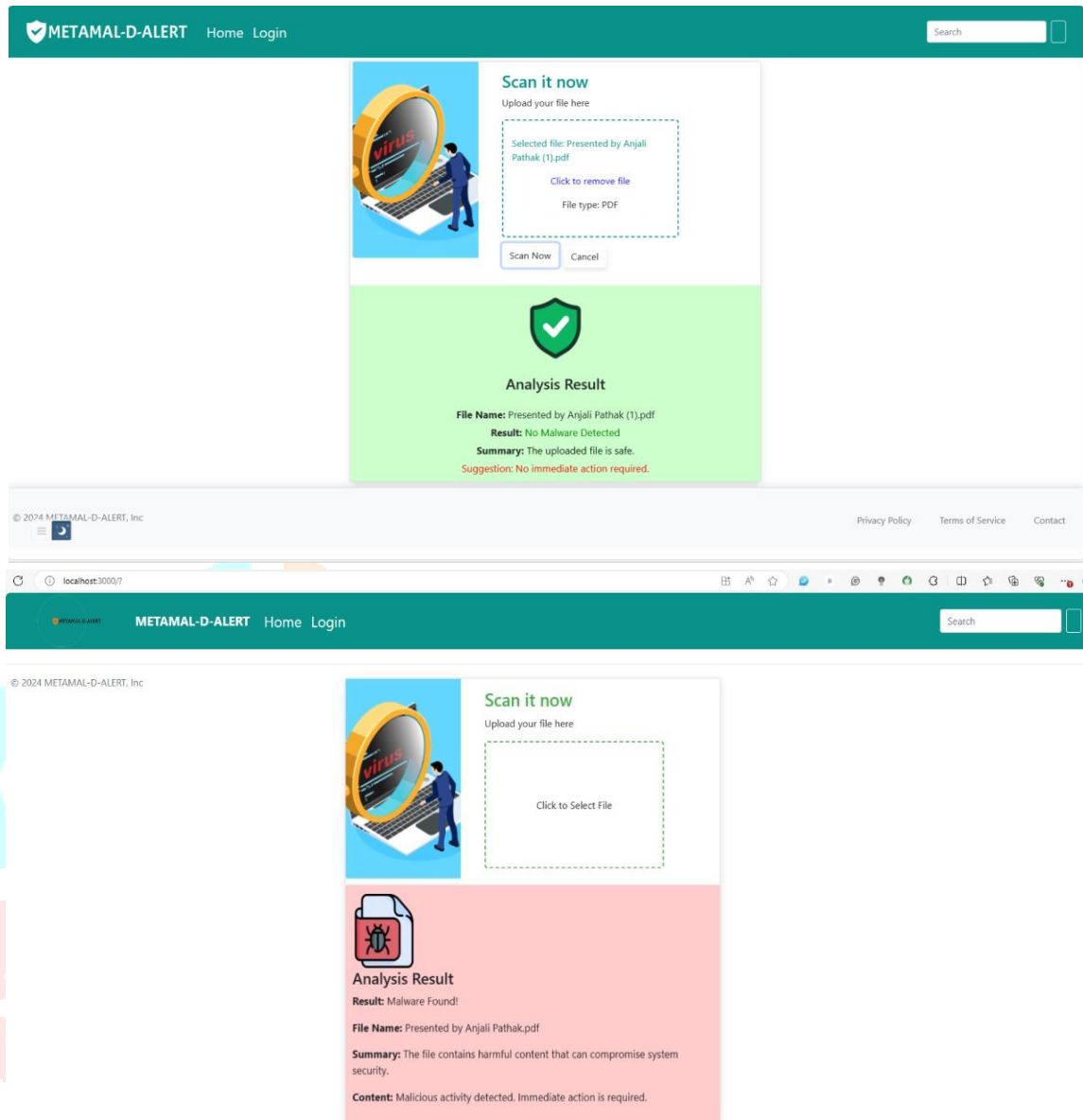


Figure 7. Image depicting our result after scanning the file.

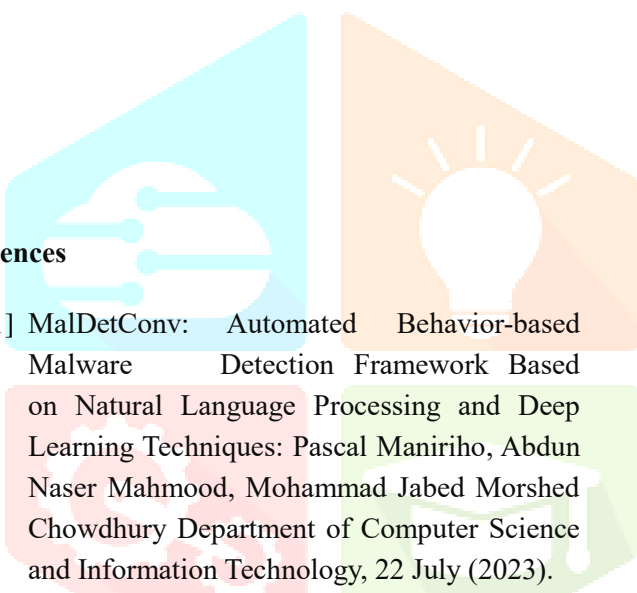
### Limitations and Future Work

Our proposed dynamic malware detection framework, while promising, faces several notable limitations. Firstly, the system's efficacy relies heavily on the fidelity and coverage of the sandbox environment, introducing variability and potential limitations in capturing complex malware behaviours accurately. Scalability concerns arise as the volume of executable files increases, potentially impacting system performance under high loads. Despite advanced feature extraction and classification, the system may still exhibit false positives or negatives,

posing challenges to user trust and reliability. Data privacy and security are paramount, necessitating stringent measures to protect user data during analysis. Addressing these limitations will be pivotal in realizing the full potential of our framework in combating cybersecurity threats. In our future work, we intend to expand its functionality beyond the current capability of scanning files for malware detection. We aim to incorporate a feature that allows users to input URLs, enabling the application to analyze these links for potential malware threats. This enhancement will broaden the scope of our security measures, providing users with a

comprehensive tool for both file and URL-based threat detection. We intend to enhance the system by introducing a robust backup and recovery strategy is essential to safeguard data and configurations against system failures or corruption. Also, when dealing with huge datasets, the accuracy and result may both be less accurate as well as computationally costly.

## References

- 
- [1] MalDetConv: Automated Behavior-based Malware Detection Framework Based on Natural Language Processing and Deep Learning Techniques: Pascal Maniriho, Abdun Naser Mahmood, Mohammad Javed Morshed Chowdhury Department of Computer Science and Information Technology, 22 July (2023).
- [2] DMalNet: Dynamic malware analysis based on API feature engineering and graph learning, Institute of Information Engineering, Chinese Academy of Science, Beijing, China, (21 August 2022).
- [3] L. Dhanya, R. Chitra, A. A. Bamini, Performance evaluation of various ensemble classifiers for malware detection, Materials Today: Proceedings (2022).
- [4] Detection of Malware by Deep Learning as CNN-LSTM Machine Learning Techniques in Real Time, School of Computer and Communication, Lanzhou University of Technology, Lanzhou 730050, China, (2022).
- [5] Deep learning based Sequential model for malware analysis using Windows exe API Calls, research gate: (2021).

## Conclusion

In summary, the "MetaMal-D-Alert" project aims to revolutionize malware detection through a user-friendly web application that integrates Docker containers, APIs, and NLP-based techniques, alongside advanced machine learning. This approach enhances the capability to identify malware in files swiftly and accurately, thus ensuring a more secure operating environment for users. The project also introduces a novel dataset of Windows API calls, and employs a behaviour-based web application leveraging a hybrid CNN-BiGRU architecture for meticulous malware detection and classification. By overcoming the shortcomings of traditional methods, the "MetaMal-D-Alert" offers a powerful solution capable of operating within high-speed network infrastructures to quickly and efficiently detect both known and emerging malware threats, demonstrating exceptional accuracy and precision.

- [6] Ö. Aslan, M. Ozkan-Okay and D. Gupta, "A review of cloud-based malware detection system: Opportunities advances and challenges", (2021).
- [7] Windows PE Malware Detection Using Ensemble Learning Nureni Ayofe Azeez, Oluwanifise Ebunoluwa Odufuwa, Sanjay Misra, Jonathan Oluranti and Robertas Damaševičius, (2021).
- [8] P. P. Kundu, L. Anatharaman, T. Truong-Huu, An empirical evaluation of automated machine learning techniques for malware detection, in Proceedings of the 2021 ACM Workshop on Security and Privacy Analytics, (2021).
- [9] R. Komatwar and M. Kokare, "A survey on malware detection and classification", J. Appl. Secur. Res., Aug. (2020).
- [10] K.O'Shea, R. Nash, An introduction to convolutional neural networks, arXiv preprint arXiv:1511.08458 (2015).
- [11] F. Ceschin, F. Pinage, M. Castilho, D. Menotti, L. S. Oliveira, A. Gregio, The need for speed: An analysis of brazilian malware classifiers, IEEE Security & Privacy 16 (6) (2018).