AMERICAN JOURNAL OF AI CYBER COMPUTING MANAGEMENT
Peer Reviewed, Rferred & Indexed Journal
E-ISSN:3069-0102
VOL.6, NO. 2(2026)

A HIERARCHICAL THREAT INTELLIGENCE SYSTEM FOR SCALABLE MALWARE DETECTION

Chitla Rahul[1], T. Sanath Kumar[2*], Durishetti Anjali[1], Bodapatla Rajesh[1], Dudapaka Goutham[1], Bukya Arun[1]

[1]UG Student, [2]Assistant Professor, [1,2]Department of Computer Science and Engineering (AI&ML)

[1,2]Vaagdevi Engineering College, Bollikunta, Warangal, 506005, Telangana, India

*Correspondence: T. Sanath Kumar (sunny.554@gmail.com)

Abstract

The rapid expansion of digital technologies and internet-based applications has significantly increased exposure to cyber threats, particularly malware attacks that compromise data security and system integrity. Traditional malware detection systems have relied on signature-based and rule-based techniques, which identify malicious files using predefined patterns. While effective for known threats, these approaches are inadequate for detecting new, evolving, and obfuscated malware, leading to reduced accuracy and increased system vulnerability. Additionally, conventional methods struggle to process high-dimensional data and handle class imbalance commonly present in real-world cybersecurity datasets. To overcome these challenges, this study proposes a machine learning–based malware classification framework that integrates data preprocessing, exploratory data analysis, and multi-model classification techniques. The system employs classifiers such as Support Vector Machine (SVM), Gaussian Naive Bayes (GNB), and Multinomial Naive Bayes (MNB), along with a novel hybrid model called GaussTree-Stack (HGTS). The HGTS model combines Decision Tree (DT) with GNB to leverage both decision-based and probabilistic learning capabilities. To further enhance performance, the Synthetic Minority Over-sampling Technique (SMOTE) is applied to address class imbalance and improve model generalization. Experimental results demonstrate that the proposed HGTS model achieves a high accuracy of 97.06%, along with strong precision, recall, and F1-score values, outperforming individual classifiers. The system is implemented with a user-friendly graphical interface and model persistence for efficient real-time predictions. This research provides a scalable, accurate, and reliable solution for malware classification, contributing to improved cybersecurity and proactive threat detection in modern digital environments.

Keywords: cybersecurity, machine learning, malware detection, SMOTE, threat analysis, data balancing.

1. Introduction

Cybersecurity threats have grown rapidly over the past decade, with malware emerging as one of the most critical risks affecting individuals, organizations, and governments worldwide. Reports indicate that hundreds of thousands of new malware variants are detected daily, creating a highly dynamic and challenging threat landscape [1]. This continuous evolution of malicious software makes it difficult to ensure complete system security using conventional approaches. Large-scale datasets, such as those provided by Microsoft, contain diverse samples across multiple malware families, reflecting the complexity and variability of modern cyber threats [2]. These datasets play a vital role in understanding malware behavior and developing effective defense mechanisms.

Modern approaches focus on handling the scale and diversity of malware by organizing classification processes into structured and efficient frameworks [3]. Such strategies help in distinguishing between closely related malware families while maintaining computational efficiency. The availability of extensive real-world datasets enables researchers to design solutions that align with practical

cybersecurity requirements and evolving threat patterns [4]. Traditional detection techniques often rely on predefined signatures, which become ineffective when encountering new or unknown malware variants. In such cases, unidentified threats may be mistakenly treated as legitimate, increasing system vulnerability. To overcome these limitations, advanced detection methods based on traffic analysis and anomaly identification have been explored. These methods identify suspicious activities by analyzing patterns in system behavior and network traffic. By focusing on deviations from normal patterns, they provide a more adaptive approach to threat detection, addressing the shortcomings of conventional systems and improving overall cybersecurity resilience.

Problem Definition: The increasing sophistication and diversity of malware create significant challenges for existing detection systems, which often rely on outdated or static approaches. These systems struggle to accurately identify new and evolving threats, leading to reduced effectiveness and increased vulnerability. Additionally, real-world datasets are often large, complex, and imbalanced, making it difficult to build models that perform well across all categories. High-dimensional data further complicates the analysis process, increasing computational requirements and affecting performance. Many systems also lack efficient integration between data processing, model training, and prediction, limiting their usability in real-time scenarios. These issues highlight the need for improved methods to handle complex data and enhance the accuracy and efficiency of malware classification.

2. LITERATURE SURVEY

The study in [5] provides a comprehensive evaluation of NGFW technologies in telecom networks, emphasizing features such as deep packet inspection, intrusion prevention systems, and threat intelligence integration. The study in [6] explores the integration of AI and machine learning (ML) into NGFWs, anomaly detection, and behavioral analysis to enhance threat detection. Although it outlines a broad range of applications, the work remains theoretical and provides limited details on implementation. The study in [7] presents a comprehensive overview of threats and defenses in Industrial Internet of Things (IIoTs) and edge environments, identifying AI/ML, federated learning, and blockchain as key solutions. This aligns with our research focus on enhancing edge security through advanced techniques. However, although the study emphasizes AI and edge computing, it does not explicitly investigate the use of NLP for malware detection, which is central to our research on integrating NLP and ensemble learning into NGFWs.

The study in [8] examines AI-driven firewalls in cloud environments, focusing on advanced intrusion-detection and adaptive-response mechanisms. It highlights dynamic resource provisioning and load balancing as key strategies for enabling real-time threat detection without overloading systems. The study in [9] investigates the use of large language models (LLMs), such as GPT-4 and LLaMA, to detect malware based on behavioral data from sandbox environments. The aim was to overcome the limitations of traditional signature- and heuristic-based methods by applying deep learning to dynamic malware activity logs and using LLMs to classify threats more accurately, particularly for zero-day attacks. The study in [10] presents a hybrid method that combines multiple feature selection techniques with ensemble learning models to detect obfuscated malware, with AdaBoost achieving the best performance.

Whereas [11] has shifted attention to NLP techniques for uncovering deeper semantic and behavioral patterns in malware. The study analyzes how NLP techniques such as name entity recognition, topic modeling, and semantic analysis can contribute to malware detection by uncovering patterns in code and behavior.

Recent studies have explored security mechanisms in distributed, resource-constrained environments, such as edge computing. For instance, phase-based system call filtering has been proposed to enhance

container security in lightweight virtualization layers often used at the edge [12]. Incentive mechanisms for federated learning in vehicular networks address collaborative training under privacy and resource constraints [13], while trend forecasting-based resilience recovery methods improve post-incident restoration in complex traffic networks [14]. Research has exposed weaknesses in content security policies when relying on object storage, illustrating that over-reliance on cloud services can introduce exploitable flaws [15]. This underscores the value of localized, intelligent analysis as implemented in our proposed NGFW architecture.

Complementary techniques in malware analysis and neural network security enrich the threat detection landscape. Intelligent analysis of blockchain applications through Java Native Interface (JNI)-layer deception [16] and scalable, distributed analysis systems oriented toward machine learning [17] demonstrate advanced static/dynamic analysis. Additionally, graph neural networks have been applied to recognize Border Gateway Protocol (BGP) communities [18], and gradient shielding techniques have revealed vulnerabilities in deep neural networks [19]. Although targeting different vectors, these works collectively highlight the role of diverse AI methods in cybersecurity, supporting our integration of NLP with ensemble learning for robust, resource-aware malware detection in edge settings.

Rekha Gangula et al. [20] proposed an intrusion detection approach using Firefly Optimization Algorithm with an ensemble classification model. The optimization algorithm selected optimal feature subsets. The ensemble classifier enhanced detection performance in complex network environments. Rekha Gangula et al. [21] proposed an intelligent intrusion prevention framework for network applications. The system integrated detection and prevention mechanisms within a unified architecture. The framework reduced attack propagation and improved response efficiency. Rekha Gangula et al. [22] proposed a deep learning and optimization-based intrusion detection method for IoT systems. The model combined deep neural architectures with optimization techniques. The approach improved detection accuracy and minimized computational overhead.

3. Proposed Methodology

The proposed methodology presents a structured and data-driven approach for malware classification by integrating data analysis techniques with multiple machine learning models. The research follows a systematic pipeline that begins with dataset ingestion, preprocessing, and feature transformation, followed by exploratory data analysis and model training. The framework incorporates classical machine learning models such as SVM, GNB, and MNB, along with a HGTS that combines DT and GNB to enhance prediction accuracy and robustness. A lightweight database is used for storing user credentials and system interactions, while model persistence techniques ensure efficient reuse of trained models, as shown in Figure 1. The system also includes a graphical user interface that enables users to interact with the system, upload datasets, and perform real-time predictions. The overall design emphasizes automation, scalability, and adaptability, allowing the system to handle complex malware datasets and improve performance through efficient data processing and model evaluation.

User Interface (Tkinter GUI)

- The user interacts with the system through a graphical interface built using Tkinter, ensuring accessibility for both technical and non-technical users.
- It provides a comprehensive suite of options, including admin/user login, dataset upload, preprocessing, model training, and real-time prediction.
- All user actions are triggered through intuitive buttons, with system status and outputs displayed in a dedicated text area for transparency.

- The interface is designed to provide a smooth, interactive experience throughout the entire malware analysis workflow.

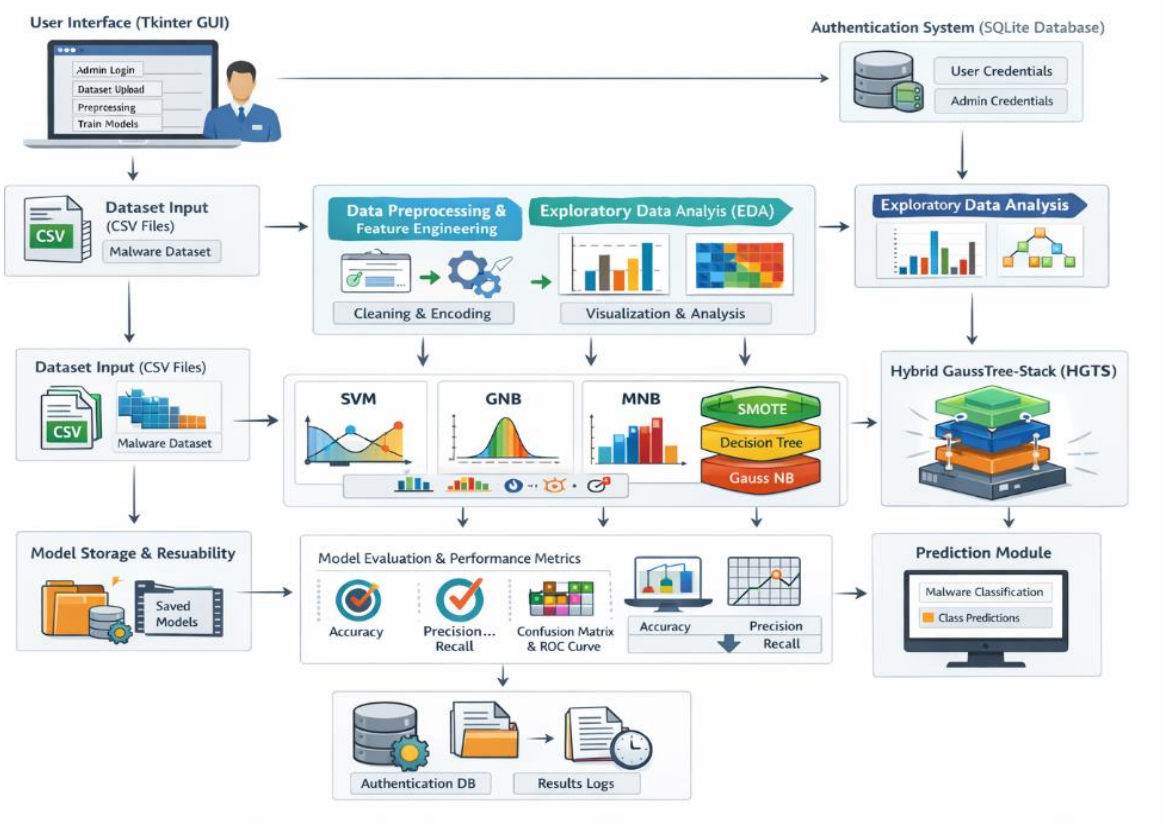


Fig. 1: System architecture.

Authentication System (SQLite Database)

- The system utilizes an SQLite database to manage admin and user credentials securely, preventing unauthorized access.
- It maintains organized tables for usernames and passwords, ensuring a structured approach to identity management.
- The database acts as a gatekeeper, providing controlled access to sensitive training and prediction functionalities based on user roles.
- It interacts directly with the GUI to facilitate seamless registration and login operations.

Dataset Input (CSV Files)

- The system utilizes CSV files as the primary input source, containing multiple features and target columns representing various malware classes.
- Datasets are uploaded by authorized admins or users through a standardized file dialog interface within the GUI.
- This raw data serves as the initial input for the security pipeline, representing the software behaviors or signatures to be analyzed.
- Once uploaded, the raw dataset is automatically forwarded to the preprocessing stage for structural refinement.

Data Preprocessing & Feature Engineering

- The dataset undergoes a thorough cleaning process, which includes removing unwanted columns and handling missing values to ensure data integrity.
- Categorical features are transformed into numerical formats using Label Encoding to ensure compatibility with mathematical models.
- Data normalization and transformation techniques are applied to standardize the feature space for improved learning efficiency.
- The processed data is then split into feature variables (X) and target labels (y) to prepare for the training phase.

Exploratory Data Analysis (EDA)

- The system performs automated visualization to uncover underlying data distributions and complex feature relationships.
- Analytical techniques such as count plots, box plots, and heatmaps are utilized to provide a visual summary of the dataset.
- This module helps identify critical factors such as class imbalance, feature importance, and multi-collinearity.
- The insights derived from EDA are used to refine model selection and optimize the overall performance of the detection system.

Data Splitting Module

- The dataset is divided into training and testing sets using stratified sampling to maintain the integrity of the data.
- This technique ensures that all malware classes are proportionally represented in both the training and evaluation phases.
- The training subset is used to build the classification models, while the testing subset is reserved for unbiased performance evaluation.
- This structured splitting approach significantly improves the reliability and generalizability of the classification results.

Baseline Machine Learning Models (SVM, GNB, MNB)

- The processed data is passed to multiple individual classification models to establish a performance benchmark.
- SVM is utilized to handle complex decision boundaries between malware families.
- GNB and MNB are employed to handle continuous and frequency-based features, respectively.
- Each baseline model independently predicts malware classes, providing a basis for rigorous performance comparison.

Hybrid GaussTree-Stack Model (HGTS)

- This core component implements a hybrid architecture combining DT and GNB through a stacking ensemble.
- SMOTE is applied prior to training to effectively address class imbalance.

- In this architecture, the DT serves as the base learner, while the Gaussian NB acts as the meta-classifier to refine final predictions.
- This hybrid approach is specifically engineered to improve classification accuracy and the system's ability to generalize to new threats.

Model Evaluation & Performance Metrics

- The system evaluates model efficacy using a comprehensive set of metrics, including Accuracy, Precision, Recall, and F1-score.
- Confusion matrices and ROC curves are generated to provide a detailed visual analysis of true positive and false positive rates.
- Results are displayed directly in the interface and saved to the system for future reference and comparative study.
- This evaluation phase is critical for identifying the most effective model for deployment in a production environment.

Prediction Module

- The trained hybrid HGTS model is deployed to predict malware classes on entirely new, unseen data.
- Input datasets are automatically preprocessed using saved encoders to maintain feature consistency during inference.
- The system provides row-wise predictions, displaying specific class labels for each analyzed instance.
- All prediction results are exported in CSV format to facilitate further forensic analysis and reporting.

Model Storage & Reusability

- Trained models are serialized and saved using joblib, allowing for persistent storage and immediate retrieval.
- This capability eliminates the need for time-consuming retraining every time a new prediction is required.
- Label encoders and scalers are also stored to ensure the data transformation pipeline remains consistent over time.
- This module significantly improves system efficiency and reduces the computational overhead of the security framework.

System Workflow Integration

- All individual components are seamlessly interconnected to form a unified, end-to-end malware detection pipeline.
- The GUI facilitates smooth communication between the preprocessing modules, machine learning models, and the SQLite database.
- The system supports a dual-role workflow, catering to administrative training tasks and user-level prediction requests.

- This integrated approach ensures operational stability, real-time usability, and a cohesive user experience.

3.1 Hybrid Gauss Tree-Stack (HGTS) Model

The HGTS model operates by combining the strengths of DT and GNB through a stacking mechanism to improve malware classification performance. It first enhances the training dataset using SMOTE to address class imbalance and ensure fair learning across all categories. The DT model acts as the base learner, capturing complex feature relationships, while GNB serves as the meta-classifier to refine predictions based on probabilistic reasoning. By integrating both models, the system achieves better accuracy, robustness, and generalization compared to individual models, as shown in Fig 2. This hybrid approach leverages both rule-based learning and statistical inference for effective classification.

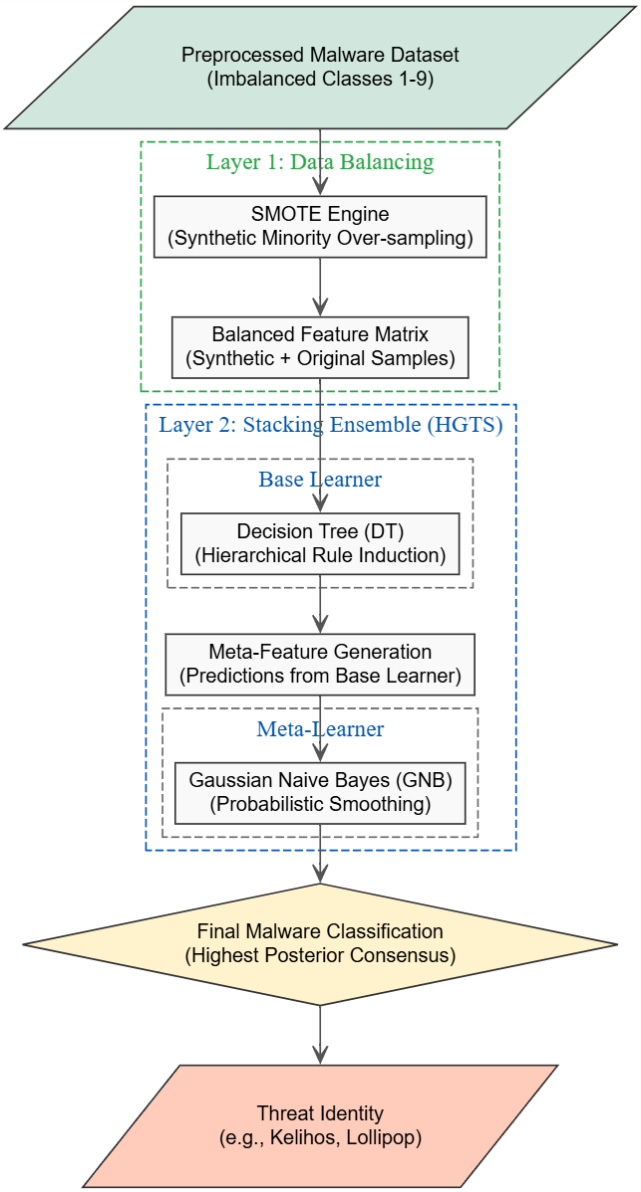


Fig. 2: Internal workflow of HGTS.

Input Feature Preparation: The preprocessed dataset is provided as input to the hybrid model. All features are cleaned, encoded, and normalized to ensure consistency. This step prepares the data for both base and meta-level learning.

Handling Class Imbalance using SMOTE: SMOTE is applied to the training data to generate synthetic samples for minority classes. This balances the dataset and prevents bias toward dominant classes. It improves the model's ability to learn all categories effectively.

Training the Base Model: The DT model is trained on the balanced dataset to learn feature patterns. It creates a tree structure based on decision rules derived from the data. This model captures non-linear relationships between features.

Generating Base Predictions: The trained DT produces predictions for the training data. These predictions are used as input for the next level of the stacking model. This step transforms original features into learned representations.

Training the Meta Model: The GNB model is trained using the outputs from the base model. It learns to interpret these predictions probabilistically. This helps refine and improve the final classification results.

Model Integration through Stacking: The base and meta models are combined using a stacking framework. The DT provides initial predictions, while GNB enhances them. This layered approach improves overall model performance and reliability.

Final Prediction on Test Data: For unseen data, the input passes through the base model and then the meta model. The final prediction is generated based on combined learning. This ensures accurate and robust malware classification.

4. Results and Discussion

Fig. 3 illustrates the exploratory data analysis results of the malware dataset, highlighting key statistical and distributional characteristics across different malware classes. The visualization presents class distribution, ASM file size variations, feature correlation heatmap, and average instruction counts, providing a comprehensive understanding of the dataset structure. It reveals the presence of class imbalance and significant differences in file sizes and instruction patterns among malware families. The correlation heatmap shows relationships between various assembly instruction features, helping identify important attributes influencing classification. These visual insights support effective feature selection and improve the overall performance of machine learning models.

AMERICAN JOURNAL OF AI CYBER COMPUTING MANAGEMENT
Peer Reviewed, Rferred & Indexed Journal
E-ISSN:3069-0102
VOL.6, NO. 2(2026)

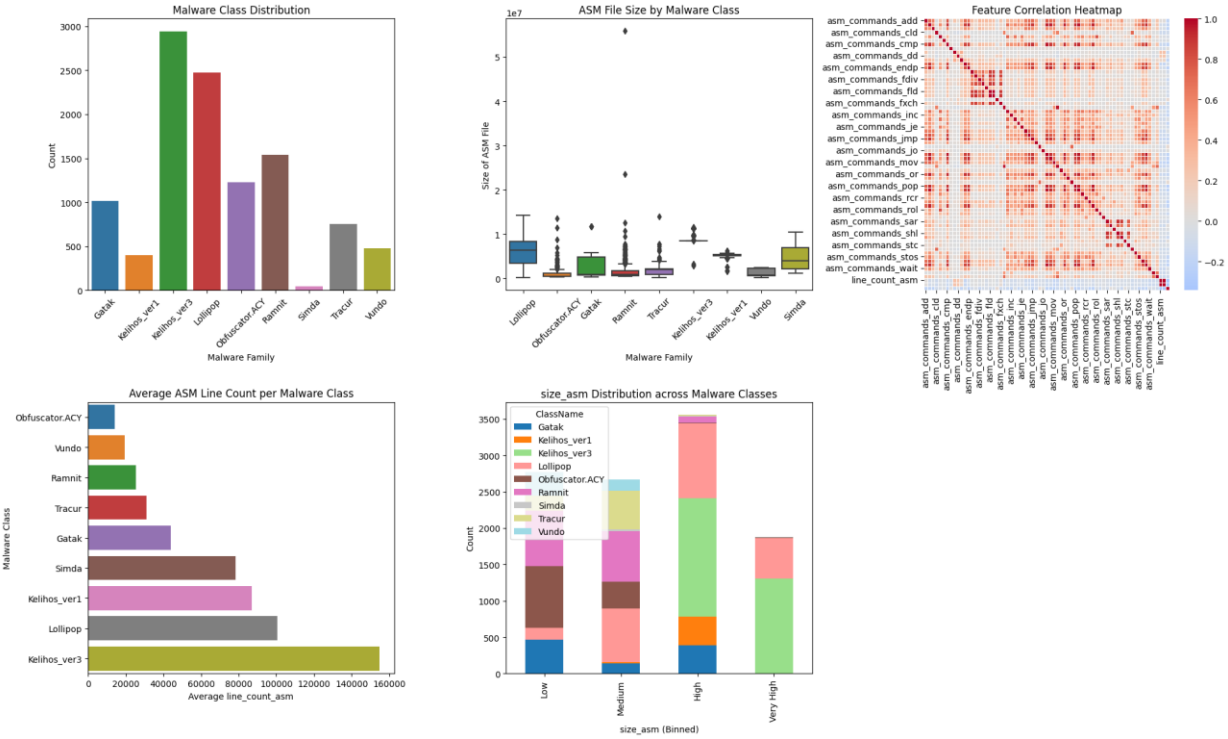
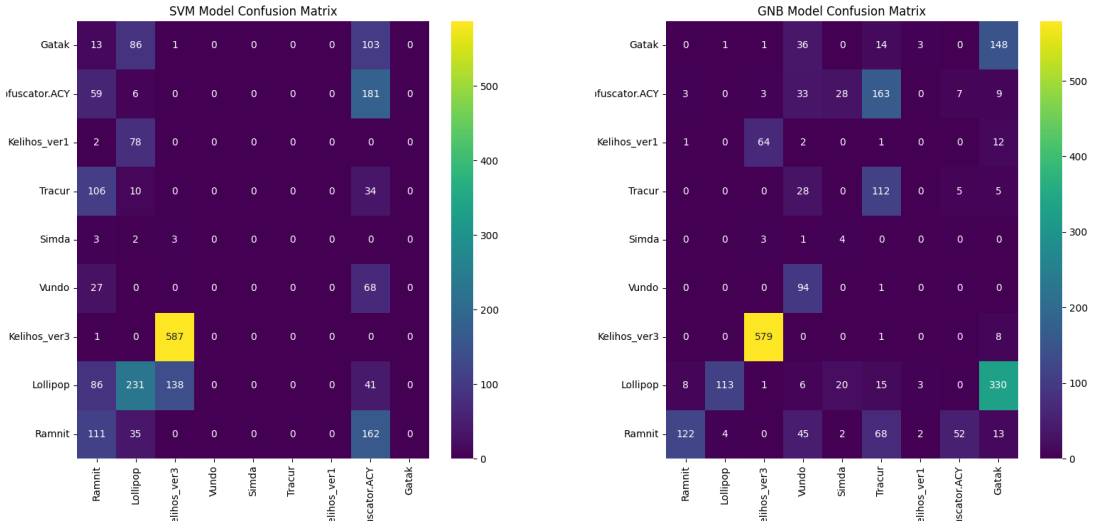
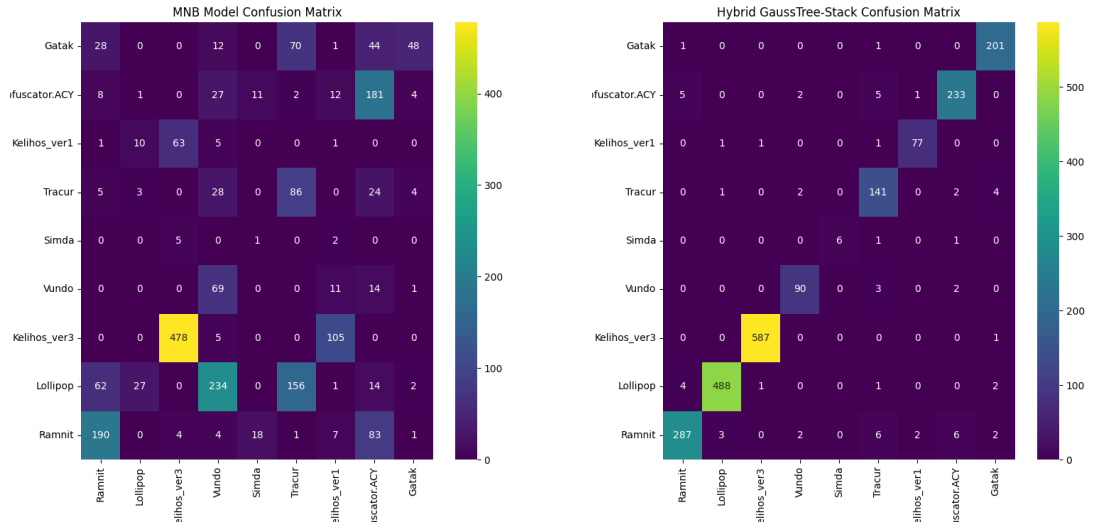


Fig. 3. EDA Plots of Research Work.

Fig. 4 (a) illustrates the confusion matrix obtained using the existing SVM model, highlighting the classification performance across different malware classes. The matrix shows that SVM can correctly classify certain classes such as Kelihos_ver3 with high accuracy, as indicated by strong diagonal values. However, there are noticeable misclassifications among classes like Lollipop and Ramnit, where predictions are distributed across multiple categories. This indicates that the model struggles with overlapping feature patterns between certain malware families. The results suggest that while SVM performs well for some classes, its overall consistency is affected by inter-class similarity.



(a) (b)



(c) (d)

Fig. 4: Confusion matrices obtained using (a) Existing SVM. (b) Existing GNB (c) Existing MNB (d) Proposed HGTS Models.

Fig. 4 (b) illustrates the confusion matrix obtained using the existing GNB model, demonstrating a probabilistic approach to malware classification. The matrix shows improved classification for some classes such as Kelihos_ver3 and Tracur, where higher diagonal values indicate correct predictions. However, the model exhibits misclassification in classes like Ramnit and Lollipop, where predictions are spread across multiple labels. This behavior is due to the independence assumption of features in GNB, which may not hold true for complex datasets.

Fig. 4 (c) illustrates the confusion matrix obtained using the existing MNB model, showing its effectiveness in handling frequency-based features. The matrix indicates relatively better performance for certain classes such as Lollipop and Kelihos_ver3, where correct classifications are prominent. However, significant misclassification is observed in classes like Ramnit and Gatak, indicating limitations in capturing complex feature relationships. The distribution of values across non-diagonal elements suggests that the model is sensitive to variations in feature distributions. These results highlight that MNB performs well for specific patterns but lacks overall generalization capability.

Fig. 4 (d) illustrates the confusion matrix obtained using the proposed HGTS model, demonstrating improved classification performance compared to individual models. The matrix shows strong diagonal dominance, indicating higher accuracy across most malware classes. Classes such as Kelihos_ver3, Lollipop, and Ramnit are classified more accurately with minimal misclassification. The hybrid approach effectively combines decision-based and probabilistic learning, reducing errors observed in other models. These results confirm that HGTS provides better generalization and robustness, making it the most effective model among the evaluated approaches.

The comparative analysis presented in Table 1 demonstrate the performance of different machine learning models used for malware classification, highlighting significant variations in their effectiveness. The SVM model achieved an accuracy of 51.06%, indicating moderate performance but limited capability in handling complex class distributions. The GNB model showed slightly better performance with an accuracy of 54.23%, benefiting from its probabilistic approach, although it still faced challenges with feature dependencies. The MNB model obtained an accuracy of 49.72%, reflecting relatively lower performance due to its limitation in capturing intricate relationships within

the dataset. In contrast, the proposed HGTS model achieved a significantly higher accuracy of 97.06%, demonstrating superior classification capability. The precision, recall, and F1-score values also indicate that HGTS outperforms the other models by a large margin. These results highlight the effectiveness of combining DT and GNB in a hybrid approach, as shown in table 3. The comparison clearly shows that traditional models struggle with complex malware patterns, while the hybrid model provides improved generalization and robustness.

Table. 1: Performance Comparison of existing and proposed algorithms.

Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
SVM	51.06	21.11	28.45	23.60
GNB	54.23	43.39	51.14	37.95
MNB	49.72	44.11	43.26	36.16
HGTS	97.06	96.13	93.90	94.80

5. Conclusion

The malware classification system provides an effective and intelligent solution for identifying different malware families using machine learning techniques. The study successfully integrates multiple models such as SVM, GNB, MNB, and the hybrid HGTS to analyse and compare their performance. Among all models, the HGTS model achieved the highest accuracy of 97.06%, demonstrating superior classification capability compared to traditional approaches. The use of SMOTE significantly improved performance by addressing class imbalance and ensuring fair representation of all classes. Data preprocessing and feature engineering played a crucial role in enhancing the quality of input data and improving model outcomes. Visualization techniques such as confusion matrices and ROC curves provided deeper insights into model behaviour and performance. The results clearly indicate that individual models have limitations in handling complex malware patterns, whereas the hybrid approach improves robustness and generalization. The system also ensures efficient model reuse through joblib, reducing computational overhead during prediction. The GUI-based interface enhances usability, making the system accessible for both technical and non-technical users. The system can be extended by incorporating deep learning models to further improve classification accuracy and handle more complex malware patterns. It can also be enhanced by integrating real-time threat detection and deployment in cloud-based or network security environments.

References

[1] R. T. Geraldi, S. Reinehr, and A. Malucelli, ‘‘Software product line applied to the Internet of Things: A systematic literature review,’’ Inf. Softw. Technol., vol. 124, Aug. 2020, Art. no. 106293.

[2] M. J. Baucas and P. Spachos, ‘‘Using cloud and fog computing for large scale IoT-based urban sound classification,’’ Simul. Model. Pract. Theory, vol. 101, May 2020, Art. no. 102013.

[3] D. Rodriguez, J. Abrahao, D. Begazo, R. Rosa, and G. Bressan, ‘‘Quality metric to assess video streaming service over TCP considering temporal location of pauses,’’ IEEE Trans. Consum. Electron., vol. 58, no. 3, pp. 985–992, Aug. 2012.

[4] M. Al-Qatf, Y. Lasheng, M. Al-Habib, and K. Al-Sabahi, ‘‘Deep learning approach combining sparse autoencoder with SVM for network intrusion detection,’’ IEEE Access, vol. 6, pp. 52843–52856, 2018.

[5] Patel, U. The role of next-generation firewalls in modern network security: A comprehensive analysis. Int. J. Adv. Res. Eng. Technol. 2024, 15, 135–154.

[6] Hossain, M.A.; Haque, M.A.; Ahmad, S.; Abdeljaber, H.A.; Eljialy, A.E.M.; Alanazi, A. AI-enabled approach for enhancing obfuscated malware detection: A hybrid ensemble learning with combined feature selection techniques. Int. J. Syst. Assur. Eng. Manag. 2024, 1, 1–19.

[7] Zhukabayeva, T.; Zholshiyeva, L.; Karabayev, N.; Khan, S.; Alnazzawi, N. Cybersecurity solutions for industrial internet of things–edge computing integration: Challenges, threats, and future directions. Sensors 2025, 25, 213. [PubMed]

[8] Blessing, E.; Ademola, F. Scalability and resource efficiency of next-gen AI-based firewalls: A case study on cloud environments. HAL 2024, hal-04972078.

[9] Akinsowon, T.; Jiang, H. Leveraging Large Language Models for Behavior-Based Malware Detection Using Deep Learning; Report No. IHS/CR-2024-1035; Sam Houston State University, Institute for Homeland Security: Huntsville, TX, USA, 2024.

[10] Silvestri, S.; Islam, S.; Papastergiou, S.; Tzagkarakis, C.; Ciampi, M. A machine learning approach for the NLP-based analysis of cyber threats and vulnerabilities of the healthcare ecosystem. Sensors 2023, 23, 651. [PubMed]

[11] Alanazi, F. A systematic literature review of autonomous and connected vehicles in traffic management. Appl. Sci. 2023, 13, 1789.

[12] Nie, S.; Ren, J.; Wu, R.; Han, P.; Han, Z.; Wan, W. Zero-trust access control mechanism based on blockchain and inner-product encryption on the internet of things in a 6G environment. Sensors 2025, 25, 550.

[13] Chen, K.; Lu, H.; Yao, Y.; Fang, B.; Liu, Y.; Tian, Z. Enhancing Container Security Through Phase-Based System Call Filtering. IEEE Trans. Cloud Comput. 2025, 13, 983–994.

[14] Hong, S.; Yue, T.; You, Y.; Lv, Z.; Tang, X.; Hu, J.; Yin, H. A resilience recovery method for complex traffic network security based on trend forecasting. Int. J. Intell. Syst. 2025, 2025, 3715086.

[15] Reddy, S. K. R. (2024). Designing Blockchain Architecture to Transform Loyalty Rewards into Cryptocurrency Investments.

[16] Lv, Y.; Shi, W.; Zhang, W.; Lu, H.; Tian, Z. Do Not Trust the Clouds Easily: The Insecurity of Content Security Policy Based on Object Storage. IEEE Internet Things J. 2023, 10, 10462–10470.

[17] Purmani, S. S. R. (2025). Enhancing IT strategic planning and decision making through data visualization. International Journal of Enhanced Research in Management & Computer Applications, 14(4), 75–81.

[18] Kalae, U. K. (2020). Developing scalable Power BI dashboards for enhanced data analysis and strategic business decision-making. International Journal of Enhanced Research in Science, Technology & Engineering, 9(3), 8–15.

[19] Gu, Z.; Hu, W.; Zhang, C.; Lu, H.; Yin, L.; Wang, L. Gradient Shielding: Towards Understanding Vulnerability of Deep Neural Networks. IEEE Trans. Netw. Sci. Eng. 2021, 8, 921–932.

[20] Alzu'bi, A.; Darwish, O.; Albashayreh, A.; Tashtoush, Y. Cyberattack event logs classification using deep learning with semantic feature analysis. Comput. Secur. 2025, 150, 104222.

[21] Rekha Gangula,Murali Mohan Vutukuru,M.Ranjeeth Kumar . Intrusion Attack Detection Using Firefly Optimization Algorithm and Ensemble Classification Model. Wireless Pers Commun 132, 1899–1916 (2023). https://doi.org/10.1007/s11277-023-10687-8

[22] Rekha Gangula, Sreenivas Pratapagiri, Sridhara Murthy Bejugama, Sudharshan Ray, Gayatri Nandam, Swapna Saturi," A Novel Intelligent Intrusion Prevention Framework for Network Applications". Wireless Pers Commun 131, 1833–1858(2023).

[23] Rekha Gangula, Murali Mohan Vutukuru, "A Deep Learning and Optimization Method for Detecting Network Intrusion in IOT," 2022 Second International Conference on Advanced Technologies in Intelligent Control, Environment, Computing & Communication Engineering (ICATIECE).