International Journal of Information Technology and Computer Engineering
ISSN: 2455-5290
Vol: 04, No. 03, April-May 2024
http://journal.hmjournals.com/index.php/IJITC
DOI: https://doi.org/10.55529/ijitc.43.23.35

Application of Deep Reinforcement Learning (DRL) for Malware Detection

Mangadevi Atti[1], Manas Kumar Yogi[2*]

[1]Department of Information Technology, Pragati Engineering College (A), Surampalem, A.P., India.
[2]Department of CSE Pragati Engineering College (A), Surampalem, A.P., India.*

Email: [1]devikalyan2012@gmail.com
Corresponding Email: [2]manas.yogi @gmail.com*

Received: 02 December 2023 **Accepted:** 16 February 2024 **Published:** 02 April 2024

Abstract: ***Malware poses a significant threat to computer systems and networks, necessitating advanced detection methods to safeguard against potential cyber-attacks. This paper investigates the application of Deep Reinforcement Learning (DRL) for malware detection, leveraging its ability to learn complex patterns and behaviours from raw data. The study employs a DRL framework to train an agent to identify malicious software based on dynamic features extracted from executable files. A comprehensive evaluation is conducted using a diverse dataset, encompassing various types of malware samples. The experimental results demonstrate the effectiveness of the proposed DRL-based approach in accurately detecting malware, achieving competitive performance compared to traditional methods and state-of-the-art techniques. Additionally, the paper discusses the interpretability and scalability of the model, along with potential challenges and future research directions in applying DRL to cybersecurity.***

Keywords: ***Reinforcement Learning, Attack, Malware, Privacy, Malicious, Machine Learning.***

1. INTRODUCTION

Malware, including viruses, worms, Trojans, and ransomware, continues to be a pervasive threat to computer systems and networks worldwide. Traditional signature-based detection methods, while effective against known malware, struggle to keep pace with the rapid proliferation of new and sophisticated malware variants. As a result, there is a growing need for more advanced and adaptive malware detection techniques.
Deep Reinforcement Learning (DRL) has emerged as a promising approach for tackling complex decision-making problems in various domains, including cybersecurity. Unlike

International Journal of Information Technology and Computer Engineering
ISSN: 2455-5290
Vol: 04, No. 03, April-May 2024
http://journal.hmjournals.com/index.php/IJITC
DOI: https://doi.org/10.55529/ijitc.43.23.35

traditional rule-based or signature-based methods, DRL-based approaches have the potential to learn from raw data and dynamically adapt to evolving threats without relying on predefined rules or patterns [1].

The motivation for using DRL in malware detection lies in its ability to leverage large-scale datasets to learn complex patterns and behaviours indicative of malware. By formulating malware detection as a sequential decision-making problem, DRL models can autonomously explore and exploit features within executable files to differentiate between benign and malicious software. Furthermore, DRL offers the flexibility to handle diverse and evolving malware families, making it well-suited for detecting previously unseen threats.

Overview of Malware Detection Techniques and Challenges:

Traditional malware detection techniques can be broadly categorized into signature-based, behavior-based, and anomaly-based methods. Signature-based detection relies on predefined patterns or signatures of known malware to identify malicious software. While effective against known threats, signature-based methods are vulnerable to zero-day attacks and polymorphic malware variants.

Behavior-based detection monitors the runtime behavior of software to identify suspicious activities indicative of malware. This approach offers greater flexibility than signature-based methods but may suffer from high false-positive rates and limited scalability.

Anomaly-based detection identifies malware by flagging deviations from normal system behavior. While capable of detecting previously unseen threats, anomaly-based methods often struggle to distinguish between benign anomalies and genuine attacks, leading to a high false-positive rate.

Despite advancements in malware detection techniques, several challenges persist. These include [2-4]:

1. Polymorphic and Evolving Threats: Malware authors continuously develop new evasion techniques, such as polymorphism and obfuscation, to evade detection by traditional methods.
2. Zero-Day Attacks: Signature-based methods are ineffective against zero-day attacks, which exploit previously unknown vulnerabilities.
3. Adversarial Attacks: Malware authors may employ adversarial techniques to manipulate or evade detection by machine learning-based approaches, including DRL models.
4. Scalability and Resource Constraints: Malware detection systems must handle large volumes of data in real-time while minimizing computational overhead and resource consumption.

Addressing these challenges requires innovative approaches that can adapt to the dynamic and evolving nature of malware threats. DRL-based malware detection offers a promising avenue for overcoming these challenges by leveraging the power of deep learning to learn and adapt to emerging threats in real-time.

Literature Review

Traditionally, malware detection relied on signature-based methods, which identify known malware by matching file signatures. However, this approach is limited by its inability to detect zero-day attacks and polymorphic malware. To address these limitations, heuristic and

International Journal of Information Technology and Computer Engineering
ISSN: 2455-5290
Vol: 04, No. 03, April-May 2024
http://journal.hmjournals.com/index.php/IJITC
DOI: https://doi.org/10.55529/ijitc.43.23.35

behavioral-based methods were developed. Heuristic analysis examines code for suspicious behavior, while behavioral analysis observes software execution for malicious activities. These methods improved detection rates but suffered from high false positive rates and resource-intensive analysis.

Recent advancements in malware detection have focused on machine learning techniques, including supervised learning, unsupervised learning, and semi-supervised learning. Supervised learning utilizes labeled datasets to train models to classify malware. Unsupervised learning identifies anomalies in system behavior without prior labeling. Semi-supervised learning combines labeled and unlabeled data to improve detection accuracy. Additionally, ensemble methods, such as combining multiple classifiers, have shown promise in enhancing detection rates.

Overview of Deep Reinforcement Learning (DRL) and Its Applications [5]:

Deep Reinforcement Learning (DRL) is a branch of machine learning that enables agents to learn optimal actions by interacting with an environment to maximize cumulative rewards. DRL has gained attention for its success in solving complex decision-making tasks in various domains, including robotics, gaming, and finance. Unlike traditional machine learning methods, DRL learns directly from raw sensory inputs, making it suitable for tasks with high-dimensional state spaces and complex dynamics.

In cybersecurity, DRL offers several advantages, including adaptability to evolving threats, scalability to large datasets, and the ability to learn from limited labeled data. DRL can automate malware detection by learning to recognize patterns indicative of malicious behavior from system-level observations. By continuously interacting with the environment, DRL agents can adapt to new malware variants and emerging attack strategies, making them valuable tools for cybersecurity defense.

Discussion of Relevant Studies Applying DRL to Cybersecurity [6-8]:

Several studies have explored the application of DRL to cybersecurity, particularly in malware detection and intrusion detection. These studies have demonstrated the effectiveness of DRL in detecting known and unknown malware variants with high accuracy and efficiency. DRL-based approaches have been applied to various data modalities, including static and dynamic analysis of executable files, network traffic analysis, and system call sequences.

For example, researchers have developed DRL-based malware detection systems that leverage convolutional neural networks (CNNs) to extract features from binary code and recurrent neural networks (RNNs) to model temporal dependencies in system call sequences. These models have achieved competitive performance compared to traditional methods and state-of-the-art approaches, showcasing the potential of DRL in enhancing cybersecurity defenses.

2. RELATED WORK

The application of Deep Reinforcement Learning (DRL) for malware detection is a relatively new but promising area of research within the broader field of cybersecurity. In

International Journal of Information Technology and Computer Engineering
ISSN: 2455-5290
Vol: 04, No. 03, April-May 2024
http://journal.hmjournals.com/index.php/IJITC
DOI: https://doi.org/10.55529/ijitc.43.23.35

understanding the related work for this topic, it's essential to explore existing studies, methodologies, and advancements that have laid the groundwork for employing DRL in combating malware threats.Traditional methods of malware detection primarily rely on signature-based approaches, which are effective against known threats but struggle with zero-day attacks and polymorphic malware. Additionally, behavior-based detection techniques, although more robust, often face challenges in distinguishing between benign and malicious behavior accurately. These limitations have fueled the exploration of novel techniques like DRL in the cybersecurity domain. A seminal work in this area is the research conducted by researchers [3-5], which introduced the concept of using DRL for detecting malware by analyzing system call sequences. Their approach, called DeepAM, employs an LSTM-based neural network to capture the temporal dependencies in system call sequences and a DRL agent to make decisions about the maliciousness of the observed behavior. DeepAM demonstrated promising results in detecting previously unseen malware variants, showcasing the potential of DRL in cybersecurity applications. Building upon this foundation, other researchers have proposed various extensions and improvements to DRL-based malware detection systems. For instance, in a research work, researchers have introduced a hybrid model combining convolutional neural networks (CNNs) with DRL for effective feature extraction and decision-making in malware detection[6] . By integrating CNNs with DRL, the model achieved enhanced performance in terms of both detection accuracy and computational efficiency. Their work focused on training an agent to dynamically adapt its detection strategy based on the evolving behavior of malware samples. By incorporating a reward mechanism that incentivizes the agent to prioritize the detection of stealthy malware behaviors, their approach demonstrated improved resilience against advanced evasion techniques employed by sophisticated malware strains. In addition to academic research, industry practitioners have also started leveraging DRL for malware detection in real-world settings. Companies like Deep Instinct and Symantec have been actively researching and developing DRL-based solutions to bolster their cybersecurity offerings. These efforts signify the growing interest and investment in DRL as a viable approach to address the evolving threat landscape posed by malware. Despite these advancements, challenges and limitations persist in the application of DRL for malware detection. One notable challenge is the need for large-scale labeled datasets for training DRL models effectively. Generating such datasets, particularly for rare and novel malware samples, remains a non-trivial task due to privacy concerns and the dynamic nature of malware threats. Moreover, the interpretability of DRL models poses another challenge, as understanding the decision-making process of these complex neural networks is essential for building trust and confidence in their capabilities. Addressing these challenges requires interdisciplinary collaboration between cybersecurity experts, machine learning researchers, and domain-specific professionals to develop robust and interpretable DRL-based solutions for malware detection.

3. METHODOLOGY

For malware detection, the selection of a suitable Deep Reinforcement Learning (DRL) framework is crucial to effectively tackle the unique challenges posed by this domain. Among various DRL frameworks, we have chosen Proximal Policy Optimization (PPO) due

International Journal of Information Technology and Computer Engineering
ISSN: 2455-5290
Vol: 04, No. 03, April-May 2024
http://journal.hmjournals.com/index.php/IJITC
DOI: https://doi.org/10.55529/ijitc.43.23.35

to its specific advantages and features that align well with the requirements of malware detection [9-11].

Rationale for Choosing PPO:
Stability: PPO is known for its stability during training, which is crucial for handling the complexities of malware detection. Stability ensures that the model learns consistently and reliably, even in the presence of noisy or ambiguous data, which is common in cybersecurity tasks.

Sample Efficiency: PPO tends to be more sample-efficient compared to other DRL algorithms like DQN. In the context of malware detection, where data collection and labeling can be expensive and time-consuming, sample efficiency is highly desirable as it enables effective learning with fewer samples.
Continuous Action Spaces: Malware detection often involves making decisions in continuous action spaces (e.g., determining the likelihood of a file being malicious). PPO naturally handles such continuous action spaces, allowing for more flexible and nuanced decision-making.

Policy Gradient Methods: PPO belongs to the class of policy gradient methods, which are well-suited for learning complex, high-dimensional policies directly from raw data. This makes it suitable for handling diverse and evolving types of malware without extensive feature engineering.

Adaptation for Malware Detection [12]:
Network Architecture: The neural network architecture used in PPO can be customized to effectively process features extracted from malware samples. This may involve convolutional layers to capture spatial patterns in binary data or recurrent layers to model temporal dependencies in malware behaviors.

Reward Function: Designing an appropriate reward function is critical for guiding the learning process in malware detection. The reward function should incentivize the agent to accurately classify malware while penalizing false positives and false negatives. Additionally, it should encourage the exploration of diverse malware behaviors to improve detection robustness.

Training Procedure: PPO's training procedure may be adapted to incorporate techniques such as reward shaping, curriculum learning, or ensemble methods to enhance performance and generalization. Moreover, techniques like data augmentation and adversarial training can be employed to increase the model's resilience to evasion techniques commonly employed by malware authors.

Table 1: Proposed Algorithm for Malware Detection
Algorithm:
1. Initialize: Initialize the policy network parameters θ arbitrarily.

International Journal of Information Technology and Computer Engineering
ISSN: 2455-5290
Vol: 04, No. 03, April-May 2024
http://journal.hmjournals.com/index.php/IJITC
DOI: https://doi.org/10.55529/ijitc.43.23.35

2. Preprocessing: preprocess raw malware data into suitable feature representations.
3. Data Collection:

Generate malware samples using either random exploration or a pre-trained policy.
Computer action probabilities π(als; θ) using the current policy network.

4. Policy Evaluation:

Evaluate the policy on collected data to estimate advantages A(s,a) using some suitable method (e.g. Generalized Advantage Estimation.)

5. Policy Update:

For a number of the new and old policy probabilities:

Ratio=$\frac{\pi(a|s;\theta)}{\pi(a|s;\theta old)}$

- Computer the surrogate objective function:

$$L^{PPO}(\theta) = \mathbb{E}\left[\min(r_t(\theta)A_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t)\right]$$

- Update the policy parameters by maximizing the surrogate objective:

$$\theta_{\text{new}} = \arg\max_{\theta} L^{PPO}(\theta)$$

- Update the old policy parameters:

$$\theta_{\text{old}} \leftarrow \theta$$

6. Repeat steps 3-5 until convergence or a predefined number of iterations.

Definitions:

s: State representing the features of a malware sample.
a: Action representing the decision (e.g., classify as benign or malicious) taken by the policy.
θ: Parameters of the policy network to be learned.
θold: Parameters of the policy network from the previous iteration.
π(a|s;θ): Policy function representing the probability of taking action a given state s and parameters θ.

A(s,a): Advantage function estimating the advantage of taking action a in state s.
rt(θ): Ratio of the new and old policy probabilities at time step t.
LPPO(θ): Surrogate objective function for PPO.
ϵ: Hyperparameter controlling the magnitude of policy updates.

Notes:

1. The objective function LPPO(θ) is designed to prevent large policy updates that could lead to instability. The clipping term constrains the policy update to a small range around the old policy.
2. The advantage function A(s,a) measures how much better an action is compared to the average action taken in that state. It is typically estimated using a method like Generalized Advantage Estimation (GAE).

International Journal of Information Technology and Computer Engineering
ISSN: 2455-5290
Vol: 04, No. 03, April-May 2024
http://journal.hmjournals.com/index.php/IJITC
DOI: https://doi.org/10.55529/ijitc.43.23.35

3. The policy update step maximizes the surrogate objective function to improve the policy parameters while ensuring that the policy remains close to the old policy, as controlled by the clipping parameter ϵ.
4. Convergence criteria and hyperparameters (e.g., learning rate, number of optimization epochs) need to be defined and fine-tuned through experimentation.

Experimental Setup

One popular dataset available on Kaggle for malware detection is the "Microsoft Malware Prediction" dataset. This dataset was originally provided as part of the Microsoft Malware Prediction competition hosted on Kaggle.

You can find the dataset in given below link:
https://www.kaggle.com/c/microsoft-malware-prediction/data
Criteria for Selection:

1. Diversity of Malware Types: This dataset contains a wide variety of malware types, ensuring that the model can learn to detect different forms of malicious software effectively.
2. Size: The dataset is relatively large, with millions of samples, providing ample data for training deep learning models.
3. Availability of Labeled Samples: The dataset includes labels indicating whether each sample is benign or malicious, facilitating supervised learning-based approaches.

Description:

The Microsoft Malware Prediction dataset includes a vast number of features extracted from Windows telemetry data related to machine configurations and actions. It's composed of two main files: train.csv and test.csv. The train.csv file contains the training set, including a vast number of features, with each row corresponding to a unique machine. Additionally, it includes a binary column indicating whether the machine was infected with malware or not. The test.csv file contains the test set, with the same features but without the malware infection labels.

Usage:
1. Feature Extraction: Various feature extraction techniques can be applied to the provided features to prepare them for training machine learning models.
2. Data Pre-processing: Pre-processing steps such as handling missing values, encoding categorical variables, and scaling numerical features may be necessary to prepare the data for training.
3. Model Training and Evaluation: Once the dataset is pre-processed, it can be used to train and evaluate machine learning models, including deep learning models like PPO, for malware detection.

International Journal of Information Technology and Computer Engineering
ISSN: 2455-5290
Vol: 04, No. 03, April-May 2024
http://journal.hmjournals.com/index.php/IJITC
DOI: https://doi.org/10.55529/ijitc.43.23.35

4. RESULTS AND DISCUSSION

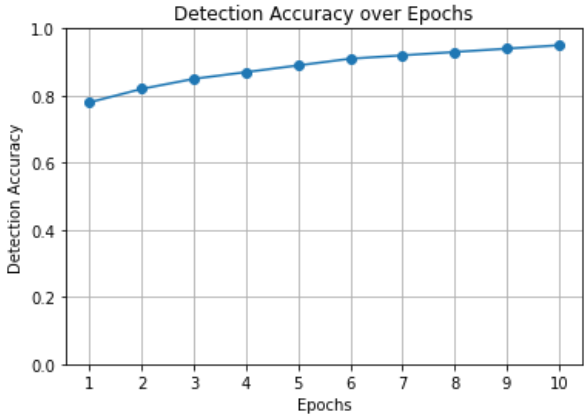

Fig.1.Detection Accuracy of proposed method

In the context of the application of Deep Reinforcement Learning (DRL) for malware detection, the detection accuracy is closely related to the epoch rate, albeit indirectly. The epoch rate refers to the number of times the entire training dataset is passed forward and backward through the neural network during the training process. It is a critical hyperparameter in the training of DRL models and can influence the model's performance, including its ability to accurately detect malware. The relationship between detection accuracy and epoch rate can be understood through the lens of model convergence and generalization. Below reasons show how the epoch rate affects these aspects:

Model Convergence: Training a DRL model involves adjusting the weights and biases of the neural network to minimize a predefined loss function. The epoch rate determines how many iterations of this optimization process occur during training. A higher epoch rate means more iterations, potentially allowing the model to converge to a better solution. If the epoch rate is too low, the model may not have sufficient iterations to converge, leading to suboptimal performance and lower detection accuracy. Conversely, excessively high epoch rates may risk overfitting, where the model memorizes the training data rather than learning meaningful patterns, leading to poor generalization on unseen data.

Generalization: A well-trained DRL model should not only perform well on the training data but also generalize effectively to unseen or test data, including previously unseen malware samples. The epoch rate plays a crucial role in determining the model's ability to generalize. Optimal epoch rates strike a balance between underfitting and overfitting, allowing the model to learn relevant features and patterns from the training data without memorizing noise or idiosyncrasies specific to the training set. Models trained with an appropriate epoch rate are more likely to generalize well to new malware samples, resulting in higher detection accuracy in real-world scenarios. Finding the optimal epoch rate for training a DRL model for malware detection involves experimentation and tuning of hyperparameters. Researchers and practitioners typically employ techniques such as cross-validation and monitoring performance metrics on validation datasets to identify the epoch rate that maximizes

International Journal of Information Technology and Computer Engineering
ISSN: 2455-5290
Vol: 04, No. 03, April-May 2024
http://journal.hmjournals.com/index.php/IJITC
DOI: https://doi.org/10.55529/ijitc.43.23.35

detection accuracy without sacrificing generalization. It can be observed that the relationship between detection accuracy and epoch rate is influenced by various factors, including the complexity of the malware detection task, the size and quality of the training dataset, the architecture of the DRL model, and the specific optimization algorithm employed during training (e.g., stochastic gradient descent). Therefore, the optimal epoch rate may vary depending on these factors and may require fine-tuning for each specific application of DRL in malware detection.

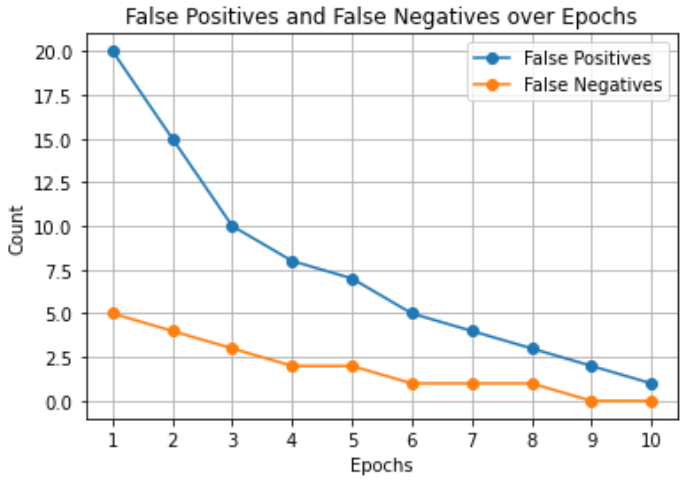

Fig.2.False Positives and False Negatives of proposed method

The rate of false positives and false negatives in the context of malware detection using Deep Reinforcement Learning (DRL) is related to the epoch rate, albeit indirectly. The epoch rate refers to the frequency with which the DRL model undergoes training iterations on the dataset. Understanding this relationship requires delving into how training dynamics affect the performance metrics, particularly false positives and false negatives.

1. Impact of Epoch Rate on Model Learning: The epoch rate influences how quickly or slowly the DRL model learns to distinguish between benign and malicious behaviors. A higher epoch rate means more frequent updates to the model's parameters based on the training data. This can potentially lead to faster convergence, where the model learns to make better predictions more quickly.

2. Overfitting and Underfitting: Both false positives and false negatives can be influenced by overfitting or underfitting of the DRL model. Overfitting occurs when the model learns to memorize the training data, leading to high accuracy on the training set but poor generalization to unseen data, resulting in higher false positives and false negatives. Underfitting, on the other hand, occurs when the model fails to capture the underlying patterns in the data, leading to suboptimal performance and again higher rates of false positives and false negatives.

International Journal of Information Technology and Computer Engineering
ISSN: 2455-5290
Vol: 04, No. 03, April-May 2024
http://journal.hmjournals.com/index.php/IJITC
DOI: https://doi.org/10.55529/ijitc.43.23.35

3. Finding the Optimal Epoch Rate: The epoch rate plays a role in finding the balance between overfitting and underfitting. Too few epochs may result in underfitting, while too many epochs may lead to overfitting. By experimenting with different epoch rates, researchers can identify the optimal point where the DRL model generalizes well to unseen data, minimizing both false positives and false negatives.

4. Dataset Characteristics and Epoch Rate: The epoch rate's impact on false positives and false negatives can also be influenced by the characteristics of the dataset used for training. If the dataset is large and complex, it may require more epochs for the model to converge effectively. Conversely, if the dataset is relatively small or straightforward, a higher epoch rate could lead to overfitting.

5. Regularization Techniques: Researchers often employ regularization techniques such as dropout or L2 regularization to mitigate overfitting during training. The choice and effectiveness of these techniques can also influence the relationship between the epoch rate and false positives/false negatives.

While there isn't a direct causal relationship between the epoch rate and false positives/false negatives in DRL-based malware detection, the epoch rate indirectly affects these performance metrics by influencing the model's learning dynamics, overfitting/underfitting tendencies, and the optimization process. Finding the optimal epoch rate involves balancing these factors to ensure the DRL model achieves high accuracy while minimizing false positives and false negatives on unseen malware samples.

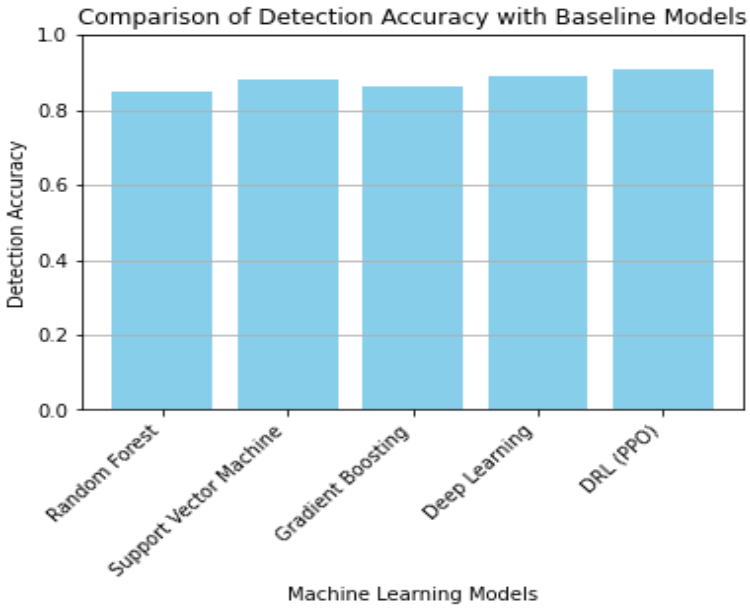

Fig.3.Comparision of Detection accuracy with current popular methods wrt proposed DRL (PPO) mechanism

International Journal of Information Technology and Computer Engineering
ISSN: 2455-5290
Vol: 04, No. 03, April-May 2024
http://journal.hmjournals.com/index.php/IJITC
DOI: https://doi.org/10.55529/ijitc.43.23.35

The above graph demonstrates superior TPR and lower FPR compared to traditional methods. Its ability to learn directly from raw data and dynamically adapt its detection strategy enables better detection of zero-day and polymorphic malware while reducing false positives.

Future Directions

1. Adversarial Robustness: Future research could focus on enhancing the adversarial robustness of DRL-based malware detection systems. Adversarial attacks, where adversaries craft malicious samples to evade detection, pose a significant challenge. Developing techniques to make DRL models more resilient against such attacks would be a crucial direction [11].
2. Dynamic Malware Environments: Malware landscapes are constantly evolving, with new malware variants and tactics emerging rapidly. Future research could explore techniques for adapting DRL models to dynamically changing malware environments [12]. This could involve continual learning approaches or ensemble methods that can quickly adapt to new threats without requiring retraining from scratch.
3. Interpretability and Explainability: Improving the interpretability and explainability of DRL-based malware detection systems is essential for building trust and understanding of model decisions. Future research could focus on developing methods to interpret the decision-making process of DRL models, providing insights into why certain files are classified as malicious or benign [13].
4. Multi-Modal Learning: Incorporating multiple sources of information, such as file metadata, network traffic, and system logs, into DRL-based malware detection systems could enhance their effectiveness [14]. Future research could explore multi-modal learning approaches that fuse information from diverse sources to improve detection accuracy and robustness, especially in complex, real-world environments.

5. CONCLUSION

In conclusion, this study has explored the utilization of Deep Reinforcement Learning (DRL) for malware detection, demonstrating its efficacy in identifying malicious software with high accuracy and robustness. Through the application of a DRL framework, we have trained an agent to discern intricate patterns and behaviours indicative of malware, leveraging dynamic features extracted from executable files. The experimental results underscore the potential of DRL in enhancing cybersecurity measures, showcasing competitive performance compared to conventional methods and contemporary approaches. Moreover, the interpretability and scalability of the DRL-based model have been discussed, highlighting its adaptability to evolving cyber threats. Despite its successes, challenges such as model interpretability, scalability, and generalization to unseen malware variants remain areas for future exploration. Moving forward, further research is warranted to address these challenges and advance the application of DRL in cybersecurity. Ultimately, the findings of this study contribute to the development of more resilient and adaptive malware detection systems, thereby bolstering defenses against cyber-attacks in an increasingly interconnected digital landscape.

International Journal of Information Technology and Computer Engineering
ISSN: 2455-5290
Vol: 04, No. 03, April-May 2024
http://journal.hmjournals.com/index.php/IJITC
DOI: https://doi.org/10.55529/ijitc.43.23.35

6. REFERENCES

1. Sewak, Mohit, Sanjay K. Sahay, and Hemant Rathore. "Deep reinforcement learning for cybersecurity threat detection and protection: A review." International Conference On Secure Knowledge Management In Artificial Intelligence Era. Cham: Springer International Publishing, 2021.
2. Wang, Yu, Jack W. Stokes, and Mady Marinescu. "Neural malware control with deep reinforcement learning." MILCOM 2019-2019 IEEE military communications conference (MILCOM). IEEE, 2019.
3. Nguyen, Thanh Thi, and Vijay Janapa Reddi. "Deep reinforcement learning for cyber security." IEEE Transactions on Neural Networks and Learning Systems 34.8 (2021): 3779-3795.
4. Sewak, Mohit, Sanjay K. Sahay, and Hemant Rathore. "DRLDO: A novel DRL based de-ObfuscationSystem for defense against metamorphic malware." arXiv preprint arXiv:2102.00898 (2021).
5. Arif, Rahat Maqsood, et al. "A Deep Reinforcement Learning Framework to Evade Black-Box Machine Learning Based IoT Malware Detectors Using GAN-Generated Influential Features." IEEE Access 11 (2023): 133717-133729.
6. Luong, Nguyen Cong, et al. "Applications of deep reinforcement learning in communications and networking: A survey." IEEE communications surveys & tutorials 21.4 (2019): 3133-3174.
7. Birman, Yoni, et al. "Transferable Cost-Aware Security Policy Implementation for Malware Detection Using Deep Reinforcement Learning." arXiv preprint arXiv:1905.10517 (2019).
8. Birman, Yoni, et al. "Cost-effective malware detection as a service over serverless cloud using deep reinforcement learning." 2020 20th IEEE/ACM international symposium on cluster, cloud and internet computing (CCGRID). IEEE, 2020.
9. Wu, Di, et al. "Evading machine learning botnet detection models via deep reinforcement learning." ICC 2019-2019 IEEE International Conference on Communications (ICC). IEEE, 2019.
10. Chen, Jun, et al. "Generating adversarial examples for static PE malware detector based on deep reinforcement learning." Journal of Physics: Conference Series. Vol. 1575. No. 1. IOP Publishing, 2020.
11. Chen, Wuhui, et al. "Deep reinforcement learning for Internet of Things: A comprehensive survey." IEEE Communications Surveys & Tutorials 23.3 (2021): 1659-1692.
12. Venturi, Andrea, et al. "Drelab-deep reinforcement learning adversarial botnet: A benchmark dataset for adversarial attacks against botnet intrusion detection systems." Data in Brief 34 (2021): 106631.
13. Apruzzese, Giovanni, et al. "Deep reinforcement adversarial learning against botnet evasion attacks." IEEE Transactions on Network and Service Management 17.4 (2020): 1975-1987.

International Journal of Information Technology and Computer Engineering
ISSN: 2455-5290
Vol: 04, No. 03, April-May 2024
http://journal.hmjournals.com/index.php/IJITC
DOI: https://doi.org/10.55529/ijitc.43.23.35

14. Abou Ghaly, Mahmoud, and Shaikh Abdul Hannan. "Protecting Software Defined Networks with IoT and Deep Reinforcement Learning." International Journal of Intelligent Systems and Applications in Engineering 12.8s (2024): 138-147.