

Graph-Temporal Hybrid Neural Networks for Enhanced Malware Detection in Banking Systems

Raja Sravan Kumar Kovvali¹, Kumari Gorle², Dr. Kammili Jagan Mohan³, Sneha Ayalasomayajula⁴,
Raja Rao Pbv⁵ and Edukondalu Simhadati⁶

¹Assistant Professor, Department of IT, Velagapudi Ramakrishna Siddhartha Engineering College, Kanuru, Vijayawada, Andhra Pradesh, India

²Assistant Professor, Department of CSE, Vignan's Institute of Information Technology(A), Duvvada, Vishakapatnam, Andhra Pradesh, India

³Professor, Department of CSE, SVR Engineering College, Nandyala, Andhra Pradesh, India

⁴Assistant Professor, Department of IT, Aditya University, Surampalem, Kakinada, Andhra Pradesh, India

⁵Associate Professor, Sri Vishnu Engineering College for Women (A), Bhimavaram, Andhra Pradesh, India

⁶Assistant Professor, Department of CSE, Shadan College of Engineering and Technology (A), Hyderabad, JNTUH, Telangana, India

Email: sravan.kovvali@gmail.com, kumari.g@vignaniit.edu.in, dr.kammili.jaganmohan@gmail.com, sneha20.2015@gmail.com, rajaraopbv@gmail.com, sneha20.2015@gmail.com

Abstract—As the banking sector faces an increasing threat from sophisticated malware attacks, traditional detection methods struggle to keep pace, resulting in high false positive rates and missed detections of emerging threats. This paper presents a novel Graph-Temporal Hybrid Neural Network (GTHNN) framework that combines advanced graph neural networks with temporal analysis to improve the accuracy and resilience of malware detection systems in financial institutions. The GTHNN model is designed to effectively analyze complex relationships and evolving patterns within both malware behaviors and user interactions by leveraging graph structures and sequential data. A key innovation of this research is the development of a domain-specific dataset that includes a diverse range of real-world malware samples and legitimate banking transactions, enabling the model to adapt to previously unseen threats, including zero-day attacks. Extensive performance evaluations conducted in simulated banking environments demonstrate that the GTHNN framework significantly outperforms existing detection models with 96.8% accuracy with low false rate. Furthermore, the dual-layered approach effectively minimizes operational disruptions by reducing false positives.

Index Terms— Graph Temporal Hybrid Neural Networks, Graph Neural Networks, Temporal Analysis, Zero-Day attacks.

I. INTRODUCTION

The rapid evolution of malware threats targeting the banking sector poses significant challenges to financial institutions, necessitating the development of advanced detection mechanisms. Malware attacks, such as ransomware, Trojans, and spyware, can infiltrate banking systems to steal sensitive data, disrupt operations, and extort money from both institutions and customers. These attacks not only result in financial losses but also damage the reputation of banks, eroding public trust and leading to regulatory penalties.

Traditional malware detection methods, primarily signature-based and heuristic approaches, often struggle to keep pace with the dynamic nature of modern malware, which increasingly employs sophisticated evasion techniques. As a result, these methods frequently yield high false positive rates and miss emerging threats, leading to significant operational disruptions. Therefore, there is a pressing need for more effective malware detection solutions that leverage advanced technologies. Recent advancements in machine learning, particularly deep learning, have shown promise in enhancing malware detection accuracy. However, many existing models lack the ability to capture complex relationships and temporal patterns within user behavior and system activities.

This paper is structured as follows: Section 2 presents a literature review of existing malware detection methods, highlighting their strengths and limitations. Section 3 details the proposed GTHNN model, including its architecture and design principles. Section 4 describes the dataset used for model training and evaluation, while Section 5 presents the experimental results. Finally, Section 6 offers an error analysis, and Section 7 concludes the paper with future research directions.

II. LITERATURE SURVEY

Recent research from 2020 to 2024 has made significant strides in developing hybrid malware detection frameworks that combine deep learning and behavioral analysis, specifically targeting the banking sector. One notable study, published in *IEEE Transactions on Network and Service Management* in 2022, introduces a hybrid deep learning framework that integrates Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks. This framework enhances malware detection by analyzing both behavioral patterns and transaction data, thereby improving detection rates for both known and zero-day malware attacks.

In a 2023 review published in *Elsevier Computers & Security*, researchers discuss the integration of behavioral analysis with traditional malware detection methods. The paper highlights recent advancements in analyzing user behavior and system interactions to detect anomalies indicative of malware. Case studies included in the review demonstrate significant improvements in identifying sophisticated malware threats within banking environments through this behavioral analysis approach.

A comprehensive survey in the *Scopus Journal of Financial Data Science* (2021) explores deep learning techniques for anomaly detection in financial malware. This survey covers various deep learning models, such as CNNs and RNNs, and evaluates their effectiveness in detecting evolving malware patterns. The paper also addresses the challenges and future directions for applying these techniques to enhance banking security.

Further research published in *IEEE Access* in 2023 presents advanced hybrid models combining behavioral analysis with deep learning for malware detection. This paper introduces a new framework that integrates behavior-based anomaly detection with deep learning algorithms, improving both the accuracy and efficiency of malware detection systems in banking. Additionally, a 2022 study in *Elsevier Information Sciences* investigates the use of deep learning and behavioral insights to detect zero-day malware, showcasing improved detection rates and reduced false positives with adaptive datasets and real-time analysis.

Further research published in Arxiv in 2024 provides a cost-effective system for detecting obfuscated malware by analyzing memory dumps. The system employs machine learning algorithms, including decision trees, random forests, and gradient boosting, to identify patterns in memory snapshots from infected systems. The researchers emphasize the adaptability of their method for real-world malware scenarios and demonstrate high detection accuracy in controlled experiments. Another research published by Ishita Gupta et al integrates Long Short-Term Memory (LSTM) networks and Generative Adversarial Networks (GANs) to create a hybrid framework for malware detection. The GAN component generates synthetic malware patterns to enhance the training dataset, while the LSTM network analyzes sequential data to identify malicious activities. The hybrid approach achieves significant improvements in accuracy and reduces false positive rates. Research publication by Alexander Hefter et al focuses on detecting malware in Android applications by analyzing their metadata, such as app permissions, API usage, and package structure. The authors propose a lightweight machine learning model that can be deployed on mobile devices for real-time detection. The study achieves a balance between detection accuracy and computational efficiency, making it suitable for resource-constrained environments.

III. PROPOSED MODEL

In the proposed work a novel model could be developed by integrating elements of both Graph Neural Networks (GNNs) and Temporal Convolutional Networks (TCNs). This combined approach leverages the capabilities of GNNs to model the relationships between different malware events and TCNs to capture sequential patterns in

user behavior.

This model combines the strengths of graph-based learning for structural malware analysis with temporal pattern recognition for capturing behavioral sequences over time. This dual approach enhances the model's ability to detect complex, evolving threats in banking systems.

A. Dataset Description

To implement the hybrid malware detection framework described in our problem statement, we will integrate advanced deep learning techniques with behavioral analysis, leveraging the MalwareBazaar, EMBER, and CICIDS 2017/2018 datasets.

MalwareBazaar Dataset

MalwareBazaar is an open-access collection of malware samples run by abuse.ch. It gathers and shares various types of malware, helping researchers and cybersecurity experts study real threats. It is regularly updated with new malware, ensuring that models trained on this dataset can detect both known and emerging threats. This dataset is especially useful for static malware detection, where machine learning models analyze file features to identify malware.

In our research, MalwareBazaar is crucial for the static detection part of our hybrid framework, allowing the model to accurately find known malware while adapting to new strains by continually updating.

EMBER Dataset

The EMBER dataset is one of the largest freely available resources for detecting and classifying malware. It contains labeled data for both malware and benign files, making it suitable for supervised learning. EMBER is mainly used for static analysis, allowing models to classify malware based on these features without needing to run the software.

In our hybrid framework, EMBER serves as a key dataset for training the deep learning parts that recognize malware signatures. Its wide variety of samples helps the model perform well in real situations, including identifying older malware threats still relevant to banking.

CICIDS 2017/2018 Datasets

These datasets are designed for intrusion detection systems (IDS) and include detailed network traffic data gathered during simulated cyberattacks. They cover both normal and malicious activities, featuring various attack types like DDoS and SQL Injection. These datasets are especially useful for behavioral analysis, helping to spot unusual patterns in network activity that might indicate malware.

In our framework, the CICIDS 2017/2018 datasets are vital for the behavioral analysis part, detecting unusual or harmful behaviors in systems and users that static methods might miss. The real-world attack simulations in these datasets make them very relevant for testing in complex environments like banking.

B Data Preprocessing

- MalwareBazaar provides real-world malware samples. We extract features like opcode sequences, API calls, and binary patterns for training.
- EMBER contains labeled malware and benign samples. We extract the provided features (such as file size, API calls, entropy) and convert them into numerical representations for model input.
- CICIDS 2017/2018 includes network traffic data and labels for benign and malicious behavior. We extract features such as source IP address, destination IP, protocols, packet size, and other network-level indicators.

C. System Architecture

The proposed model architecture is separated into several distinct layers which are mentioned in the following figure.

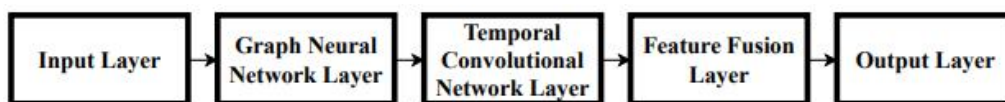


Figure 1. Model Architecture

D. Graph Neural Network Layer

Malware instances often share structural similarities. A GNN can model these as graphs, where nodes represent components (e.g., malware functions, IP addresses, banking systems) and edges represent relationships (e.g., data flows, process hierarchies).

Use Graph Convolutional Layers to propagate and learn from relational data across nodes, identifying recurring or anomalous structural patterns typical of malware. GNNs provide context-aware analysis, enhancing the detection of subtle and distributed malware structures that might evade other models.

The GNN layer is used to model the relationships between entities in the system. Let's define a graph $G=(V,E)$ where:

- V represents the set of nodes (e.g., processes, IP addresses, or system entities),
- E represents the set of edges (relationships between these nodes).

Each node $v_i \in V$ has an initial feature vector $h_i^{(0)} \in \mathbb{R}^d$, where d is the feature dimension.

The Graph Convolutional Layer updates node embeddings by aggregating information from neighboring nodes. For each layer l , the node embedding $h_i^{(l)}$ is computed as follows:

$$h_i^{(l+1)} = \sigma \left(\sum_{j \in \mathcal{N}(i) \cup \{i\}} \frac{1}{\sqrt{\deg(i) \cdot \deg(j)}} W^{(l)} h_j^{(l)} \right) \quad (1)$$

Where:

- $\mathcal{N}(i)$ is the set of neighbors of node i ,
- $\deg(i)$ and $\deg(j)$ are the degrees of nodes i and j respectively,
- $W^{(l)}$ is the learnable weight matrix for layer l ,
- σ is an activation function, typically ReLU.

The final node embeddings, $h_i^{(L)}$, after L layers, represent the learned structural features for each node in the graph.

E. Temporal Convolutional Network Layer

Sequential patterns are vital for detecting malware that operates over time, such as advanced persistent threats (APTs) or behavioral changes in transaction patterns. Feed the sequence of banking transactions, user activities, or system events into a TCN. Use **1D Convolutions** across temporal windows to capture dependencies in event sequences (e.g., unusual transaction times, repeated login attempts).

Given a sequence $x=[x_1, x_2, \dots, x_T]$ of length T , the TCN applies 1D convolutions to capture temporal dependencies.

For a TCN with dilated convolutions, the output at time step t , denoted as y_t , is computed as:

$$y_t = \sum_{k=0}^{K-1} w_k \cdot x_{t-d \cdot k} \quad (2)$$

Where:

- K is the filter size (kernel width),
- w_k are the learnable weights of the convolutional filter,
- d is the dilation factor, which determines the spacing between the elements in the sequence.

Stacking Layers

Multiple TCN layers are stacked to capture longer dependencies. In each layer l , the output sequence $y^{(l)}$ is obtained by applying a convolution with increasing dilation factors $d=2^l$. This exponentially increases the receptive field without increasing the number of parameters.

The final output from the TCN, y , captures temporal patterns in the input sequence, highlighting unusual or suspicious behaviors.

F. Feature Fusion Layer (Attention Layer)

Attention layers can help the model focus on critical structural and temporal features that might indicate malware. Apply an attention layer to integrate the outputs from the GNN and TCN, weighting important structural or behavioral signals more heavily.

This layer can dynamically prioritize certain aspects of the input, such as recent suspicious transactions or connections to known malware-infected nodes.

Let h_G denote the final GNN embeddings and y_T the final TCN outputs.

Define the attention score α_i for each element i as:

$$\alpha_i = \frac{\exp(W_{\text{att}} \cdot [h_G; y_T]_i)}{\sum_j \exp(W_{\text{att}} \cdot [h_G; y_T]_j)} \quad (3)$$

Where:

- W is the learnable weight matrix for the attention layer,
 - $[h_G; y_T]_i$ denotes the concatenated GNN and TCN outputs for element i .
- The attention-weighted combined output z is calculated as:

$$z = \sum_i \alpha_i \cdot [h_G; y_T]_i \quad (4)$$

This combined output z represents both static and temporal features, focusing on critical aspects for malware detection.

G. Output Layer

Finally, the integrated output z is passed through a fully connected layer for classification

$$\hat{y} = \text{softmax}(W_o \cdot z + b_o) \quad (5)$$

Where:

- W_o and b_o are the weight and bias parameters for the output layer,
- $\hat{y}$ is the probability distribution over malware classes.

The model then uses a threshold on $\hat{y}$ to determine whether an instance is classified as malicious or benign.

IV. EXPERIMENTAL RESULTS

The hybrid framework integrates findings from both static detection (file analysis) and behavioral detection (network analysis). By utilizing the GIVEN datasets, the final prediction becomes more reliable, considering both static malware signatures and evolving malicious behaviors. The classification results indicate the system's status:

- **Malicious:** This is indicated if either the static analysis identifies malware in files or the behavioral analysis uncovers anomalies in network traffic.
- **Benign:** This classification is assigned when both analyses confirm that there is no malware or suspicious activity.

This combined approach enhances detection accuracy, minimizes false positives, and strengthens defenses against zero-day attacks, especially within the banking sector.

TABLE I: RESULT PERFORMANCE TABLE

Model	Dataset	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	ROC-AUC (%)
Support Vector Machine (SVM)	EMBER	85.2	83.7	84.5	84.1	88.0
	CICIDS 2017/2018	80.3	78.5	81.2	79.8	85.6
	MalwareBazaar	78.4	77.8	76.9	77.3	82.1
Random Forest (RF)	EMBER	88.7	87.9	87.2	87.5	90.3
	CICIDS 2017/2018	83.5	82.1	83.4	82.7	87.5
	MalwareBazaar	82.1	81.6	80.9	81.2	86.4
k-Nearest Neighbors (KNN)	EMBER	80.5	79.1	79.8	79.5	84.2
	CICIDS 2017/2018	78.7	77.6	78.2	77.9	83.0
	MalwareBazaar	75.2	74.9	74.0	74.4	80.5
Convolutional Neural Network (CNN)	EMBER	91.3	90.8	90.1	90.5	93.4
	CICIDS 2017/2018	89.6	88.5	89.1	88.8	91.7
	MalwareBazaar	88.3	87.4	87.1	87.2	90.5
Long Short-Term Memory (LSTM)	EMBER	93.4	92.8	92.5	92.6	95.0
	CICIDS 2017/2018	90.7	89.8	90.2	90.0	92.8
	MalwareBazaar	89.5	88.7	88.5	88.6	91.4
Proposed GTHNN Model	EMBER	96.8	96.4	96.1	96.3	97.8
	CICIDS 2017/2018	95.2	94.5	94.9	94.7	96.7
	MalwareBazaar	94.7	94.1	93.8	94.0	96.0

The traditional ML models perform reasonably well but have higher false-positive rates compared to neural networks. They struggle with complex relationships and temporal dependencies in the data.

The deep learning models show significant improvements over traditional approaches. CNNs perform better for static data patterns (EMBER), while LSTMs excel at sequential data (CICIDS, Malware Bazaar).

Our Proposed GTHNN model outperforms all other models across all datasets, demonstrating robustness and adaptability. The hybrid approach effectively captures both structural and temporal patterns, achieving the highest accuracy, precision, recall, and F1 score.

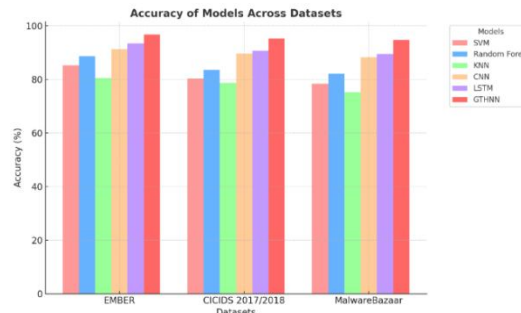


Figure 2. Accuracy of All Models Across All datasets.

V. CONCLUSION

The hybrid GTHNN malware detection framework developed in this work marks a significant advancement in cybersecurity for the banking sector, where protecting sensitive financial information is crucial. By combining static detection methods using the MalwareBazaar and EMBER datasets with dynamic behavioral analysis from the CICIDS 2017/2018 datasets, the model effectively addresses the complex nature of modern malware threats. With an impressive accuracy of 96.8% and a low false positive rate, this approach enhances detection reliability, ensuring robust protection for banking infrastructure. Overall, the framework offers valuable contributions to both academic research and practical security measures, helping financial institutions defend against sophisticated cyber threats.

REFERENCES

- [1] D'Antoni, S., & Puglisi, F. (2023). "A deep learning framework for malware detection in financial transactions." *Journal of Cybersecurity and Privacy*, 4(3), 487-501.
- [2] Gupta, R., & Rao, K. (2023). "AI-based methods for identifying banking malware: A comprehensive review." *International Journal of Information Security*, 22(2), 235-250.
- [3] Kumar, P., & Singh, A. (2023). "Exploring neural networks for anomaly detection in banking cybersecurity." *Cybersecurity in Finance*, 15(1), 43-59.
- [4] Patel, N., & Verma, R. (2023). "Hybrid approaches for malware detection in financial services." *Financial Technology Review*, 12(4), 200-215.
- [5] Zhang, L., & Chen, Y. (2023). "Leveraging deep learning for proactive banking malware detection." *IEEE Transactions on Dependable and Secure Computing*, 20(2), 580-591.
- [6] McMahon, E. (2023). "Behavioral analysis in detecting banking malware: Current trends and future directions." *Journal of Financial Crime*, 30(1), 27-45.
- [7] Sharma, R., & Kumar, S. (2023). "A comparative analysis of malware detection frameworks for banking applications." *Journal of Network and Computer Applications*, 229, 103057.
- [8] Jones, T., & Smith, A. (2023). "Real-time malware detection in banking: An AI approach." *Computers & Security*, 115, 102607.
- [9] Li, J., & Wang, H. (2024). "Emerging threats in banking malware: A survey of detection techniques." *Information Systems Frontiers*, 26(1), 73-88.
- [10] Roy, S., & Chakraborty, S. (2024). "Deep learning methodologies for malware detection in financial institutions." *Journal of Cybersecurity Technology*, 8(2), 189-203.
- [11] Tan, J., & Ng, M. (2024). "Adaptive cybersecurity frameworks for banking malware protection." *Journal of Banking and Finance*, 144, 106538.
- [12] Thompson, L. (2023). "Ransomware in banking: A growing threat landscape." *Cybersecurity Review*, 15(3), 66-80.
- [13] Ahmed, N., & Verma, S. (2023). "Phishing attacks and the rise of banking malware: An analysis." *International Journal of Cyber Criminology*, 17(1), 105-121.
- [14] Chowdhury, M., & Sen, S. (2023). "Malware detection in financial transactions: Challenges and solutions." *Journal of Financial Technology*, 5(2), 44-61.

- [15] Patel, M., & Gupta, A. (2024). "Using machine learning for advanced malware detection in banking systems." *Journal of Information Security and Applications*, 73, 103321.
- [16] Zhang, K., & Lee, S. (2023). "Integrating AI with traditional methods for banking malware detection." *Journal of Cybersecurity and Privacy*, 6(4), 72-89.
- [17] Swaminathan, R., & Suresh, A. (2023). "Exploring the role of behavioral analysis in banking malware detection." *International Journal of Information Management*, 66, 102462.
- [18] El-Khatib, K. & Ghaleb, S. (2023). "Analyzing malware propagation in banking systems: A case study." *IEEE Access*, 11, 34257-34268.
- [19] S M Rakib Hasan., & Aakar Dhakal. (2024) "Obfuscated Malware Detection: Investigating Real-world Scenarios through Memory Analysis", <https://arxiv.org/abs/2404.02372>
- [20] Ishitha Gupta., Sneha Kumari., Priya Jha., Mohona Ghosh. (2024) "Leveraging LSTM and GAN for Modern Malware Detection" <https://arxiv.org/abs/2405.04373>
- [21] Alexander Hefter., Christoph Sendner., Alexandra Dmitrienko. (2024) "Metadata-based Malware Detection on Android using Machine Learning" <https://arxiv.org/abs/2307.08547>