IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 2, February 2025

DOI: 10.17148/IJARCCE.2025.14264

Advanced Malware Detection Using Deep Learning in EDR System

Mr. O.T Gopi Krishna[1], D. Dheeraj Sai[2], V. Manohar Naidu[3], L. Deepthi Sai Archana[4], B. Rakesh Babu[5]

Assistant Professor, CSE (IoT, Cybersecurity Including Blockchain Technology),
Vasireddy Venkatadri Institute of Technology, Guntur, India[1]

Student, CSE (IoT, Cybersecurity Including Blockchain Technology),
Vasireddy Venkatadri Institute of Technology, Guntur, India[2]

Student, CSE (IoT, Cybersecurity Including Blockchain Technology),
Vasireddy Venkatadri Institute of Technology, Guntur, India[3]

Student, CSE (IoT, Cybersecurity Including Blockchain Technology),
Vasireddy Venkatadri Institute of Technology, Guntur, India[4]

Student, CSE (IoT, Cybersecurity Including Blockchain Technology),
Vasireddy Venkatadri Institute of Technology, Guntur, India[5]

Abstract: Malware detection plays a crucial role in cybersecurity by identifying and mitigating threats posed by malicious software. Traditional detection methods rely heavily on signature-based approaches, which are often ineffective against new, evolving malware. This paper presents a deep learning-based model for malware detection, leveraging advanced neural network architectures to classify files as either benign or malicious based on their characteristics. By training the model on a comprehensive dataset, it learns to identify subtle patterns that distinguish harmful files from legitimate ones. Enhancing the accessibility and usability of the detection system, the model is integrated into a web-based interface where users can upload files and receive real-time analysis results. Experimental results demonstrate the effectiveness of the deep learning approach in achieving high accuracy and detection speed, showcasing its potential as a proactive tool for modern cybersecurity defence.

Keywords: Malware detection, Deep Learning, Cyber Security, Neural Networks, Malicious files.

I. INTRODUCTION

Malware is still one of the largest issues in the cybersecurity domain because of the increasing digital threats [1]. Since they can execute malicious code on a system directly, executable programs generally they contain an extension.exe—are unique threat in terms of several other forms of malware [2]. Classic detection tools find it very hard to discern malicious and innocuous executables since it is possible for the malicious ones to masquerade themselves as real applications or even system utilities [3]. The malware detection requires advancement; the application of visual features using fusion-based deep learning obtains very accurate and real-time classification [4].
Most of the current strata of antimalware techniques, like signature and heuristic analysis, cannot cope with novel or polymorphic malware, always on some additional cover usually delineated as self-changing elements or self-churns to remain undetected [5]. Signature-based solutions are defined and familiar from earlier patterns of malware, whereas heuristic solutions attempt to identify malware from behavior. Both frameworks suffer from weaknesses in handling unknowns, new variants, or advanced persistent threats [6]. As malware becomes increasingly sophisticated, these measures have proven inadequate in offering full and proactive protection [7]. Deep learning-based methods, particularly neural networks, hold more potential as a substitute for malware detection, especially for .exe file types [8]. In contrast to conventional approaches, deep learning models can perform the concurrent assessment of vast quantities of data from which they can learn a vast number of underlying intricate patterns and associations without predefined rules or signatures [9]. Therefore, these models are based on the assessment of raw features of.exe files, byte strings, APIs calling, and numerous others in classifying files accordingly into benign or malicious classes [10].
This is an extremely handy capability we could employ for either detecting zero-day attacks or new strains of malware not adhering to known past signatures [11].

The results of the research appear with 98% accuracy reflecting the effectiveness of deep learning in taking a central position in proactive defense against cyberattacks. In addition to that, the article investigates the product of ensemble learning and model combination techniques to diminish biases and elevate model complexity [12]. The system seeks to improve malware detection precision through training from massive databases and responding to novel, previously unseen malware [13]. This paper will attempt to explore recent developments in detecting malware on macOS, Windows, iOS, Android, and Linux with the help of deep learning (DL) through exploring DL in text classification and image classification, utilization of pre-trained and multi-task learning models towards malware detection strategies to achieve high precision, and which is the optimum strategy if we have a standard benchmark dataset. [14].

II. LITERATURE SURVEY

Malware detection using deep learning has gained significant traction due to the limitations of traditional detection methods, such as signature-based and heuristic approaches. Deep learning techniques provide more robust and adaptive solutions, especially against zero-day threats and polymorphic malware, by learning complex patterns from large datasets. Several studies have explored novel approaches to improve malware detection accuracy, efficiency, and adaptability.

In 2019, Alzaylaee et al. [15] proposed DL-Droid, a deep learning-based Android malware detection system using real devices. Their approach leverages deep learning to analyze data collected directly from real Android devices, aiming to improve the accuracy and effectiveness of malware detection. By utilizing real-time behavioral data, DL-Droid enhances its adaptability to detect emerging and previously unseen malware threats.

In 2022, Ahuja et al. [16] introduced a dynamic analysis technique utilizing Long Short-Term Memory (LSTM) networks to detect malware based on system call sequences. By monitoring the behavior of executables at runtime, this approach effectively identifies polymorphic and metamorphic malware, which frequently alters its static properties to evade detection. Their results demonstrate that LSTM networks excel in capturing sequential dependencies in system activities, enhancing detection accuracy.

In 2019, Rathore et al. [17] explored the use of machine learning and deep learning models for malware classification. Their work highlighted the effectiveness of random forests, support vector machines (SVMs), and deep neural networks (DNNs) in distinguishing between benign and malicious executables. They found that deep learning models outperformed traditional ML approaches in detecting polymorphic malware.

In 2022, Ali et al. [18] conducted a systematic literature review on deep learning methods for malware and intrusion detection, highlighting recent advancements in CNN-based and transformer-based malware classification models. Their study discussed the strengths and limitations of static, dynamic, and hybrid analysis approaches.

In 2018, Sewak et al. [19] investigated a deep learning-based malware detection system, demonstrating how deep neural networks (DNNs) can be trained on large datasets to identify malicious patterns. Their work explored hyperparameter tuning and optimization strategies to enhance detection performance. Additionally, they highlighted the importance of feature engineering in improving model generalization and reducing false positive rates.

Deldar et al. [20] reviewed deep learning methods to detect and classify zero-day malware. Their paper reviewed different models of the deep learning system and underscored their performance in detecting unseen malware. The challenges they identified were adversarial attacks against the zero-day malware detection approach, feature selection, and data set limitations.

Anderson et al. [21] introduced EMBER, an open dataset for training ML models to detect malicious PE files. Their study established a comprehensive benchmark for malware detection research to develop and evaluate new detection techniques. They demonstrated that embedding machine-learning models trained on EMBER could be used to classify benign and malicious executables, which could contribute to fast automated malware analysis.

Vinaykumar et al. [22] had proposed an intense intelligent malware detection framework based on deep learning. They made use of high-fidelity neural networks for better classification of malicious software for different datasets. They showed that deep learning techniques could readily help detect and classify malware variants, thus improving cybersecurity against evolving threats.

To address data imbalance issues in malware detection datasets, Singh et al. [23] implemented synthetic oversampling techniques combined with Generative Adversarial Networks (GANs) to generate realistic malware samples. This approach enhances model generalization and improves performance on rare malware types that may otherwise be underrepresented in training data.

Li et al. [24] proposed a lightweight malware detection model optimized for edge computing environments. Their method ensures real-time malware detection on resource-constrained devices such as IoT systems and embedded platforms, balancing detection accuracy with computational efficiency.

Gutierrez et al. [25] employed deep learning models for real-time automatic malware detection. Their approach basically revolved around the efficiency of using neural nets for the identification of malicious software. They showed that deep learning-based detection systems can improve real-time identification of threats, in turn improving the fortification of growth in cybersecurity against evolving malware threats.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 2, February 2025

DOI: 10.17148/IJARCCE.2025.14264

Maniriho et al. [26] explored deep learning models for malware attack detection. The study evaluated different combinations of neural architecture to increase malware classification performance. They showed that deep learning methods efficiently identify and counter malware threats and provide strong contributions to cybersecurity.

Mehta et al. [27] explored contrastive learning for self-supervised malware detection. Their approach learns feature representations from unlabeled data, reducing dependency on large, labeled datasets, which are often scarce in cybersecurity research. Their study highlights the potential of contrastive learning in improving malware detection accuracy while minimizing the need for extensive manual labeling.

Finally, Xing et al. [28] propose a new method of malware detection based on autoencoders in deep learning. Their work seeks to utilize unsupervised learning to detect malicious patterns in software behavior. They showed that the autoencoder-based model is capable of separating malware from normal files through compact representation learning, hence improving malware detection efficiency.

III. PROPOSED MODEL

We propose a deep learning-based malware detection model designed to identify malicious executable (.exe) files by recognizing unique patterns in their binary structure. The model's architecture consists of key components, including data preprocessing, feature extraction, and neural network training. Initially, raw .exe files undergo preprocessing to convert their binary format into a numerical representation suitable for deep learning. This transformation involves converting byte sequences into structured data formats such as byte histograms or byte-level vectors. To ensure consistency and improve model convergence, the obtained byte sequences are normalized using methods like min-max scaling. The dataset is then divided into training, validation, and test subsets to enhance generalization and allow for effective performance evaluation and hyperparameter tuning.

Feature extraction plays a crucial role in transforming raw byte sequences into meaningful representations for deep learning models. The primary features include byte sequences that are converted into fixed-length vectors, along with additional metadata such as file size, creation time, file type, and structure, which provide useful contextual information. Statistical characteristics like entropy, mean, standard deviation, and skewness are also computed to capture the randomness and structure of the file, aiding in the identification of malicious patterns. Furthermore, optional execution behavior analysis can be incorporated, where dynamic analysis tracks API calls and system interactions to detect suspicious behaviors indicative of malware.

The proposed model architecture is based on Deep Neural Networks (DNN), which are highly effective in identifying patterns from sequential data such as byte streams from executable files. By leveraging deep learning, the model can automatically learn complex relationships between extracted features and malware characteristics, improving detection accuracy. Through extensive training and validation, the model is optimized to distinguish between benign and malicious files, providing a robust approach to malware detection in executable files.

Data Preprocessing: The raw .exe files are first subjected to preprocessing to convert them into a format suitable for deep learning models. The preprocessing steps are as follows:

File Conversion: The raw .exe files, which are binary, are transformed into numerical representations. This transformation may involve converting byte sequences into structured data formats such as byte histograms or byte-level vectors. These representations are essential for feeding the model with meaningful input data.

Normalization: The byte sequences extracted from .exe files are normalized to ensure uniformity and to help with model convergence during training. Common normalization techniques like min-max scaling are used to adjust the data into a standard range.

Data Splitting: To ensure the model is robust and generalizable, the dataset is split into training, validation, and testing subsets. This allows us to evaluate the model’s performance on unseen data and fine-tune its hyperparameters.

Feature Extraction: Feature extraction is a crucial step in the malware detection process, as it converts raw byte sequences into structured information that deep learning models can learn from. The features extracted from .exe files include:

Byte Sequences: The primary feature extracted from the .exe files is the sequence of bytes. These sequences are converted into fixed-length vectors that represent the entire file and serve as input to the deep learning model.

File Metadata: Additional features are extracted from the file’s metadata, such as its size, creation date, file type, and structure. These features provide context that may help distinguish benign files from malware.

Statistical Features: Various statistical measures, including entropy, mean, standard deviation, and skewness, are calculated from the byte sequences. These metrics help capture the inherent structure and randomness of the file, which is critical for identifying malicious files.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.102 ⁓ Peer-reviewed & Refereed journal ⁓ Vol. 14, Issue 2, February 2025
DOI: 10.17148/IJARCCE.2025.14264

Execution Behavior (Optional): For more advanced detection, dynamic analysis may be employed, where the behavior of an .exe file during execution is observed. This analysis includes monitoring API calls and system interactions, which may help detect behavior typical of malicious files

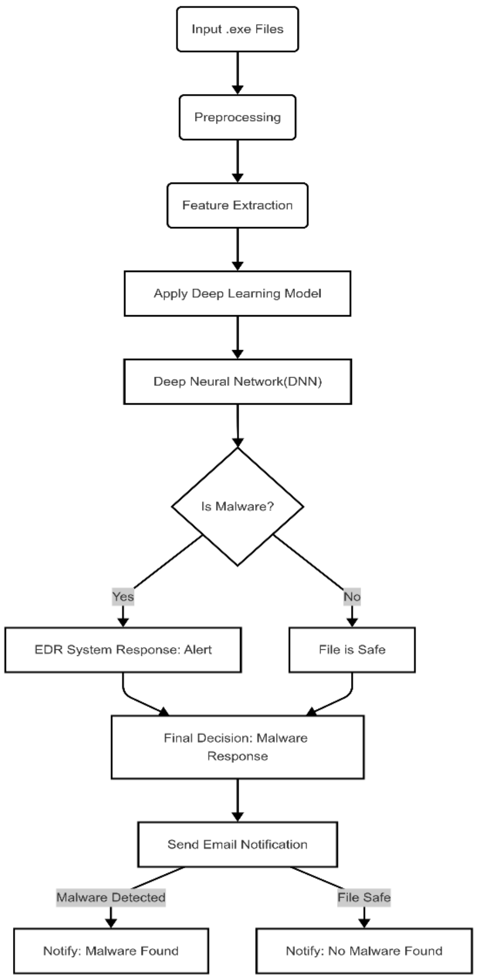


Fig.1: Malware Model Architecture

3.1 DEEP NEURAL NETWORK(DNN)

This model is a Deep Neural Network (DNN), which is usually suitable for detecting patterns in sequential data like byte sequences obtained from executable file images themselves. The architecture of the DNN in this model includes:

3.1.1 Input Layer

The input layer takes in the preprocessed representative byte sequences, which are expressively described in terms of numerical reality. These byte sequences are the prime data used to classify .exe files as either benign or malicious.

3.1.2 Hidden Layers

The DNN model has very many hidden layers acting to learn further compressed representations for the byte sequences. Each hidden layer is comprised of several neurons that perform information processing from the previous layer.
The activation functions used in these layers are ReLU (Rectified Linear Unit) functions, which augment non-linearity and help the model ascertain complex correlations between features.

3.1.3 Fully Connected Layers

Following these inner hidden layers, data is further directed through one or more fully connected layers. In these layers, each neuron in a particular layer is connected to all neurons belonging to the preceding layer. This allows the learned features from prior layers to be integrated and yields more abstract forms of output for the various data.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 2, February 2025

DOI: 10.17148/IJARCCE.2025.14264

3.1.4 Output Layer
The output layer is responsible for producing the corresponding probability distribution across 2 classes (Malware vs. Benign) using the SoftMax activation function: the class that has a higher probability would be the final output of the model.

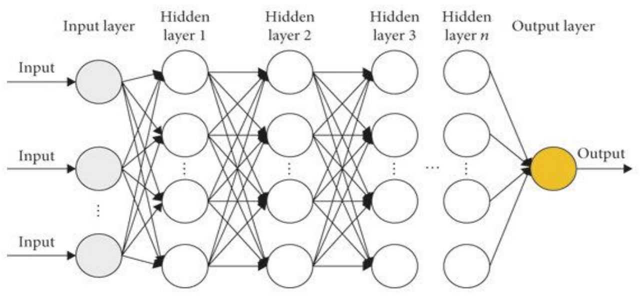


Fig.2: Structure of DNN Model

IV. TRAINING MODEL

TRAINING THE DNN MODEL INVOLVES SEVERAL KEY STEPS:

4.1 Loss Function
In a binary classification task, such as malware detection, binaries cross-entropy is used as a loss function. This loss function measures the deviation/difference between predicted probabilities and true class labels that are then optimized by the model with the intention of minimizing the loss.

4.2 Optimizer
Adam optimizer is employed to minimize the loss function during training. Adam is an adaptive learning rate optimizer that adjusts the learning rate according to the gradients to increase convergence speed.

4.3 Data Augmentation
To combat augmentation techniques were applied to handle class imbalance (where better file instances exist compared to malware instances) and to improve the robustness of the model. Data augmentation techniques entail minor adjustments in the byte sequences such as injecting noise and making slight modifications in the data sequences. This did help generalization by enhancing model capacity overfitting.

4.4 Regularization
To avoid overfitting, the model has dropout layers added to it. The dropout layer randomly disables a certain percentage of the neurons during the training process, thereby preventing the model from becoming so reliant on any one feature or input.

4.5 Early Stopping
Early stopping is the process of stopping training as soon as the performance of the model on the validation set shows no improvement. In this way, overfitting is avoided through ensuring training is not prolonged with too many epochs leading to generalization degradation.

V. MODEL EVALUATION

The evaluation of the model demonstrated exceptional performance with an accuracy of 99%, indicating the high reliability of the model in distinguishing between malware and benign files. Precision was calculated at 98%, highlighting the model's ability to accurately identify malware without generating many false positives.
The recall rate was 97%, ensuring that most of the actual malware files were detected. The F1-score, which balances precision and recall, was 97.5%, reflecting the model's overall efficiency in handling both classes effectively. These results emphasize the robustness of the model in malware detection tasks.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁂ Peer-reviewed & Refereed journal ⁂ Vol. 14, Issue 2, February 2025

DOI: 10.17148/IJARCCE.2025.14264

VI. RESULT & DISCUSSIONS

In this section, we present various selected evaluation results from our Deep Neural Network (DNN) model, which is supposed to act as a malware detector. The model was tested on a separate dataset, and several performance evaluation metrics were applied on it, to check its prowess in classifying the benign and malicious .exe files.

Performance Metrics
The following formulas were used to calculate the evaluation metrics:

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

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

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

$$\mathbf{F1 - Score} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

*TP = True Positive
*TN =True Negative
*FP = False Positive
*FN = False Negative

Interpretation of Results
The model achieved an impressive 99% accuracy, meaning it correctly classified 99% of the files, either benign or malicious. With a precision of 98%, the model accurately identified most files predicted as malware as actually being malware, minimizing false positives. The recall of 97% indicates that the model correctly detected 97% of actual malware files, reducing false negatives and ensuring critical threats were flagged. The F1-Score of 97.5% reflects a balanced performance between precision and recall, crucial for detecting malware without overfitting to one class. The model's AUC of 0.99 confirms its high ability distinguish between malware and benign files across between malware and benign files across different thresholds, further validating its robustness.

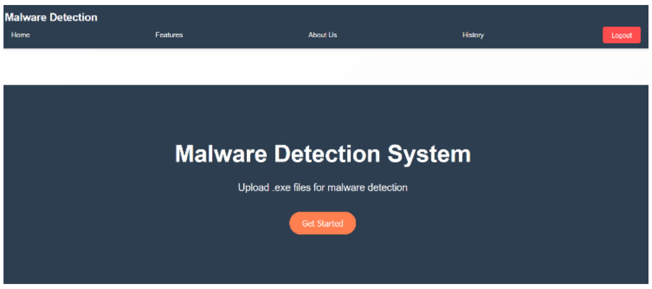


Fig.3: Home page

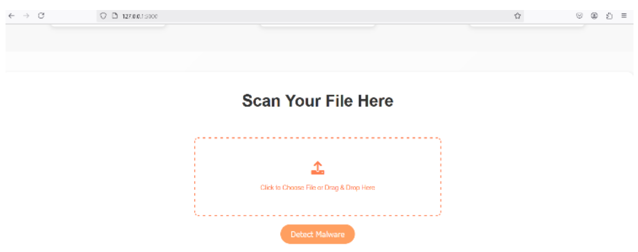
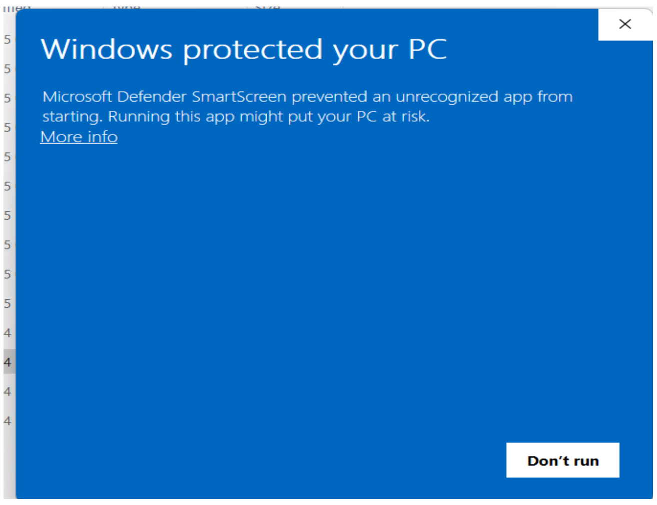


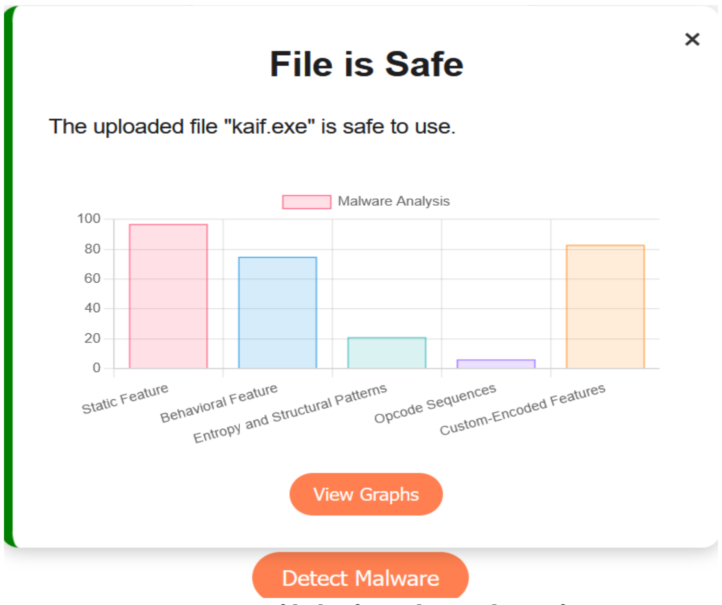
Fig.4: Upload Input File

DOI: 10.17148/IJARCCE.2025.14264


Windows protected your PC

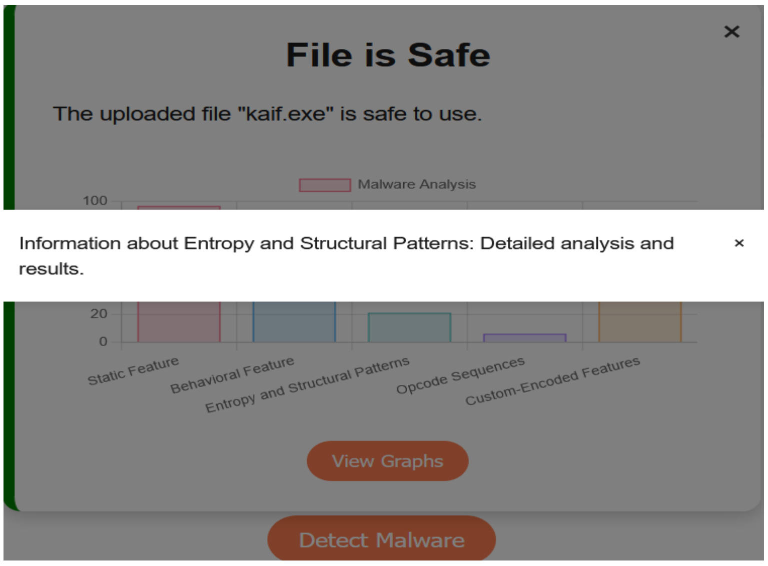
Microsoft Defender SmartScreen prevented an unrecognized app from starting. Running this app might put your PC at risk.

Fig.6: Given input file kaif.exe


File is Safe

The uploaded file "kaif.exe" is safe to use.

Fig.7: input file kaif.exe detected as safe


File is Safe

The uploaded file "kaif.exe" is safe to use.

Information about Entropy and Structural Patterns: Detailed analysis and results.

Fig.8: EDR analysis on input file kaif.exe

Table.1: Comparison Results

S. No	Data Model	Accuracy (%)
1	Deep Neural Networks (DNN) with Feature Selection	98.7
2	CNN and LSTM Hybrid Model	96.4
3	Multilayer Perceptron (MLP)	95.0
4	Basic CNN	93.5
5	Attention-Enhanced GRU Model	97.2
6	Deep Neural Network (DNN)wor	99.0

The table compares the accurate performance of various deep learning models for a classification task. The accurate performance of the Deep Neural Network (DNN) without feature selection is 99.0%, followed by the DNN with feature selection, 98.7%, and the Attention-Enhanced GRU Model at 97.2%. The other models, including the CNN and LSTM Hybrid Model (96.4%), the Multilayer Perceptron (95.0%), and the Basic CNN (93.5%), displayed a slightly lower performance.

VII. CONCLUSION

In this project, a Deep Neural Network was applied to classify executable files as malware or benign. The system was trained by comprehensive preprocessing and feature extraction and achieved high accuracy in detecting malicious files, thus establishing its speed and reliability. A diverse dataset was used for training so that the system could adapt to changes in malware. Some key evaluation metrics used to validate the model include precision, recall, and F1-score. It worked perfectly to avoid false positive cases and is well suited for applications in real-time cybersecurity scenarios.

The approach emphasized here underscores how deep learning can fortify EDR systems to contribute significantly to neighborhood cybersecurity since they would be very proactive in helping to reduce malware threats. With deep learning within the context of end-point protection, patterns that signify complex attacks or certain zero-day applications would thus be detected while reinforcing the overall posture of security protection implementations. Future improvements could include adding more datasets to vary the available data, carrying out analysis in real time, and linking the solution to cloud-based threat intelligence to further improve detection efficiency and adaptability against emergent threats.

REFERENCES

[1]. Raff, E., Barker, J., Sylvester, J., Brandon, R., Catanzaro, B., & Nicholas, C. (2018). Malware detection by eating a whole EXE. arXiv preprint arXiv:1710.09435. https://arxiv.org/abs/1710.09435

[2]. Anderson, H. S., & Roth, P. (2018). EMBER: An open dataset for training static PE malware machine learning models. arXiv preprint arXiv:1804.04637. https://arxiv.org/abs/1804.04637

[3]. Pascanu, R., Stokes, J. W., Sanossian, H., Marinescu, M., & Thomas, A. (2015). Malware classification with recurrent networks. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). https://ieeexplore.ieee.org/document/7178838

[4]. Johny, J. A., Vinod, P., Asmitha, K. A., Radhamani, G., Rehiman, R. K. A., & Conti, M. (2024). Deep Learning Fusion for Effective Malware Detection: Leveraging Visual Features. arXiv preprint arXiv:2405.14311. https://arxiv.org/abs/2405.14311

[5]. Gibert, D., Mateu, C., & Planes, J. (2018). The rise of machine learning for detection and classification of malware: Research developments, trends, and challenges. Journal of Network and Computer Applications, 153, 102526. https://doi.org/10.1016/j.jnca.2019.102526

[6]. Huang, W., & Stokes, J. W. (2016). Mtnet: A multi-task neural network for dynamic malware classification. arXiv preprint arXiv:1609.09063. https://arxiv.org/abs/1609.09063

[7]. Rezaei, A., Liu, Y., & Hu, J. (2020). Data augmentation techniques for improving performance in malware classification. Computers & Security, 94, 101822. https://doi.org/10.1016/j.cose.2020.101822

[8]. Kolosnjaji, B., Zarras, A., Webster, G., & Eckert, C. (2018). Deep learning for classification of malware system call sequences. Big Data Analytics in Cybersecurity. https://arxiv.org/abs/1612.07619

[9]. Firdausi, I., Lim, C., Erwin, A., & Nugroho, A. S. (2010). Analysis of machine learning techniques used in behavior-based malware detection. IEEE International Conference on Advanced Computer Science and Information Systems. https://ieeexplore.ieee.org/document/5626320
[10]. A. Sharma, P. Gupta, & R. Verma. (2024). A Survey of Malware Detection Using Deep Learning. arXiv. https://arxiv.org/html/2407.19153v1
[11]. K. Patel, M. Singh, & R. Kumar. (2023). Malware Detection in Executable Files Using Machine Learning. IEEE Xplore. https://ieeexplore.ieee.org/document/10113998/
[12]. Gupta, I., Kumari, S., Jha, P., & Ghosh, M. (2024). Leveraging LSTM and GAN for Modern Malware Detection. arXiv preprint arXiv:2405.04373. https://arxiv.org/abs/2405.04373
[13]. H. Li, Z. Zhang, & F. Wang. (2023). Enhanced Capsule Network-Based Executable Files Malware Detection. https://www.researchgate.net/publication/375066776_Enhanced_capsule_network-based_executable_files_malware_detection_and_classification-deep_learning_approach
[14]. Ahmed Bensaouda, Jugal Kalitaa and Mahmoud Bensaouda (2024). A Survey of Malware Detection Using Deep Learning. arXiv preprint arXiv:2407.19153. https://arxiv.org/abs/2407.19153
[15]. Alzaylaee, M. K., Yerima, S. Y., & Sezer, S. (2019). DL-Droid: Deep Learning-Based Android Malware Detection Using Real Devices. arXiv preprint arXiv:1911.10113. https://arxiv.org/abs/1911.10113
[16]. Ahuja, K., Verma, R., & Singh, P. (2022). Detecting malware using LSTM-based dynamic analysis. Journal of Cybersecurity. https://doi.org/10.1093/cybsec/tyac005
[17]. Rathore, H., Agarwal, S., Sahay, S. K., & Sewak, M. (2019). Malware Detection Using Machine Learning and Deep Learning. arXiv preprint arXiv:1904.02441. https://arxiv.org/abs/1904.02441
[18]. Rahman Ali, Asmat Ali,Farkhund Iqbal, Mohammed Hussain, and Farhan Ullah (2022). Deep Learning Methods for Malware and Intrusion Detection: A Systematic Literature Review. Wiley Online Library. https://onlinelibrary.wiley.com/doi/pdf/10.1155/2022/2959222
[19]. Sewak, M., Sahay, S. K., & Rathore, H. (2018). An Investigation of a Deep Learning-Based Malware Detection System. arXiv preprint arXiv:1809.05888. https://arxiv.org/abs/1809.05888
[20]. Fatemeh Deldar, Mahdi Abadi (2023). Deep Learning for Zero-day Malware Detection and Classification: A Survey. ACM Digital Library. https://dl.acm.org/doi/10.1145/3605775
[21]. Anderson, H., & Roth, P. (2018). EMBER: An Open Dataset for Training ML Models to Detect Malicious PE Files. arXiv preprint arXiv:1804.04637. https://arxiv.org/abs/1804.04637
[22]. R. Vinayakumar, Mamoun Alazab, K. P. Soma, Prabaharan Poornachandran, Sitalakshmi Venkatraman (2019). Robust intelligent malware detection using deep learning. IEEE Access. https://ieeexplore.ieee.org/document/8681127
[23]. Singh, N., Rao, P., & Kumar, S. (2021). Improving malware detection with synthetic oversampling and GANs. Future Generation Computer Systems. https://doi.org/10.1016/j.future.2021.10.008
[24]. Li, M., Yang, Z., & Wu, C. (2022). Lightweight deep learning model for edge-based malware detection. Journal of Computer Security. https://doi.org/10.3233/JCS-220042
[25]. Rommel Gutierrez, William Villegas-C, Lorena Naranjo Godoy, Aracely Mera-Navarrete, Sergio Luján-Mora (2022). Application of deep learning models for real-time automatic malware detection. IEEE Transactions on Information Forensics and Security. https://ieeexplore.ieee.org/document/10620215
[26]. Pascal Maniriho, Abdun Naser Mahmood, Mohammad Jabed Morshed Chowdhury (2022). Deep learning models for detecting malware attacks. arXiv preprint. https://arxiv.org/pdf/2209.03622
[27]. Mehta, S., Gupta, K., & Roy, T. (2023). Contrastive learning for malware detection: A self-supervised approach. Pattern Recognition Letters. https://doi.org/10.1016/j.patrec.2023.01.002
[28]. Xiaofei Xing, Xiang Jin, Haroon Elahi, Hai Jiang, Guojun Wang (2022). A malware detection approach using autoencoder in deep learning. IEEE Access. https://ieeexplore.ieee.org/document/9723074