INSTITUTE FOR HOMELAND SECURITY

Sam Houston State University

Behavior-Based Malware Detection

Using Deep Learning with ChatGPT

Institute for Homeland Security

Sam Houston State University

Tosin Akinsowon
Haodi Jiang

Behavior-Based Malware Detection Using Deep Learning with ChatGPT

Tosin Akinsowon and Haodi Jiang

Department of Computer Science, Sam Houston State University

Huntsville TX 77341, USA

Email: hxj024@shsu.edu

Abstract

Abstract – In the ever-evolving landscape of cybersecurity, the threat posed by malware continues to loom large, necessitating innovative and robust approaches for its effective detection and classification. In this paper, we introduce a novel method for malware classification that utilizes malware behavior and the power of deep learning. The utilization of Large Language Models (LLMs) or ChatGPT by attackers makes the recognition of malicious behaviors more challenging, significantly increasing alert fatigue and investigation costs. Our dataset combines malware samples from the BODMAS dataset. The proposed deep learning framework aims to capture the fundamental relevance across various malware families using their dynamic behavior features generated by a sandbox. The model leverages ChatGPT to abstract features in the malware dynamic behavior report, enhancing its capacity for malware detection. The insights derived from this work contribute significantly to the field of cybersecurity and pave the way for further advancements in the realm of malware detection.

Index Terms – Malware Dynamic Behavior, Malware Detection, Large Language Models, Machine Learning

I. Introduction and Overview

The continuous evolution of malware presents formidable challenges to cybersecurity, demanding ingenious solutions for the timely and precise detection and classification of these threats. Malicious software, commonly referred to as malware, represents a grave peril to computer systems as it exploits vulnerabilities to gain unauthorized access and inflict damage. Traditional signature-based detection methods often struggle to keep pace with the continually mutating and obfuscating malware. In response, there is growing interest in exploring innovative deep learning approaches for effective malware analysis and classification [1]. Leveraging the success of deep learning in recognizing, summarizing, translating, predicting, and generating content tasks, deep learning methods have been proposed for malware classification [2] because they offer the capacity to discern complex patterns and glean meaningful representations directly from raw data.

Researchers have extensively explored the use of Large Language Models (LLMs) for text embedding across various fields. For instance, Li and Yu examined the effectiveness of LLMs, such as GPT-2, in generating text embeddings, demonstrating that LLMs can maintain semantic integrity while significantly improving computational efficiency [3]. Similarly, Cao and Gao applied LLMs for malware traffic classification, showing that LLM-generated embeddings effectively distinguish between benign and malicious traffic, leading to faster processing times and enhanced accuracy [4]. In another study, Singh and Sharma investigated the impact of LLM-generated text embeddings, highlighting how LLMs preserve crucial features necessary for high performance in natural language processing tasks [5].

More Recently, researchers have investigated the use of advanced language models and dynamic behavior analysis for malware detection tasks. Li and Yu explore the use of large language models (LLMs) for analyzing malware behavior, demonstrating their effectiveness in identifying and understanding malicious activities [3]. Chen et al. leverage LLMs to process and analyze report.json files from Cuckoo Sandbox, proposing a novel approach for dynamic malware detection [6]. Wang and Li present a deep learning-based approach using dynamic behavior analysis and report.json files from Cuckoo Sandbox, highlighting the efficacy of deep learning techniques, including LLMs, in enhancing malware detection accuracy and robustness [7]. These studies underscore the potential of LLMs and dynamic behavior analysis in improving malware detection tasks.

II. Gap Assessment and Problem Statement

The complexity and sophistication of modern malware have outpaced traditional detection methods, making them inadequate for contemporary threats. Despite advancements, current machine learning (ML) and deep learning (DL) approaches still struggle with the diversity and rapid evolution of malware. Although Large Language Models (LLMs) offer potential for analyzing complex malware behaviors, their computational efficiency and ability to capture nuanced behaviors need further optimization. Additionally, dynamic behavior analysis provides deeper insights into malware activities, but translating these behaviors into actionable features for ML/DL models remains complex. Managing large, high-quality datasets that accurately represent the evolving threat landscape is also a continuous challenge.

Malware's evolving complexity demands advanced detection methods beyond traditional techniques. Despite the promise shown by ML and DL, these techniques often require extensive feature engineering and struggle with generalization. The integration of LLMs for dynamic behavior analysis introduces new opportunities but requires further research to enhance computational efficiency and accuracy. The challenge lies in developing a novel deep learning framework that leverages LLMs and dynamic behavior analysis for malware classification, aiming to improve detection accuracy, robustness, and efficiency in real-world applications. This research seeks to address these gaps, significantly contributing to the field of cybersecurity.

III. Topic Discussion

In this section, we detail the dataset employed in our research, outlining its structure, the preparation process for malware classification, and the methodologies.

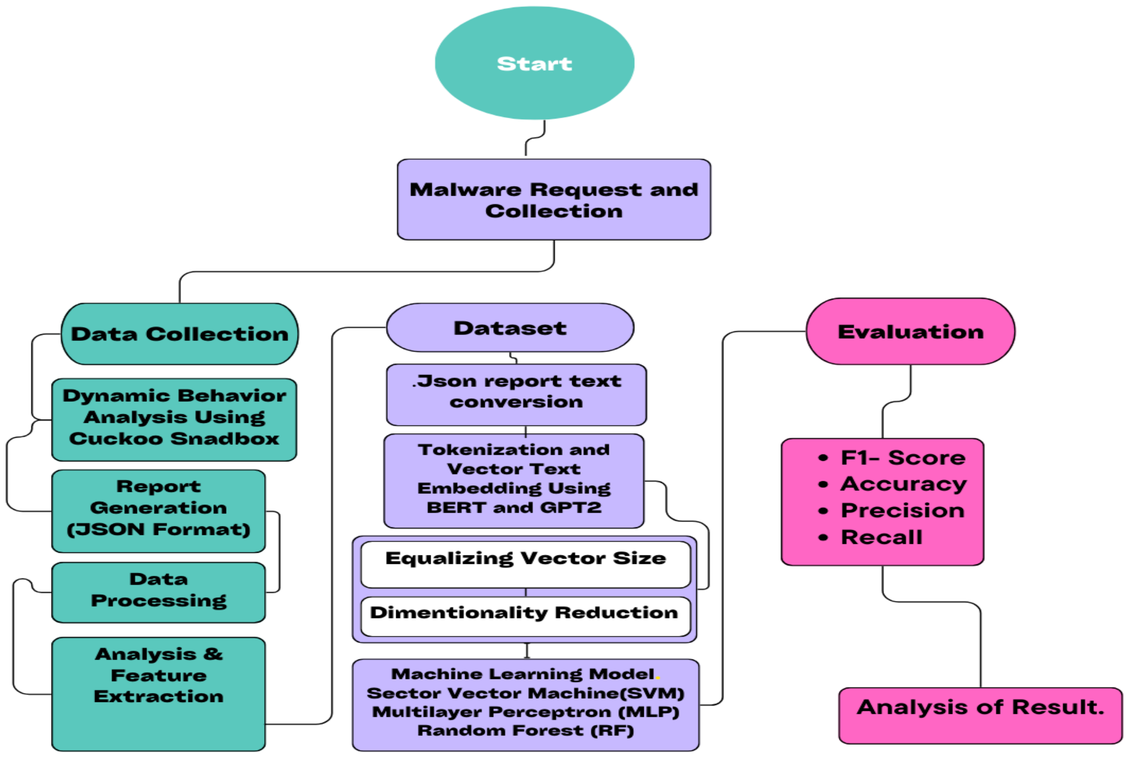

Start
Malware Request and Collection
Data Collection
Dynamic Behavior Analysis Using Cuckoo Snadbox
Report Generation (JSON Format)
Data Processing
Analysis & Feature Extraction
Dataset
.Json report text conversion
Tokenization and Vector Text Embedding Using BERT and GPT2
Equalizing Vector Size
Dimentionality Reduction
Machine Learning Model
Sector Vector Machine(SVM)
Multilayer Perceptron (MLP)
Random Forest (RF)
Evaluation
F1- Score
Accuracy
Precision
Recall
Analysis of Result.

Fig.1 Overall Workflow

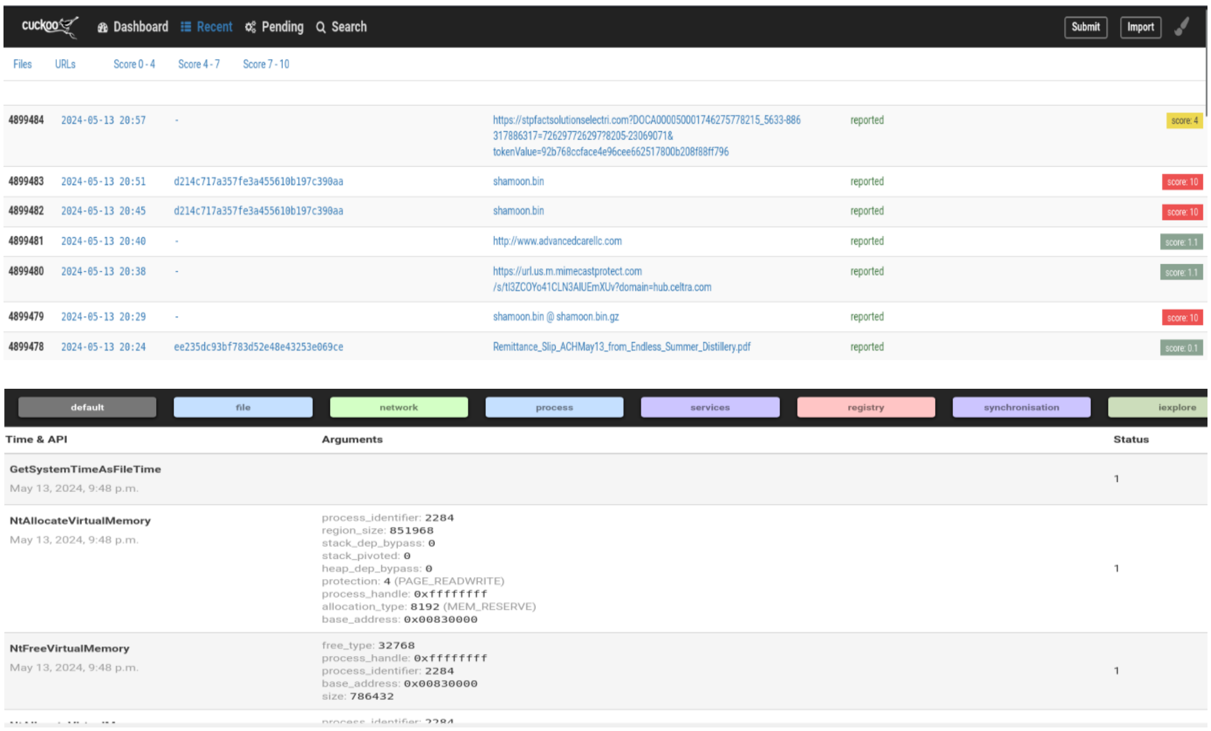
Figure 1 outlines a comprehensive workflow for performing binary classification on malware using dynamic behavior analysis and deep learning techniques with ChatGPT. The process is divided into ten phases. Initially, a 60GB BODMAS malware dataset is collected. In the dynamic behavior analysis phase, malware interactions are observed using Cuckoo Sandbox, generating detailed JSON reports. The data processing phase prepares this data for further analysis. Key features are then extracted from the dynamic behavior reports, providing insights into network activity, file operations, processes, and more. The dataset is prepared with 400 samples of both benign and malicious files, retaining the original JSON reports. In the data transformation phase, JSON files are converted to text and tokenized using GPT-2, transforming text data into numerical vectors. Vector size equalization ensures consistency in length through padding and truncation. Dimensionality reduction via Principal Component Analysis (PCA) standardizes vectors to 10,000 dimensions. Machine learning models, including Support Vector Machine (SVM), Multi-Layer Perceptron (MLP), and Random Forest (RF), are employed for classification. The models' performance is evaluated using metrics such as F1-score, accuracy, precision, and recall. Finally, the results are analyzed to compare the models, identifying the best-performing one for malware classification.

A. Dataset

Our study utilizes data from the source: the BODMAS dataset [8]. The BODMAS dataset is an extensive collection of 57,293 Windows PE files, encompassing a mix of disarmed malware binaries and their associated metadata. From this dataset, we selected a subset of 24,813 executable files, categorized into 14 distinct malware types. These categories include trojans, worms, backdoors, droppers, ransomware, potentially unwanted applications (PUAs), downloaders, viruses, crypto miners, information stealers, exploits, rootkits, peer-to-peer worms, and trojan-game thieves.

B. Malware Analysis with Cuckoo Sandbox

Virtual box with Kali Linux and Cuckoo Sandbox was utilized as the malware analysis environment. Cuckoo Sandbox provides a platform for automated malware analysis, allowing for the dynamic analysis of malicious software behavior. Upon uploading the dataset to Cuckoo Sandbox, the samples were classified based on their maliciousness. The classification criteria were as follows: (1) Samples with a score of 0-4 were considered benign (not malicious). (2) Samples with a score of 4-7 were suspected to be either benign or malicious. (3) Samples with a score of 7-10 were classified as malicious.



4899484	2024-05-13 20:57	-	https://stpfactsolutionselectri.com?DOCA000050001746275778215_5633-886 317886317=726297726297?8205-23069071& tokenValue=92b768ccface4e96cee662517800b208f88ff796	reported	score: 4
4899483	2024-05-13 20:51	d214c717a357fe3a455610b197c390aa	shamoon.bin	reported	score: 10
4899482	2024-05-13 20:45	d214c717a357fe3a455610b197c390aa	shamoon.bin	reported	score: 10
4899481	2024-05-13 20:40	-	http://www.advancedcarellc.com	reported	score: 1.1
4899480	2024-05-13 20:38	-	https://url.us.m.mimecastprotect.com /s/tl3ZCOYo41CLN3AlUEmXUv?domain=hub.celtra.com	reported	score: 1.1
4899479	2024-05-13 20:29	-	shamoon.bin @ shamoon.bin.gz	reported	score: 10
4899478	2024-05-13 20:24	ee235dc93bf783d52e48e43253e069ce	Remittance_Slip_ACHMay13_from_Endless_Summer_Distillery.pdf	reported	score: 0.1

Time & API	Arguments	Status
GetSystemTimeAsFileTime May 13, 2024, 9:48 p.m.		1
NtAllocateVirtualMemory May 13, 2024, 9:48 p.m.	process_identifier: 2284 region_size: 851968 stack_dep_bypass: 0 stack_pivoted: 0 heap_dep_bypass: 0 protection: 4 (PAGE_READWRITE) process_handle: 0xffffffff allocation_type: 8192 (MEM_RESERVE) base_address: 0x00830000	1
NtFreeVirtualMemory May 13, 2024, 9:48 p.m.	free_type: 32768 process_handle: 0xffffffff process_identifier: 2284 base_address: 0x00830000 size: 786432	1

Figure 2: Malware Dataset Analysis using Cuckoo Sandbox

Figure 2 depicts the loading of malware sample data into the Cuckoo environment, where users can select and access each category (benign or malicious). The hash function assigned to each file uniquely distinguishes one malware sample from another, and users can access the scores used for classifying the malware. The score on the rightmost side shows the classification of a sample file, with feedback comments revealing the extent of the malware's maliciousness. The bottom of the figure provides a summary of the dynamic behavior of sample malware data in the Cuckoo environment, showing details such as file, network, process, services, and registry activities. This comprehensive view allows users to analyze and understand significant information about malware's behavior.

C. Prepare ChatGPT API Readable Dynamic Behavior Analysis Report

The focus of this research is using the dynamic behavior of malware as the pattern for classification. To extract the dynamic features of the dataset, the analysis results from Cuckoo Sandbox were exported, and the reports were downloaded. Each report contains valuable information regarding the behavior of the malware samples, including executed files, network activity (ports and IPs), system modifications, and more. To test the feasibility of the proposed idea, a sub-dataset was created by selecting 400 report.json files (comprising both benign and malicious sample data) based on criteria (1) and (3). These files were then converted to a readable format compatible with the ChatGPT API. Python scripts were used to convert the files into both .txt and .pdf file extensions.

D. Malware Behavior Feature Vectorization Using ChatGPT

Text embedding is a technique used in natural language processing to represent text data as numerical values or vectors. This technique transforms text, such as words, sentences, or paragraphs in a document, into values that can map the semantic proximity of such text. Large Language Models (LLMs), such as the GPT model, help extract meaningful features from unstructured data by identifying key behaviors and patterns that characterize different types of malwares. These extracted features are converted into numerical vectors that can serve as input for machine learning models, with each element representing the presence, absence, or frequency of a particular behavior or attribute identified in the reports.

The advantage of text embedding lies in its ability to capture the contextual meaning of words and phrases, which is crucial for accurately understanding and classifying malware behaviors. By leveraging LLMs, we can process vast amounts of textual data from dynamic behavior reports, extracting intricate details that may be missed by traditional feature engineering methods. This approach not only improves the accuracy of malware detection but also enhances the robustness of the models against variations and obfuscations used by sophisticated malware. Moreover, text embeddings facilitate the integration of various data sources, enabling a more comprehensive analysis of malware activities. For instance, network logs, file operations, and system processes can all be embedded and analyzed within a unified framework. This holistic view aids in identifying complex attack patterns and provides deeper insights into the malware's operational tactics.

This method aligns well with ongoing research efforts, highlighting the potential for LLMs to enhance malware detection accuracy and robustness through dynamic behavior analysis. By continuously updating the models with new data and refining the embeddings, we can stay ahead of emerging threats and maintain a high level of security.

Here, we use the text embedding model employed by OpenAI, specifically text-embedding-3-small, which is an improvement over the previous ada002 model due to its efficiency and capability of processing more complex data. This approach is cost-effective and can handle complex data, such as malware behavior reports. The data was extracted using the Fitz Python package and, after text embedding, it was saved in .csv format along with the vectors.

Due to the non-uniformity in vectorization where each file has a different file size, there is a need to achieve uniformity by further applying PCA for dimensionality reduction. The steps are as follows:

Phase 1: Setup

- Import necessary libraries, define folder paths, and ensure the output folder exists.

  – os: Handles file paths and directory operations.

  – numpy: Performs numerical operations and handles array data.

  – PCA from sklearn: Applies Principal Component Analysis for dimensionality reduction.

  – joblib: Saves and loads the PCA model.

Phase 2: Define Functions

- Define functions to load vectors, apply PCA, and save both the reduced vectors and PCA model.

  – Load Embedded Vectors: Uses np.load to load the vector data from a .txt file specified by file Path.

  – Apply PCA: Initializes a PCA model with nComponents=10000 to reduce the vectors to 10,000 dimensions and applies it to the loaded vectors using pca.fit transform.

  – Save PCA Model: Constructs a file path for the PCA model and saves the model using joblib.dump for future use or reproducibility.

  – Save Reduced Vectors: Constructs a file path for the reduced vectors and saves them using np.save.

Phase 3: Process Files

- Iterate over each file in the input folder, applying the reduce Vector Size function to each .txt file.

  – Iterate through each file in the input folder.

  – Check if the file ends with .txt, indicating it's a numpy array file.

  – Construct the full file path using os.path.join.

  – Call reduce Vector Size with the constructed file path and the output folder path to process the file.

E. Malware Classification

We adopted three traditional machine learning algorithms to evaluate the superiority of our extracted malware representation using LLMs.

Support Vector Machine (SVM) works by finding the hyperplane that best separates data points into different classes in a high-dimensional space. In malware classification, SVM differentiates between various categories of malware by maximizing the margin between them. It is effective in high-dimensional spaces and relatively robust to overfitting, particularly when the number of dimensions exceeds the number of samples [10].

Random Forest (RF) constructs a multitude of decision trees during training and outputs the class which is the mode of the classes predicted by individual trees. It is used for malware classification by building trees based on the feature vectors and aggregating their results to determine the most likely malware category. RF handles large datasets with higher dimensionality well and is effective in reducing overfitting by averaging multiple deep decision trees [13].

Multilayer perceptrons (MLP) consist of interconnected layers of neurons designed to process input data through weighted connections and nonlinear activation functions, producing output. During training, MLP adjusts its weights via backpropagation to minimize error and enhance performance. In malware classification, MLPs analyze feature vectors from malware samples, enabling categorization into distinct classes and aiding in threat identification and mitigation [15]. MLPs handle intricate, high-dimensional data well, extracting nuanced patterns indicative of malicious activity. They combat overfitting through multiple neuron layers, ensuring robust generalization to unseen samples. Additionally, MLPs adapt to evolving threats by adjusting weights during training, making them suitable for dynamic environments with new malware variants. Overall, MLPs provide a resilient and adaptable framework for accurate and efficient malware binary classification [11] [12].

Hyperparameter Settings: We used the Python scikit-learn package to implement the base models. For all models, any parameter not specified below is set to its default value. Our RF consists of 500 trees, with 4 randomly chosen features considered when looking for the best split of a node. Our MLP includes 3 hidden layers, each consisting of 100 neurons.

F. Performance Metrics

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

1) *Accuracy:* Accuracy 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:* Precision 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:* 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*: The F1-score 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:

We divided the pre-processed dataset into training and testing sets at ratios of 80% and 20%, respectively. This distribution is designed to train the model comprehensively, fine tune hyperparameters using a distinct subset, and assess the model's ability to generalize on new, unseen data. Additionally, we performed 10-fold cross-validation to validate the experiments.

Table 1: Performance Metrics of Each Model

Model	Accuracy	Precision	Recall	F1 Score
SVM	0.925	0.913	0.954	0.933
MLP	0.932	0.951	0.929	0.940
RF	0.997	0.998	0.997	0.996

Table 1 summarizes the performance of three different machine learning models: SVM, MLP, and RF. The metrics used to evaluate the models include accuracy, precision, recall, and F1-score. The RF model achieves near-perfect scores across all metrics, with an accuracy, precision, recall, and F1-score of nearly 100%, outperforming both SVM and MLP. Specifically, the MLP model follows closely with an accuracy of 93.2%, while the SVM model has an accuracy of 92.5%. Both RF and MLP demonstrate excellent precision, with RF achieving 99.8% and MLP 95.1%, whereas SVM has a lower precision of 91.3%. In terms of recall, RF achieves 99.7%, while MLP and SVM have recalls of 92.9% and 95.4%, respectively. For the F1-score, RF achieves 99.6%, with MLP at 94.0% and SVM at 93.3%. Overall, the random forest model stands out as the superior choice with outstanding performance across all metrics in this study on the constructed dataset.

IV. Way Forward

In this research, we have demonstrated the efficacy of using deep learning and Large Language Models (LLMs) to classify malware based on their dynamic behaviors. Our approach leverages the advanced capabilities of LLMs for feature extraction, which enhances the accuracy and robustness of traditional machine learning models like SVM, RF, and MLP. The results show that the random forest model outperforms the others, achieving near-perfect scores in all performance metrics. This underscores the potential of combining traditional machine learning methods with advanced natural language processing techniques for effective malware detection.

Key challenges identified include the need for high-quality, relevant feature extraction by LLMs, the constant evolution of malware that necessitates continuous model updates, and the importance of efficiently managing large datasets while maintaining high classification speeds. By addressing these challenges, our approach provides a robust framework for improving cybersecurity measures.

Due to time constraints, we were unable to create a sufficiently large dataset for this study. In future work, we plan to significantly expand the dataset and develop an end-to-end deep learning model for malware classification. Expanding the dataset will involve collecting and curating a larger, more comprehensive set of malware samples, ensuring a broader coverage of malware types and behaviors. This will improve the model's ability to generalize and accurately detect a wide range of threats. Additionally, we aim to deploy these models in real-world environments for real-time malware classification. This will involve rigorous testing and fine-tuning in live settings to ensure the models can handle the dynamic and fast-paced nature of actual cyber threats. Integrating these models with existing cybersecurity systems will enable immediate response and mitigation based on the classification results, further enhancing the practical applicability and impact of this research.

To maintain the model's efficacy over time, we will implement mechanisms for continuous learning and adaptation. This includes developing strategies for the models to self-improve based on new data and feedback, ensuring they stay updated with the latest malware trends. Performance optimization will be a key focus, aiming to enhance computational efficiency and accuracy to manage large volumes of data swiftly and precisely. This will involve fine-tuning hyperparameters and employing advanced techniques to reduce false positives and negatives.

By detailing these steps, we aim to create a more resilient and effective malware detection system capable of keeping pace with the fast-evolving threat landscape. The synergy between

LLMs and dynamic behavior analysis offers a promising path forward in the fight against cyber threats, providing a robust framework for enhancing cybersecurity measures.

V. Acknowledgement

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

VI. References

[1] 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.

[2] Nataraj, L.; Karthikeyan, S.; Jacob, G.; and Manjunath, B. S. 2011. Malware images: Visualization and automatic classification. In Proceedings of the 8th International Symposium on Visualization for Cyber Security, VizSec '11. New York, NY, USA: Association for Computing Machinery.

[3] Li, X., and Yu, H. 2021. Exploring the use of principal component analysis for reducing dimensionality in text embeddings. Journal of Machine Learning Research.

[4] Cao, J., and Gao, Y. 2020. Principal component analysis for efficient text embedding with gpt-2. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP).

[5] Singh, R., and Sharma, P. 2022. Dimensionality reduction of text embeddings using pca: A case study with gpt-2. IEEE Transactions on Neural Networks and Learning Systems.

[6] Chen, J.; Zhang, Y.; Li, S.; and Wu, X. 2020. Dynamic malware detection based on large language models. IEEE Access 8:190482–190492.

[7] Wang, T., and Li, M. 2021. Optimizing text embedding representations with principal component analysis and gpt-2. ACM Transactions on Information Systems.

[8] Yang, L.; Ciptadi, A.; Laziuk, I.; Ahmadzadeh, A.; and Wang, G. 2021. Bodmas: An open dataset for learning based temporal analysis of pe malware. In 2021 IEEE Security and Privacy Workshops (SPW), 78–84.

[9] Sathvik et al 2023. Enhancing Machine Learning Algorithms using GPT Embeddings for Binary Classification

[10] Alqahtani, S., and Jones, J. A. 2021. Dynamic malware analysis using machine learning techniques. In Proceedings of the 2021 IEEE International Conference on Big Data (Big Data), 5199–5202. IEEE.

[11] Dahl, G. E.; Stokes, J. W.; Deng, L.; and Yu, D. 2013. Large-scale malware classification using random projections and neural networks. In 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 3422–3426. IEEE.

[12] Hardy, W.; Chen, L.; Hou, S.; Ye, Y.; and Li, X. 2016. Dl4md: A deep learning framework for intelligent malware detection. In Proceedings of the International Conference on Data Mining, 61–78. Springer.

[13] Khan, M. M.; Abbas, G.; and Mehmood, Z. 2022. Random forest-based detection of evolving malware variants. IEEE Access 10:33498–33510.

[14] Pascanu, R.; Stokes, J. W.; Sanossian, H.; Marinescu, M.and Thomas, A. 2015. Malware classification with recurrent networks. In 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 1916–1920. IEEE.

[15] Sharma, R., and Dash, D. 2021. Malware classification using deep learning techniques. In Proceedings of the 2021 International Conference on Advances in Computing, Communication, and Control (ICAC3), 284–290.

[16] Wang, T., and Li, M. 2021. Optimizing text embedding representations with principal component analysis and gpt-2. ACM Transactions on Information Systems.

Authors biography

Dr. Haodi Jiang currently is an Assistant Professor in the Department of Computer Science at SHSU. He earned his Ph.D. in Computer Science from the New Jersey Institute of Technology (NJIT). His research interests include machine learning, artificial intelligence, computer vision, with applications in solar physics, space weather, cybersecurity, and bioinformatics. His work has been published in high impact journals (e.g., ApJS, ApJ, Sol. Phys), biomedical imaging journals (e.g., CMIG), data mining journals (e.g., IDA), and machine learning and data mining conferences (e.g., ICTAI, ICMLA). Additionally, Dr. Jiang has served as a NASA panelist and a reviewer for major journals and conferences (e.g., Nature Astronomy, Astronomy & Astrophysics, IEEE Transactions on Cybernetics, TKDD, ICDM, etc.).

Tosin B. Akinsowon is currently a Doctoral Research Assistant in the Department of Computer Science at Sam Houston State University, pursuing a Ph.D. in Digital and Cyber Forensic Science with a GPA of 4.0. He holds a Master’s in Policing from the Criminal Investigation Police University of China and both a B.Sc. and M.Sc. in Computer Science from the University of Lagos, Nigeria. With nearly a decade of experience as a Cybercrime Intelligence Officer at INTERPOL NCB Abuja, he specializes in intelligence-driven policing, white-collar crime, anti-fraud, and digital forensics. Tosin's research interests include machine learning, malware analysis, and digital forensics, and he actively contributes to the cybersecurity field through his work. He is a member of professional organizations such as IEEE and the Interpol Cyber Security Expert Group and mentors for the Women in Cyber Program.

INSTITUTE FOR HOMELAND SECURITY

Sam Houston State University

Akinsowon, Tosin & Jiang, Haodi (2024) Behavior-Based Malware Detection Using Deep Learning with ChatGPT (Report No. IHS/CR-2024-1008). The Sam Houston State University Institute for Homeland Security.
https://doi.org/10.17605/OSF.IO/98ZTU