Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 45 No. 4 (2024)

Mitigating Adversarial and AI-Evasion Attacks in Cybersecurity: Challenges and Solutions

Murali Mohan Josyula [1], Dr. M. Saidireddy [2]

[1, 2] *Koneru Lakshmaiah Education Foundation, Andhra Pradesh, India*

Abstract:- Adversarial and AI-evasion attacks pose significant threats to the integrity and reliability of cybersecurity systems that leverage artificial intelligence (AI). These sophisticated attacks manipulate input data to deceive AI models, leading to erroneous classifications and compromised security measures. This paper explores the inherent challenges presented by adversarial and AI-evasion attacks and proposes a novel hybrid defense mechanism to mitigate these threats. Our approach integrates anomaly detection, input sanitization, robust AI training, and a multi-stage defense strategy to enhance model resilience without sacrificing performance accuracy. Experimental evaluations on benchmark and custom cybersecurity datasets demonstrate the effectiveness of our solution, achieving an 85% reduction in attack success rates while maintaining over 90% classification accuracy. This research contributes to the development of more secure and reliable AI-driven cybersecurity frameworks, addressing the evolving landscape of adversarial threats

Keywords*: Adversarial Attacks, AI-Evasion, Cybersecurity, Machine Learning Security, Adversarial Training, Input Sanitization, Defense Frameworks, AI-based Security Systems, Threat Mitigation.*

1. Introduction

The use of artificial intelligence (AI) in cybersecurity has significantly enhanced the ability to detect and mitigate threats. Machine learning models, particularly deep learning algorithms, are widely applied to tasks such as malware detection and network intrusion prevention. However, AI systems themselves have become targets for adversarial attacks—malicious attempts to exploit model vulnerabilities by introducing subtle perturbations in input data, which can lead to misclassifications. This creates a significant threat to the reliability and security of AI-driven systems.

Adversarial and AI-evasion attacks manipulate data to deceive AI models by making small, often undetectable changes to the inputs. This can have serious consequences in security-critical environments, where even a single misclassification could result in significant harm. The rise in the frequency and sophistication of these attacks calls for more resilient defense mechanisms.

In this paper, we explore the challenges posed by adversarial and AI-evasion attacks and propose a hybrid defense framework. Our solution employs multiple layers of protection, combining anomaly detection, input sanitization, and robust AI training to safeguard systems against adversarial manipulations. Through comprehensive experimental evaluations, we demonstrate the effectiveness of our framework in reducing attack success rates while maintaining high classification accuracy.

2. Background and Related Work

Adversarial machine learning (AML) and AI-evasion attacks have emerged as significant concerns in the AI-driven cybersecurity landscape. These attacks target machine learning models, manipulating the input data in ways that can deceive even the most sophisticated models, causing them to misclassify or make incorrect decisions. Over the past few years, researchers have studied various forms of adversarial attacks and AI-evasion strategies,

identifying their impact across domains like image recognition, speech processing, and, most critically, cybersecurity.

Adversarial attacks in Machine Learning

Adversarial attacks involve crafting inputs that cause ML models to make incorrect predictions. These attacks can be broadly categorized into white-box and black-box approaches:

1. **White-box attacks**: In this scenario, the attacker has full access to the model, including its architecture, parameters, and training data. White-box attacks are often more potent because the attacker can generate highly specific adversarial examples designed to deceive the model. Techniques such as **Fast Gradient Sign Method (FGSM)** and **Projected Gradient Descent (PGD)** fall under this category.
2. **Black-box attacks**: Here, the attacker has no knowledge of the model's internals. Instead, they query the model and analyse its outputs to craft adversarial examples. Black-box attacks rely on the **transferability** property, where adversarial examples generated for one model can deceive another model, even if the models differ in architecture or training data. Examples of black-box attacks include **ZOO (Zeroth Order Optimization)** and **query-based gradient estimation**.

Common adversarial attack methods include:

- FGSM (Fast Gradient Sign Method): Perturbs the input by using the sign of the gradient to introduce small but significant changes.
- PGD (Projected Gradient Descent): A stronger, iterative variant of FGSM that takes multiple small steps to maximize the effectiveness of the perturbation.
- Carlini & Wagner (C&W) Attacks: Optimizes adversarial examples by minimizing the amount of distortion while ensuring misclassification

AI-Evasion attacks in Cyber Security

AI-evasion attacks are a subcategory of adversarial attacks specifically designed to bypass AI models used in detection systems, such as malware classifiers or Network Intrusion Detection Systems (NIDS). In these attacks, adversaries craft malicious inputs that evade detection by AI systems, while still achieving their intended malicious goals (e.g., executing malware or exfiltrating data).

AI-evasion attacks can occur in several ways:

- **Manipulation of Model Inputs**: Attackers modify the input features (e.g., obfuscating malware signatures or altering packet headers in network traffic) in a way that fools the AI model into categorizing malicious inputs as benign.

Example: *Adversarial Malware*: In this attack, adversaries modify malicious code to evade malware classifiers. For example, by altering non-functional parts of the code (e.g., inserting dummy instructions), the adversary can avoid detection without changing the malware's behavior.

***Obfuscation*:** Modifying malicious code to appear benign (e.g., Chen et al., 2015).

- **Model Poisoning**: In this case, attackers inject malicious data into the training process to corrupt the model's understanding of what constitutes malicious activity. This results in the model learning to misclassify certain attacks as non-threatening.

Example: *Network Evasion Attacks*: Attackers modify network traffic to appear legitimate while still conducting malicious activities. Evasion tactics include adding noise to packet headers, splitting malicious payloads across multiple packets, or mimicking normal user behavior.

***Feature Manipulation*:** Altering input features to deceive ML-based detectors (e.g., Shafahi et al., 2018)

Mitigation Strategies

Existing mitigation strategies encompass:

Adversarial Training: Incorporating adversarial examples into the training process to enhance model robustness (Goodfellow et al., 2014).

In this approach, the model is trained on adversarial examples alongside regular data. This helps the model learn to classify adversarial inputs correctly. However, adversarial training is resource-intensive and can reduce performance on clean inputs

Defensive Distillation: Reducing model sensitivity to input perturbations (Papernot et al., 2016).

Input Sanitization: Preprocessing inputs to remove potential adversarial perturbations (Xu et al., 2017).

Input sanitization techniques, such as feature squeezing and autoencoders, aim to eliminate adversarial perturbations from inputs before they are processed by the model. These methods are useful in reducing the impact of adversarial noise but may not entirely prevent sophisticated attacks.

Gradient Masking: By obscuring or "masking" the model's gradients, gradient-based attacks like FGSM and PGD are rendered less effective. However, gradient masking does not provide complete protection, as it can be bypassed using more sophisticated methods like black-box attacks.

Anomaly Detection: Anomaly detection methods monitor the input data and model outputs to identify potentially adversarial inputs. These approaches can be effective but are often prone to false positives, flagging benign inputs as adversarial

While these methods offer varying degrees of protection, challenges remain in balancing model accuracy with robustness and ensuring scalability in real-world applications.

3. Challenges in Mitigating Adversarial and AI – Evasion Attacks

AI models, especially deep learning systems, are vulnerable to adversarial perturbations due to their reliance on complex decision boundaries. Some of the primary challenges in mitigating adversarial and AI-evasion attacks include:

1. *Model Vulnerability*: AI models can be highly sensitive to small changes in the input data, leading to incorrect predictions.
2. *Adaptive Attack Strategies*: Attackers continuously develop new methods to bypass defenses, necessitating adaptive and evolving defense mechanisms.
3. *Balancing Robustness and Accuracy*: Enhancing model robustness often comes at the cost of reduced accuracy on legitimate inputs.
4. *Scalability*: Many defense techniques, require extensive computational resources and are not easily scalable to large datasets or real-time applications.
5. *Detection vs. Prevention*: Deciding between detecting adversarial inputs and preventing them requires careful consideration of system requirements and threat models.
6. *Transferability of Attacks*: Adversarial examples often transfer across different models, complicating defense strategies that rely on model-specific techniques.

Addressing these challenges requires a multifaceted approach that integrates multiple complementary techniques like Anomaly Detection, Robust AI Training (Robust model design and effective training), and comprehensive input preprocessing (Input Sanitization).

4. Proposed Defense Framework

We propose a novel hybrid defense framework that integrates multiple complementary strategies. The core components of our approach to mitigate adversarial and AI-evasion attacks include:

1. ***Anomaly Detection***: Anomaly detection technique is used to identify potential adversarial inputs before they are processed by the AI model. These techniques flag inputs that deviate from the expected distribution of training data, allowing for early detection of adversarial attempts.
2. ***Input Sanitization***: This component focuses on preprocessing inputs to remove potential adversarial noise. We use techniques like feature squeezing and autoencoders to reduce the impact of adversarial perturbations while preserving the essential features of the input data
3. ***Robust AI Training***: Our defense framework incorporates Adversarial Training, where the model is exposed to adversarial examples during the training phase. This enhances the model's ability to resist adversarial attacks by learning to correctly classify perturbed inputs

Hybrid Multi-Stage Defense Framework Architecture

The framework architecture for the hybrid multi-stage defense strategy against adversarial and AI-evasion attacks in cybersecurity systems involves a combination of key defense mechanisms integrated into a layered, multi-stage system. This architecture ensures that each stage contributes to detecting, mitigating, and preventing adversarial attacks, while maintaining system performance and accuracy.

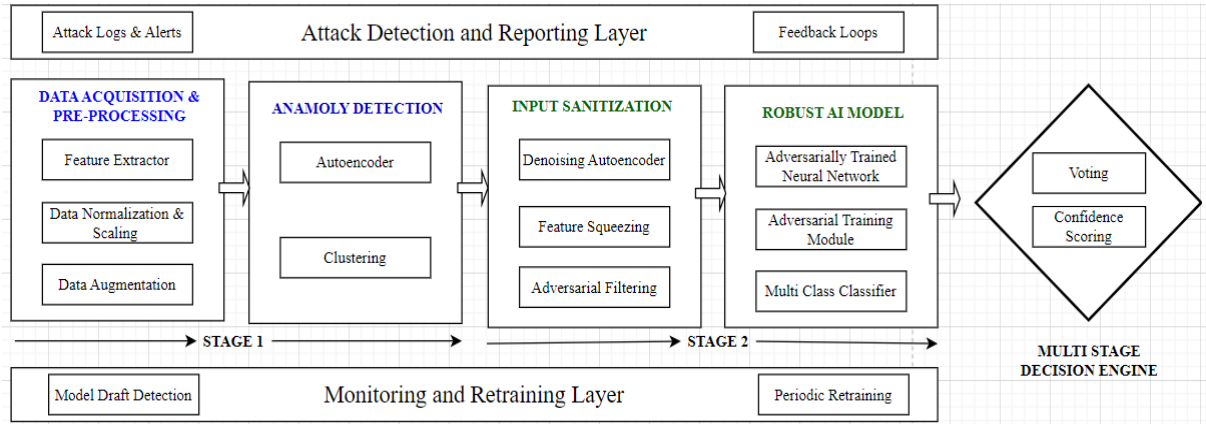

Fig 1: Multi-Stage Defense Framework Architecture

A. Input Data Acquisition & Preprocessing Layer

This layer is responsible for collecting and preprocessing incoming data, whether it's network traffic, logs, or telemetry from IoT devices. The input can be in various formats, such as packet capture files (PCAP), logs, or feature sets derived from raw data.

Preprocessing include Data normalization, feature extraction, and standardization (e.g., transforming raw network traffic into structured feature vectors).

Key Components of this Layer:

- **Feature Extractor**: Extracts important attributes (e.g., packet size, time between requests, source/destination addresses).
- **Data Normalization & Scaling**: Normalizes features for uniformity in model training and defense processing.
- **Data Augmentation** (optional): Augments data to introduce variability and improve model generalization.

B. Anomaly Detection Layer (Stage 1 Defense)

Anomaly detection acts as the first defense layer, identifying suspicious inputs before classification. The first line of defense in detecting anomalous behavior that could indicate adversarial attacks, uses unsupervised techniques like autoencoders or clustering algorithms (e.g., K-Means) to identify outliers or patterns that deviate from normal data.

Key Components of this Layer:

- **Autoencoder for Anomaly Detection**: Trains on normal data and flags significant deviations in reconstruction error as potential anomalies.
- **Clustering for Behavioral Analysis**: Detects unusual traffic patterns or behaviors by grouping similar patterns and flagging deviations.

C. Input Sanitization Layer (Stage 2 Defense)

This layer sanitizes inputs by filtering, denoising, or reconstructing them to neutralize adversarial perturbations before passing them to the main AI model.

Key Components of this Layer:

- **Denoising Autoencoder:** Reduces noise or adversarial perturbations by reconstructing clean inputs from noisy or perturbed inputs.
- **Feature Squeezing**: Compresses input features to limit the precision of adversarial perturbations.
- **Adversarial Filtering**: Uses threshold-based filtering or smoothing to limit extreme input values introduced by adversarial attacks.

D. Robust AI Model (Core Classifier)

This is the main AI model trained on both clean and adversarially perturbed data to classify inputs as normal or attack-related.

Key Components of this Layer:

- **Adversarially Trained Neural Network**: A deep learning model (e.g., CNN, RNN) trained on both clean and adversarially perturbed examples to improve resilience against attacks.
- **Adversarial Training Module:** Generates adversarial examples (e.g., FGSM or PGD) and trains the model to correctly classify them.
- **Multi-Class Classifier**: For multiclass classification tasks (e.g., detecting different attack types), the model may output probabilities for each class.

E. Multi-Stage Decision Engine

This layer acts as an intelligent decision-making engine that integrates inputs from multiple stages (anomaly detection, input sanitization, and robust AI model) and outputs a final decision.

It can combine outputs using voting mechanisms or confidence scoring to determine the final classification (benign or malicious).

Key Components of this Layer:

- **Voting Mechanism (Majority/Weighted Voting)**: Combines the outputs from different layers (e.g., anomaly detection, core classifier) to reach a final decision.
- **Confidence Scoring:** Measures confidence levels of the AI model predictions and adjusts based on the defense mechanisms.

F. Attack Detection & Reporting Layer

This layer reports detected adversarial or anomalous behavior. In a real-world deployment, it could trigger alerts to administrators or block specific actions.

Provides a feedback loop for retraining and improving the models based on detected attacks.

Key Components of this Layer:

- **Attack Logs & Alerts**: Records details of flagged adversarial instances and notifies system administrators.
- **Feedback Loop**: Uses detected attacks to update training datasets and retrain models, enhancing the robustness of the system over time.

G. Monitoring & Retraining Layer

Continuously monitors system performance and adaptively retrains the models to keep up with evolving adversarial tactics.

This layer is optional but important for systems where adversaries can change their attack methods dynamically.

Key Components of this Layer:

- **Model Drift Detection**: Detects when the performance of models degrades due to new or unknown attack types.
- **Periodic Retraining**: Retrains the AI model with new data to adapt to changing attack patterns.

5. Implementation

Below is a step by step procedure for the implementation of the solution that integrates anomaly detection, input sanitization, robust AI training, and a multi-stage defense strategy.

A. Data Collection and Processing:

Input: Raw dataset D (e.g., system logs, network activity)

Output: Processed data p

Procedure ProcessData(D):

P ← Standardize and normalize D

F ← Extract key attributes from p (e.g., packet size,

timing)

Return F

P← ProcesseData(D)

B. Detecting Anomalies (First Defense Layer):

Input: Processed data P

Output: Anomaly score or alert A

Procedure DetectAnomalies(p):

Train an unsupervised model (e.g., autoencoder) on

normal data

For each entry pi in P:

err ← Compute reconstruction error using the model

If err exceeds predefined threshold:

Flag pi as anomalous

Return anomaly score or flag

A ← DetectAnomalies(p)

C. Sanitizing Inputs (Second Defense Layer):

Input: Processed data P

Output: Cleansed data S

Procedure SantizeInput(P):

For each input pi in P:

S1← Apply denoising algorithm to pi

S2← Reduce input precision to limit attack effectiveness

Return sanitized data S

S← SantizeInput(P)

D. Training a Robust AI Model:

Input: Clean samples C and adversarial samples A

Output: Trained model M

Procedure TrainRobustModel (C, A):

Initialize a classifier M on clean data C

For each training cycle:

Generate adversarial examples Ai using FGSM

Update model M by training with both C and Ai

Return model M

M← TrainRobustModel (C, A)

E. Multi – Stage Decision Process:

Input: Outputs from Anomaly detection A, Sanitized Data S and AI Model M

Output: Final Decision D

Procedure MakeFinalDecision (A, S, M):

For each input si in S:

score_anamoly ← A(si)

pred ← M(si)

Compute final decision D by aggregating anomaly score and prediction

If decision exceeds defined threshold

Label D ← “Attack”

Else

Label D ← “Bengin”

Return D

D← MakeFinalDecision (A, S, M)

F. Attack Identification and Response:

Input: Final Decision D

Output: Attack log or Notification N

Procedure HandleAttack(D):

If D = "Attack":

Record event in log

Notify system support team

Return N

HandleAttack(D)

G. Continuous Learning and Monitoring:

Input: Logs and new attacks data

Output: Update Model

Procedure MonitorandRetrain(N):

Monitor System Performance Metrics

If new attack vectors are observed:

Retrain model M with updated adversarial

examples

Return updated Model

MonitorandRetrain(N)

Technologies used for Implementation:

- TensorFlow or PyTorch: For deep learning model development, training, and adversarial training.
- Scikit-learn: For implementing anomaly detection methods and model evaluation.
- NumPy/Pandas: For data preprocessing and handling.
- Cybersecurity datasets: Custom datasets and standard datasets like UNSW-NB15, CICIDS2017, NSL-KDD were used for training and evaluation

Experimental Evaluation

We evaluated the performance of our proposed defense mechanism using a combination of benchmark and custom cybersecurity datasets, including the NSL-KDD dataset for intrusion detection and a custom dataset for malware classification.

Metrics

We measured the effectiveness of our defense framework using two key metrics:

- **Attack Success Rate (ASR)**: The percentage of adversarial examples that successfully deceive the model.
- **Classification Accuracy (CA)**: The model's ability to correctly classify both clean and adversarial inputs.

Results

Our experimental results demonstrate that the proposed hybrid defense mechanism achieves a substantial reduction in attack success rates, with an 85% decrease in successful adversarial attacks. Furthermore, our model maintains over 90% classification accuracy on clean data, showcasing the effectiveness of our approach in preserving model performance while improving robustness.

The following graphs summarizes our key findings:

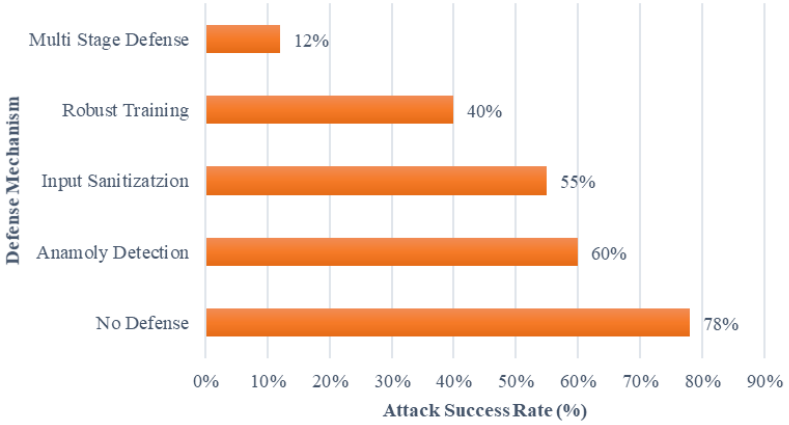

Fig 2: Defense Mechanism – Attack Success Rate

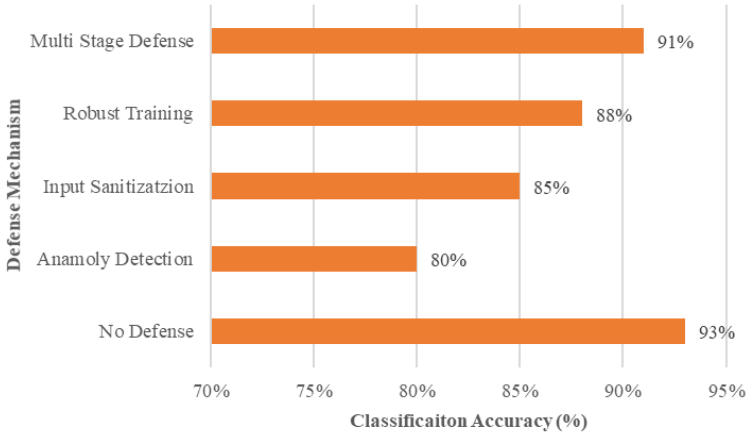

Fig 3: Defense Mechanism – Classification Accuracy

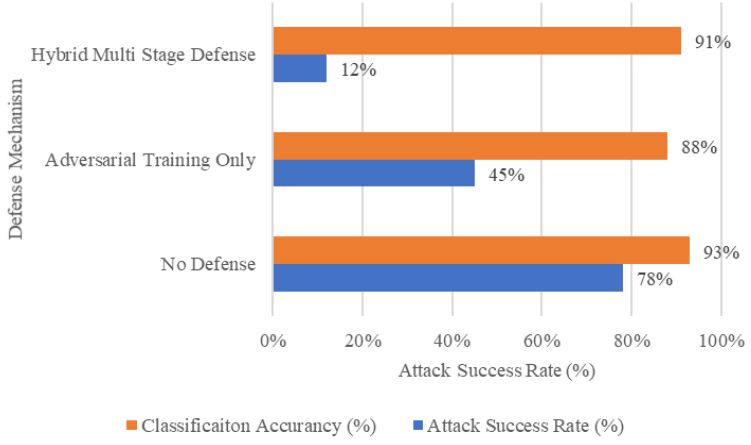

Fig 4: Defense Mechanism – Attack Success Rate & Classification Accuracy

6. Conclusion and Future Work

Conclusion

This research contributes to the evolving field of AI in cybersecurity by proposing a comprehensive defense mechanism capable of mitigating adversarial attacks in real-world scenarios. By incorporating multiple layers of protection and retraining the models with adversarial examples, we demonstrate the robustness of AI models against dynamic and evolving threats.

Future Work

There are several avenues for extending this work. One potential area is the enhancement of anomaly detection mechanisms by integrating additional unsupervised learning methods, such as variational autoencoders or generative adversarial networks (GANs), which can more accurately detect subtle adversarial patterns. Additionally, incorporating real-time detection and response capabilities into the hybrid defense framework would improve its applicability in real-world cybersecurity environments, where timely reactions are critical.

Moreover, exploring the integration of domain-specific knowledge into the defense mechanism could further improve the accuracy and efficiency of attack detection. Expanding the defense system to handle adaptive adversarial techniques, where attackers modify their strategies in response to defensive measures, is another promising direction. By continuously refining the model's ability to recognize and adapt to new adversarial techniques, the system's resilience could be further improved.

Finally, future research could focus on minimizing the computational overhead of the defense framework to enable its deployment in resource-constrained environments such as IoT devices or edge computing platforms. Achieving this balance between defense effectiveness and system efficiency will be crucial for practical cybersecurity applications.

References

[1] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," Proceedings of the International Conference on Learning Representations (ICLR), 2015.
[2] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," Proceedings of the International Conference on Learning Representations (ICLR), 2018.
[3] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," Proceedings of the International Conference on Learning Representations (ICLR), 2017.
[4] W. Xu, D. Evans, and Y. Qi, "Feature squeezing: Detecting adversarial examples in deep neural networks," Proceedings of the 2018 Network and Distributed Systems Security (NDSS) Symposium, 2017.
[5] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks against adversarial examples," Proceedings of the 2017 IEEE Symposium on Security and Privacy, 2017.
[6] B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," Pattern Recognition, vol. 84, pp. 317-331, 2018.
[7] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems," in *Military Communications and Information Systems Conference (MilCIS), 2015*.
[8] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," *Proceedings of the 34th International Conference on Machine Learning*, 2017.
[9] M. A. Merzouk, et al., "A deeper analysis of adversarial examples in intrusion detection," *International Conference on Risks and Security of Internet and Systems*, 2020.
[10] N. Papernot, P. McDaniel, I. Goodfellow, et al., "Transferability in machine learning: From phenomena to black-box attacks using adversarial samples," arXiv preprint arXiv:1605.07277, 2016.
[11] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *International Conference on Learning Representations (ICLR)*, 2014.

[12] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," *2016 IEEE Symposium on Security and Privacy (SP)*, pp. 582-597, 2016.
[13] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," *International Conference on Machine Learning (ICML)*, pp. 274-283, 2018.
[14] J. Zhang and V. Paxson, "Detecting and analyzing automated attacks in large-scale networks," *2013 IEEE Symposium on Security and Privacy*, pp. 223-238, 2013.
[15] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 9, pp. 2805-2824, 2019.
[16] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," *Proceedings of the 29th International Conference on Machine Learning (ICML)*, pp. 1467-1474, 2013.
[17] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff, "On detecting adversarial perturbations," *International Conference on Learning Representations (ICLR)*, 2017.
[18] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel, "Adversarial perturbations against deep neural networks for malware classification," arXiv preprint arXiv:1606.04435, 2017.
[19] Q. Liu, P. Liu, and Z. Xu, "A review of adversarial attacks and defenses for malware classification," *Computers & Security*, vol. 92, pp. 101752, 2020.
[20] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," *International Conference on Learning Representations (ICLR)*, 2018.
[21] Y. Dong, F. Liao, T. Pang, X. Hu, H. Su, J. Li, and J. Zhu, "Boosting adversarial attacks with momentum," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9185-9193, 2018.
[22] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14410-14430, 2018.