INSTITUTE FOR HOMELAND SECURITY

Sam Houston State University

AI-Driven Malware Detection: Leveraging LLM - Based Behavioral Embeddings for Cybersecurity

Wadduwage Shanika Perera, Tosin Akinsowon and Haodi Jiang

AI-Driven Malware Detection: Leveraging LLM - Based Behavioral Embeddings for Cybersecurity

Wadduwage Shanika Perera, Tosin Akinsowon and Haodi Jiang

Department of Computer Science, Sam Houston State University
Huntsville TX 77341, USA
Email: haodi.jiang@shsu.edu

Abstract

Malware remains a serious threat to critical infrastructure, resulting in data breaches and financial damage in sectors like healthcare, energy, and transportation. As malware techniques become more advanced, traditional detection methods like signature-based and heuristic approaches often fail to keep up, especially when confronting zero-day threats. While machine learning and deep learning have shown potential, they still rely heavily on manual feature engineering for input data, which often limits their ability to generalize to unseen or evolving attacks. Recent advancements in large language models (LLMs) offer promising new direction for improving malware detection by embedding and analyzing complex static and dynamic features. This study introduces a novel AI-driven framework that leverages behavioral embeddings generated by large language models (LLMs) for multi-class malware classification. The model is trained and evaluated on the Avast-CTU Public CAPEv2 dataset. Experimental results show strong performance, highlighting the effectiveness of LLM-based embeddings in capturing behavioral patterns for accurate malware classification.

Index Terms: Malware Static and Dynamic Behavior, Malware classification, Large Language Models, Artificial Intelligence

Introduction and Overview

The continuous evolution of malware makes timely and accurate detection increasingly difficult, pushing the need for more advanced cybersecurity solutions. Signature-based methods are still widely used because they are fast and straightforward, but they depend on recognizing known code patterns and often fall short when dealing with encrypted, obfuscated, or fast-changing threats [1,2]. To address these limitations, dynamic analysis has gained increasing attention in recent research. This approach examines malware behavior during execution, capturing runtime activities such as file operations, registry modifications, and API calls. Compared to static techniques, behavioral analysis tends to be more robust against common evasion strategies and can expose malicious actions that static methods may overlook [3].

The growing availability of behavior logs from sandbox environments has made machine learning (ML) and deep learning (DL) attractive tools for automating malware detection [9]. However, the effectiveness of these models depends heavily on how malware behavior is represented. A wide range of feature representation methods have been proposed, including feature hashing [4], hierarchical multi-instance learning (HMIL) [3], standard learned embeddings with positional encoding [7], trainable embeddings [8,10], opcode-based representations using Word2Vec [5], and task-specific embeddings such as Keras continuous bag of words (CBOW) [6]. While these methods have shown value, they often produce sparse, inflexible, or context-limited representations, which can limit model performance on unfamiliar or evolving malware.

Recent advances in natural language processing suggest a new path forward. Large Language Models (LLMs), originally developed for understanding unstructured text, have demonstrated strong performance across a variety of cybersecurity tasks. Their powerful self-attention mechanisms and ability to capture contextual meaning have enabled applications aligned with the NIST cybersecurity framework, including identifying, protecting, detecting, responding, and recovering from cyber threats [10]. These capabilities position LLMs as promising tools for processing complex behavioral data in malware detection.

This study investigates the use of LLMs to generate rich, contextual embeddings directly from raw malware behavioral reports, eliminating the need for manual feature engineering or predefined hierarchical structuring. A pre-trained Gemini3[15] is used to transform malware behavior logs into deep feature vectors representation, which are then classified using a convolutional neural network (CNN). This design reduces complexity in the analysis pipeline by enabling flexible behavior-based malware classification, a capability especially critical in sectors like healthcare, energy, and transportation.

Gap Assessment and Problem Statement

Despite notable advances in dynamic analysis and deep learning, several critical limitations persist in current malware detection systems. Many existing approaches rely on predefined vocabulary or hierarchical data structures that require extensive preprocessing. These methods are often difficult to scale and adapt to the diversity of real-world malware. Approaches such as HMIL or feature hashing have limited generalization when handling noisy or high-dimensional behavioral logs [3,4,9].

Large Language Models offer a promising alternative by allowing behavioral data to be processed as unstructured text. This makes it possible to extract semantic patterns directly from raw logs without relying on handcrafted features. Despite this potential, recent work has largely concentrated on developing customized transformer architectures, which often require significant computational resources and retraining [7,8]. In contrast, only a few studies have investigated how pre-trained, general-purpose LLMs can be incorporated into modular pipelines for malware classification without task-specific fine-tuning.

Our work addresses this critical gap by introducing a framework that integrates a pre-trained Gemini large language model with a customized convolutional neural network classifier. Raw malware behavioral logs are first transformed into semantic embeddings via the LLM, which are then classified by the CNN without the need for task-specific fine-tuning. This design enhances both scalability and operational efficiency, making the system a potential solution for real-world malware detection and deployment in dynamic cybersecurity environments.

Topic Discussion

This section introduces the dataset used in this study, describing its structure, how we prepared it for malware classification, and the key methods applied in the process.

A. Dataset

The dataset used in this study is the Avast-CTU Public CAPE Dataset, originally introduced by Bosansky et al. [3]. It contains 48,976 behavioral malware reports in JSON format, each categorized into one of ten malware families: Adload (704), Emotet (14,429), HarHar (655), Lokibot (4,191), njRAT (3,372), Qakbot (4,895), Swisyn (12,591), Trickbot (4,202), Ursnif (1,343), and Zeus (2,594), as shown in Figure 1. The reports were primarily collected between 2017 and 2019, allowing for the study of long-term shifts in malware behavior, including concept drift. Each sample is annotated with metadata such as malware family, malware type, detection date, and a SHA-256 hash for unique identification. The dataset includes both static

and dynamic features extracted from sandbox execution, offering a comprehensive foundation for analyzing malware behavior and classification performance.

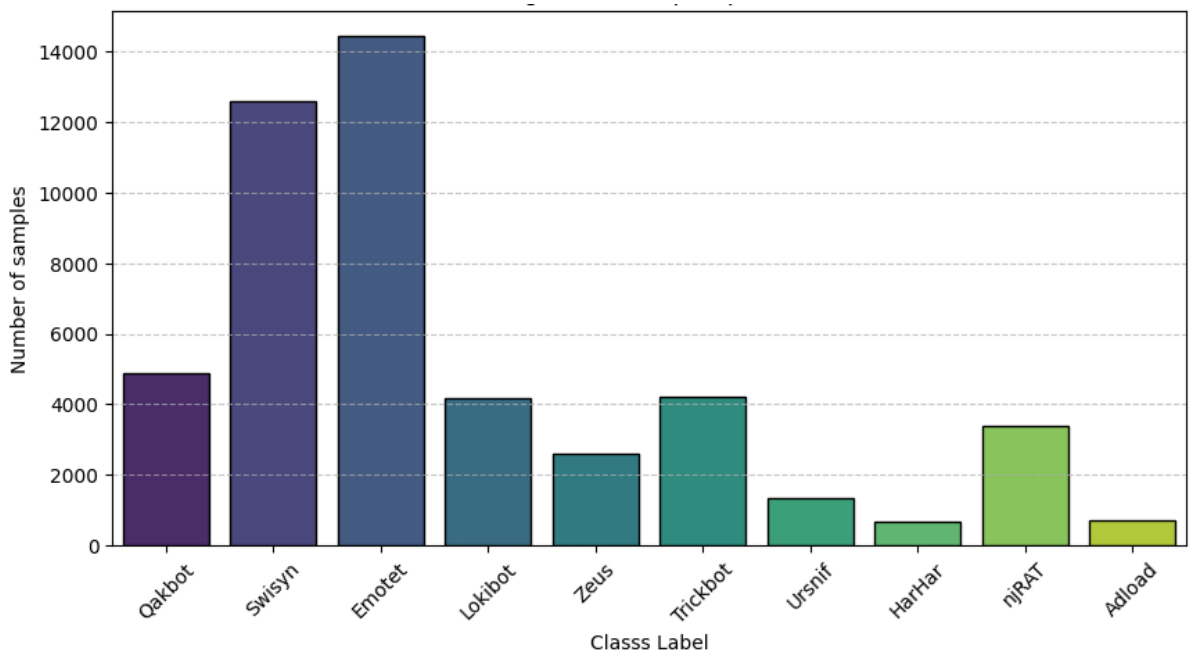

Figure 1: The Distribution of Avast-CTU Public CAPE Dataset

The malware samples used in this study were analyzed using CAPEv2, an open-source dynamic analysis framework that extends the original Cuckoo sandbox [3]. CAPEv2 enhances runtime visibility by unpacking binaries, capturing payloads, and logging detailed behavioral activities. Each sample execution produces a full JSON report that contains information such as process behavior, file system interactions, API calls, and other runtime features. To address challenges associated with excessively large reports and potential label leakage, reduced JSON versions were created. These reduced reports retain the key behavioral summary and static PE file metadata, which are sufficient for machine learning tasks while significantly reducing processing overhead. This balance enables efficient and scalable model training in the present study.

We followed the practice of Bosansky et al. [3] by splitting the dataset according to the sample detection date. All samples dated before August 1, 2019, were used for training (37,512 samples, approximately 76%), while those collected afterward formed the test set (11,464 samples, approximately 24%). This temporal division reflects real-world deployment conditions by incorporating concept drift, enabling a more realistic assessment of model generalization on evolving malware behaviors.

B. Workflow

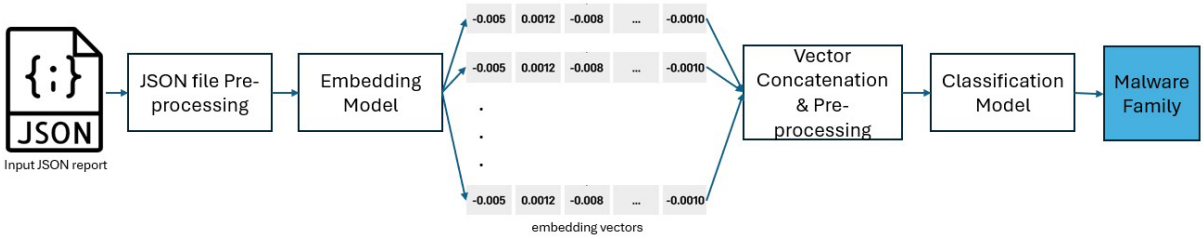

Figure 2: Overall Workflow

Figure 2 illustrates the overall workflow for malware classification based on behavioral reports. The process begins with pre-processing the reduced JSON files produced by the CAPEv2 sandbox, which contain static and dynamic attributes of each malware sample. Due to the token limitations of the embedding model, each JSON file is partitioned into smaller chunks using a structure-aware splitter designed for nested JSON data. These chunks are then encoded using the textembedding-gecko@003 model [15], a pre-trained large language model available on Google Cloud Vertex AI. Each chunk is transformed into a dense, fixed-length semantic embedding vector. The resulting vectors are concatenated, then processed through Principal Component Analysis (PCA) [16] for dimensionality reduction and mean-padding to standardize input lengths. The resulting fixed-length embeddings are passed to a customized CNN model trained to classify the malware into one of ten families based on learned behavioral patterns. This end-to-end framework eliminates manual feature engineering and supports scalable, high-resolution malware classification directly from behavioral logs.

C. Feature Representation

Feature representation is essential to the effectiveness of malware detection models. In this study, we used dense text embeddings derived from raw JSON behavior reports using a Large Language Model (LLM). These embeddings convert unstructured reports into high-dimensional vectors that preserve semantic and contextual information, making them suitable for downstream classification tasks. We employed the textembedding-gecko@003 model from Google Cloud's Vertex AI platform [12], which is designed to generate rich and meaningful representations from text inputs. Each reduced malware report, containing selected static and behavioral features, was serialized into plain text and passed to the embedding model. The resulting vectors offer a compact and informative summary of each report's content, enabling more accurate and scalable malware classification.

D. Data Pre-processing

To accommodate the 2,048-token input limit imposed by the free-tier Google Cloud account, malware JSON reports were divided into smaller segments prior to

embedding. Since most files exceeded the limit, we approximated token count using the Gemini model guideline that one token equals roughly four characters [12]. To preserve the structural integrity of the behavioral reports during segmentation, we employed LangChain's RecursiveJsonSplitter [13], which recursively decomposes oversized nested JSON objects into manageable chunks while retaining hierarchical relationships and converting long lists into dictionaries to optimize boundary alignment. Each chunk was then passed to the textembedding-gecko@003 model, which returned 768-dimensional vectors. These chunk-level embeddings were concatenated to represent each full sample. Due to varied file lengths, the resulting embeddings differed in size, requiring normalization. We applied PCA to reduce longer vectors to the median embedding length and padded shorter ones using their mean values, ensuring consistent input dimensionality for subsequent learning.

E. Model Training

Table 1: Architecture of the Customized CNN

Layer	Parameters and Activations	Feature Map	Output Shape
Input	-	1	1x10000
Conv1D	kernel=5, stride=1, padding=2, activation=relu	32	32x10000
BatchNorm1D	momentum=0.1	32	32x10000
MaxPool1D	kernel=2, stride=2	32	32x5000
Conv1D	kernel=5, stride=1, padding=2, activation=relu	64	64x5000
BatchNorm1D	momentum=0.1	64	64x5000
MaxPool1D	kernel=2, stride=2	64	64x2500
Conv1D	kernel=5, stride=1, padding=2, activation=relu	128	128x2500
BatchNorm1D	momentum=0.1	128	128x2500
MaxPool1D	kernel=2, stride=2	128	128x1250
Flatten	-	-	160000
Dense	in=160000, out=512 activation=relu	-	512
Dropout	rate = 0.5	-	512
Dense (Output)	in=512, out=10, activation=softmax	-	10

A customized convolutional neural network (CNN) model was developed to automatically extract hidden feature representations from the input embeddings. The architecture employs stacked 1D convolutional layers, each followed by batch normalization and ReLU activation, enabling the network to capture complex patterns within the embeddings. Max-pooling layers are inserted after each

convolutional block to downsample feature maps and reduce computational complexity while preserving key information. The model takes a 1×10,000 embedding vector as input and passes it through three convolutional layers with increasing channel depths of 32, 64, and 128, each using a kernel size of 5 with padding of 2. After the final pooling layer, the resulting feature maps are flattened into a vector of size 160,000, which is fed into a dense layer with 512 units and ReLU activation. A dropout layer with a rate of 0.5 is applied for regularization, followed by a final dense layer using softmax activation to perform multi-class classification across ten malware families. This architecture, detailed in Table 1, allows the model to learn discriminative, malware features directly from the input embeddings.

The model was trained using the Adam optimizer with a learning rate of 0.0001 and a batch size of 64. Categorical cross-entropy loss was used to support multi-class classification, and 100 epochs were conducted to ensure stable convergence and effective generalization across diverse malware behavioral patterns.

F. Performance Metrics

The evaluation metrics used to assess the performance are as follows:

1) *Accuracy:* it measures the proportion of correctly classified samples out of the total number of samples in the dataset. It is a widely used metric for classification tasks and provides an overall indication of how well the model performs in classifying different types of malwares.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

2) *Precision:* it measures the proportion of correctly predicted instances of a specific malware class out of all instances that were classified as that class by the model. It is calculated by dividing the number of true positive predictions (correctly classified instances of a particular class) by the sum of true positive and false positive predictions for that class.

$$Precision = \frac{TP}{TP + FP}$$

3) *Recall:* also known as sensitivity, measures the proportion of correctly predicted instances of a specific malware class out of all actual instances of that class in the dataset. It is calculated by dividing the number of true positive predictions for that class by the sum of true positive and false negative predictions for that class.

$$Recall = \frac{TP}{TP + FN}$$

4) *F1 Score*: It is the harmonic mean of precision and recall.

$$F1\ Score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Where:

TP (True Positive):	The number of correctly predicted instances of a specific malware family, indicating the model's ability to accurately classify that specific type of malware.
FP (False Positive):	The instances where the model incorrectly classifies samples as a certain malware family when they belong to a different class or benign.
TN (True Negative):	The number of correctly identified samples that do not belong to the predicted class.
FN (False Negative):	The instances where the model incorrectly classifies samples as benign or other malware families when they belong to the predicted class.

G. Experiment Results:

To evaluate the effectiveness of our proposed approach, we conducted a series of experiments using the prepared dataset and the LLM-generated embeddings. The results demonstrate the capability of the customized convolutional neural network (CNN) model to accurately classify malware behaviors across diverse families.

The model achieved an accuracy of 0.99, recall of 0.99, and an F1-score of 0.99, consistently performing well across all evaluation metrics. These results underscore its ability to reliably identify malware samples spanning ten distinct families. The model effectively generalizes from contextual embeddings derived from behavioral malware reports, capturing discriminative patterns without manual feature engineering. By applying convolutional operations to high-dimensional embedding vectors, the architecture successfully extracts local patterns and learns representations that are both robust and meaningful for classification tasks.

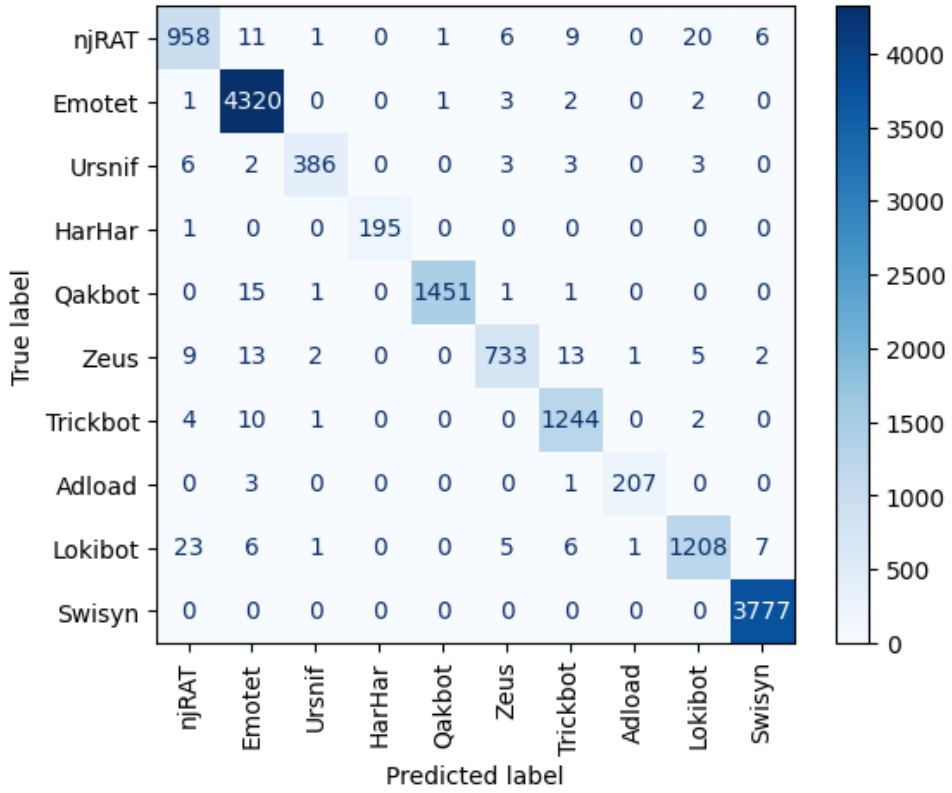

Figure 3: Confusion matrix of the customized CNN model on the test set.

Figure 3 shows the confusion matrix for the customized CNN model, which demonstrates strong diagonal dominance as most predictions align closely with the true labels. High-prevalence malware families such as Emotet and Swisyn are detected with near-perfect accuracy, while low-frequency families like HarHar and Adload are also classified with minimal error. The few misclassifications that appear tend to involve families with overlapping behavioral characteristics, such as Lokibot and njRAT, likely due to shared patterns in API usage or registry activity. These errors are infrequent and do not significantly affect the overall performance, demonstrating the CNN model's reliability and stability in our malware classification tasks.

Table 2: Evaluation Metrics of Selected Models on the Avast-CTU Public CAPE Dataset

Model	Accuracy	Precision	Recall	F1 Score
Bosansky et al. [3]	0.945	-	-	-
Regeciova et al. [21]	0.960	0.999	0.923	0.960
Our Customized CNN	0.990	0.990	0.990	0.990

Table 2 presents a comparative evaluation of our proposed CNN-based framework against selected baseline models using the Avast-CTU Public CAPE dataset in the literature. Bosansky et al. [3] utilized semantic vector representations and a Multilayer Perceptron (MLP), achieving accuracy of 0.945, though no additional metrics were reported. Regeciova et al. [14] relied on YARA rule-based pattern matching, attaining very high precision (0.999) but lower recall (0.923), yielding an F1-score of 0.960. In contrast, our CNN-based framework achieved balanced and superior performance across all evaluation metrics, with an accuracy, precision, recall, and F1-score of 0.99. The findings underscore the advantage of combining contextual embeddings derived from large language models with convolutional architectures, offering a scalable and reliable approach for behavior-based malware classification.

Way Forward

This work presents an AI-driven framework for malware classification built on the Avast-CTU Public CAPE dataset, combining the representational strength of large language models (LLMs) with a customized convolutional neural network (CNN). By converting raw behavioral reports into contextual embeddings, the approach moves beyond manual feature engineering and supports scalable, behavior-aware classification grounded in semantic patterns. The framework achieves strong and balanced results across all evaluation metrics, demonstrating its effectiveness in accurately identifying diverse malware families.

Several technical challenges arose during development. One of the concerns was ensuring that the LLM-generated embeddings meaningfully captured both semantic and structural characteristics of the behavioral logs. Preparing these embeddings for classification required careful pre-processing steps, including chunking, dimensionality reduction, and input standardization, to preserve critical patterns while ensuring model compatibility. At the same time, designing a CNN architecture capable of learning from such representations required careful tradeoffs to maintain sufficient model capacity without compromising efficiency. Together, these challenges highlight the importance of aligning representational choices with model architecture in behavior-based malware classification.

Moving forward, we plan to develop an end-to-end pipeline suitable for real-world deployment. To enhance interpretability, we will incorporate explainable AI (XAI) techniques that highlight which segments of the behavioral sequence or embedding contribute most to classification decisions. This will improve transparency and make the system more actionable for analysts. We also intend to implement continuous learning strategies that enable the model to adapt to evolving malware behaviors over time, reducing reliance on manual retraining. To broaden the model's capabilities, we will integrate additional forms of feature representation, such as

image-based views of malware binaries. Combined with behavioral data, these features are expected to improve generalization, especially for novel or obfuscated threats. Finally, we will continue to optimize the system's performance, with particular attention to inference speed and memory efficiency, ensuring its viability in real-time and resource-constrained environments.

The ability to detect threats through behavior-based embeddings is especially critical in sectors such as healthcare, energy, and transportation, where even minor breaches can have significant consequences. By improving both detection accuracy and operational efficiency, the framework supports the resilience of these essential systems. At the same time, the integration of LLMs, deep learning, and XAI equips cybersecurity professionals with advanced tools that foster workforce readiness. Together, these efforts point toward a practical and scalable path for developing adaptive, transparent, and robust cybersecurity solutions.

Acknowledgement

The authors would like to thank the Homeland Security Institute for funding and support in developing this technique paper.

References

[1] Liu, W.; Ren, P.; Liu, K.; and Duan, H. 2011. Behavior-based malware analysis and detection. In *2011 First International Workshop on Complexity and Data Mining*, 39–42. IEEE.

[2] Ye, Y.; Li, T.; Adjeroh, D.; and Iyengar, S. S. 2017. A survey on malware detection using data mining techniques. *ACM Computing Surveys (CSUR)* 50(3): 41.

[3] Bosansky, B.; Kouba, D.; Manhal, O.; Sick, T.; Lisy, V.; Kroustek, J.; and Somol, P. 2022. Avast-CTU public CAPE dataset. arXiv preprint arXiv:2209.03188.

[4] Zhang, Z.; Qi, P.; and Wang, W. 2020. Dynamic malware analysis with feature engineering and feature learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(01):1210–1217.

[5] Kale, A. S.; Di Troia, F.; and Stamp, M. 2021. Malware classification with word embedding features. arXiv preprint arXiv:2103.02711.

[6] Maniriho, P.; Mahmood, A. N.; and Chowdhury, M. J. M. 2022. MalDetConv: Automated behaviour-based malware detection framework. arXiv preprint arXiv:2209.03547.

[7] Trizna, D.; Demetrio, L.; Biggio, B.; and Roli, F. 2024. Nebula: Self-attention for dynamic malware analysis. *IEEE Transactions on Information Forensics and Security* 19:6155–6167.

[8] Jindal, C.; Salls, C.; Aghakhani, H.; Long, K.; Kruegel, C.; and Vigna, G. 2019. Neurlux: Dynamic malware analysis without feature engineering. arXiv preprint arXiv:1910.11376.

[9] Jiang, H.; Turki, T.; and Wang, J. T. L. 2018. DLGraph: Malware detection using deep learning and graph embedding. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 1029–1033. IEEE.

[10] Trizna, D. 2022. Quo Vadis: Hybrid machine learning meta-model based on contextual and behavioral malware representations. In *Proceedings of the 15th ACM Workshop on Artificial Intelligence and Security*, 127–136. ACM.

[11] Motlagh, F. N.; Hajizadeh, M.; Majd, M.; Najafi, P.; Cheng, F.; and Meinel, C. 2024. Large language models in cybersecurity: State-of-the-art. arXiv preprint arXiv:2402.00891.

[12] Google. 2025. Text embeddings API. https://cloud.google.com/vertex-ai/generative-ai/docs/model-reference/text-embeddings-api. Accessed January 2025.

[13] Google. Understand and count tokens. https://ai.google.dev/gemini api/docs/tokens?lang=python, Jan. 2025.

[14] Regeciova, D., Trnka, M., & Bosansky, M. (2023). GenRex: Generalizing Regex-based Malware Classification via Behavioral Pattern Learning. Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security (AsiaCCS '23), pp. 483–497.

[15] Gemini Team, Anil, R., Borgeaud, S., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., ... Vinyals, O. (2025). Gemini: A Family of Highly Capable Multimodal Models. arXiv preprint arXiv:2312.11805. Retrieved from https://arxiv.org/abs/2312.11805.

[16] Maćkiewicz, A., & Ratajczak, W. (1993). Principal Components Analysis (PCA). Computers & Geosciences, 19(3), 303–342. https://doi.org/10.1016/0098-3004(93)90090-R

Author Biographies

Dr. Haodi Jiang is an Assistant Professor in the Department of Computer Science at Sam Houston State University. He earned his Ph.D. in Computer Science from the New Jersey Institute of Technology (NJIT). His research interests encompass machine learning, artificial intelligence, computer vision, and their applications in solar physics, space weather, cybersecurity, and bioinformatics. His research has been supported by grants from NASA and the Institute for Homeland Security (IHS). His work has been published in high-impact solar physics and space weather journals (e.g., *ApJS*, *ApJ*, *Sol. Phys*, *Front. Astron. Space Sci.*), biomedical imaging journals (e.g., *CMIG*), data mining journals (e.g., *IDA*), and leading machine learning and data mining conferences (e.g., *ICTAI*, *ICMLA*). Dr. Jiang has served as a NASA panelist and as a reviewer for major journals and conferences (e.g., *Nat. Astron.*, *Sci. Data, A&A*, *IEEE Trans. Cybern.*, *TKDD*, *ICDM*, *RAS Tech. Instrum.*, *PeerJ Comput. Sci.*). He also serves on the SABID 2025 program committee at IEEE BigData.

W. Shanika Perera is a PhD student in Digital and Cyber Forensic Science at Sam Houston State University, Texas, with a research focus on the intersection of security and artificial intelligence. Her work explores the application of machine learning in cybersecurity and application security, and the secure deployment of machine learning models. She has published several peer-reviewed papers on the topic and has presented her research at international conferences.

Tosin B. Akinsowon is a PhD student in Digital and Cyber Forensic Science at Sam Houston State University. He holds degrees in Computer Science (B.Sc., M.Sc.) from the University of Lagos and a Master's in Policing from the Criminal Investigation Police University of China. With nearly ten years at INTERPOL NCB Abuja as a Cybercrime Intelligence Officer, he specializes in digital forensics, white-collar crime, and cybercrime intelligence. His research focuses on machine learning, malware analysis, and cybersecurity. Tosin is a member of IEEE, the Interpol Cyber Security Expert Group, and mentors in the Women in Cyber Program.

INSTITUTE FOR HOMELAND SECURITY

Sam Houston State University

The Institute for Homeland Security at Sam Houston State University is focused on building strategic partnerships between public and private organizations through education and applied research ventures in the critical infrastructure sectors of Transportation, Energy, Chemical, Water / Wastewater, Healthcare, and Public Health.

The Institute is a center for strategic thought with the goal of contributing to the security, resilience, and business continuity of these sectors from a Texas Homeland Security perspective. This is accomplished by facilitating collaboration activities, offering education programs, and conducting research to enhance the skills of practitioners specific to natural and human caused Homeland Security events.

Institute for Homeland Security
Sam Houston State University

Perera, W. S., Akinsowon, T., & Jiang, H. (2025). AI-Driven Malware Detection: Leveraging Large Language Models to Embed Static and Dynamic Features for Advanced Cybersecurity (Institute for Homeland Security Report No. 2025-1020). Institute for Homeland Security.

https://doi.org/10.17605/OSF.IO/28BZN