




Review

A Comprehensive Review of Adversarial Attacks and Defense Strategies in Deep Neural Networks

Abdulruhman Abomakhelb ^{1,*} , Kamarularifin Abd Jalil ², Alya Geogiana Buja ², Abdulraqeb Alhammadi ^{3,*}  and Abdulmajeed M. Alenezi ⁴ 

¹ Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA (UiTM), Shah Alam 40450, Selangor, Malaysia

² Faculty of Computer and Mathematical Sciences, University Teknologi MARA (UiTM), Merlimau 77300, Melaka, Malaysia; kamarul@tmsk.uitm.edu.my (K.A.J.); geogiana@uitm.edu.my (A.G.B.)

³ Faculty of Artificial Intelligence, Universiti Teknologi Malaysia, Kuala Lumpur 54100, Malaysia

⁴ Department of Electrical Engineering, Faculty of Engineering, Islamic University of Madinah, Madinah 41411, Saudi Arabia; a.alenezi@iu.edu.sa

* Correspondence: abd.abomakhleb@gmail.com (A.A.); abdulraqeb.alhammadi@utm.my (A.A.); Tel.: +60-133951322 (Abdulraqeb Alhammadi)

Abstract: Artificial Intelligence (AI) security research is promising and highly valuable in the current decade. In particular, deep neural network (DNN) security is receiving increased attention. Although DNNs have recently emerged as a prominent tool for addressing complex challenges across various machine learning (ML) tasks and DNNs stand out as the most widely employed, as well as holding a significant share in both research and industry, DNNs exhibit vulnerabilities to adversarial attacks where slight but intentional perturbations can deceive DNNs models. Consequently, several studies have proposed that DNNs are exposed to new attacks. Given the increasing prevalence of these attacks, researchers need to explore countermeasures that mitigate the associated risks and enhance the reliability of adapting DNNs to various critical applications. As a result, DNNs have been protected against adversarial attacks using a variety of defense mechanisms. Our primary focus is DNN as a foundational technology across all ML tasks. In this work, we comprehensively survey and present the latest research on DNN security based on various ML tasks, highlighting the adversarial attacks that cause DNNs to fail and the defense strategies that protect the DNNs. We review, explore, and elucidate the operational mechanisms of prevailing adversarial attacks and defense mechanisms applicable to all ML tasks utilizing DNN. Our review presents a detailed taxonomy for attacker and defender problems, providing a comprehensive and robust review of most state-of-the-art attacks and defenses in recent years. Additionally, we thoroughly examine the most recent systematic review concerning the measures used to evaluate the success of attack or defense methods. Finally, we address current challenges and open issues in this field and future research directions.

Keywords: DNNs; adversarial attacks; defense mechanisms; AI security



Received: 7 November 2024

Revised: 24 February 2025

Accepted: 10 March 2025

Published: 15 May 2025

Citation: Abomakhelb, A.; Jalil, K.A.; Buja, A.G.; Alhammadi, A.; Alenezi, A.M. A Comprehensive Review of Adversarial Attacks and Defense Strategies in Deep Neural Networks. *Technologies* **2025**, *13*, 202. <https://doi.org/10.3390/technologies13050202>

Copyright: © 2025 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Throughout recent years, DNNs have achieved excellent success in an extensive range of exciting real-world applications, including image classification, speech recognition, and self-driving cars [1,2]. Deep learning (DL) comprises two primary states that transform input data into features and then features into the desired output. The application of architecture for interfaces between these stages is required. Different architectures, such as

DNN and AutoML [3], are used in building DL models. DNNs, commonly used in this field, are being explored more than AutoML, a promising solution that is beginning to be applied in various research areas [4,5]. However, with the proliferation of DNN-based applications, security is becoming a significant concern. Several studies have shown that adversarial examples, a type of polluted input, can easily deceive a DNN [6,7]. ML algorithms have exhibited vulnerabilities to adversarial attacks during training or testing for more than a decade [8,9]. As a result of the first attacks on linear classifiers proposed in 2004 [10,11], Biggio et al. [12,13] demonstrated for the first time that gradient-based optimization attacks can mislead nonlinear ML algorithms, including support vector machines (SVMs) and neural networks [8]. Although such vulnerabilities are not unique to learning algorithms, they have recently gained mass popularity following the demonstration by Szegedy et al. [6,14] that even DL algorithms that exhibit superhuman performance on image classification tasks suffer from these issues. DNNs can be induced to misclassify an input image, even by manipulating only a few pixels. As a result, adversarial examples have become increasingly popular [6,9,13]. As an illustration of an attack during the testing phase, consider a classification task. In adversarial attacks, subtle perturbations are typically crafted using an optimization algorithm and injected into a legitimate image, creating an adversarial example commonly referred to as an adversarial example. When processed for classification, an adversarial image often induces convolution neural networks (CNNs) to make predictions that deviate from the expected outcome, displaying high confidence in the divergent prediction. Despite being imperceptible to the human eye, these introduced adversarial perturbations have the potential to significantly misguide the decision-making of a targeted DNN model with a notable level of confidence. In the context of an automobile, this could result in misidentifying signs, such as road signs that stop at other signs, leading to severe consequences [15]. In the realm of media integrity, particularly in deepfake detection, adversarial attacks in digital environments highlight another case of how deepfake videos can deceive individuals and systems by generating hyper-realistic yet fabricated content. These manipulations pose significant challenges, including eroding public trust, spreading misinformation, and exploiting cognitive biases [16]. A similar attack can be executed for all ML tasks, such as learning policy and representation.

In 2013, researchers began investigating the vulnerabilities of DNN security by adding perturbation to images to produce misclassified events. Szegedy et al. [6] introduce pioneering research that identifies vulnerabilities in DNNs and introduces adversarial examples. In 2014, Goodfellow et al. [14], the fast gradient sign method (FGSM) was introduced as an approach to generating adversarial examples, an approach to generating adversarial examples. The field advanced further in 2017 when Carlini and Wagner developed more sophisticated attacks, known as C&W attacks [17], which exposed vulnerabilities in existing defenses and set a new standard for evaluating model robustness. In 2017, Madry et al. [18] introduced a theoretical framework for robust optimization to mitigate such attacks. However, in 2018, Athalye et al. [19] used adaptive attacks to deceive several defense strategies and opened research for evaluating defense strategies to protect DNNs. It confirms the proposed techniques' effectiveness and sets the stage for future progress in reducing the risks associated with backdoor attacks in ML systems. In 2020, the focus shifted to backdoor attacks, such as BadNet [20], which exploit training data to embed triggers that evade traditional adversarial defenses [21]. Recent advances between 2021 and 2024 have introduced novel defense mechanisms that leverage explainability and feature-level analysis. These mechanisms also use digital forensics to trace poison data to address sophisticated threats, including backdoor and clean-label attacks. These advancements reflect the field's continuous efforts to enhance neural network security. On the other hand, attacks and defenses on DNN rapidly develop depending on the DNN

life cycle scenarios, such as training or testing-phase attacks. Biggio et al. [12] proposed data poisoning in the early work on training-phase attacks. Gu et al. [20] introduced a backdoor attack that demonstrated how to embed hidden triggers in DNNs during training. In 2020, with the rise in federated learning, Bagdasaryan et al. [22] demonstrated how malicious clients could poison the global model by sending manipulated updates. On the other hand, in 2018, Madry et al. [18] proposed adversarial training. These defenses are against training-phase attacks. Szegedy et al. [6] proposed adversarial examples in the testing phase. The researchers then proposed different attacks, such as the model extraction attacks by Tramèr et al. [23] in 2016, and Membership Inference Attacks by Shokri et al. [24] in 2017. Researchers rapidly proposed defenses, even reactive and proactive, and in 2022, Shawn et al. [25] introduced digital forensics to trace back source poisoning attacks and, for the first time, proposed digital forensics to solve poisoning attacks in DNN security.

In 2022, Khamaiseh and Bagagem [15] conducted a review of security vulnerabilities and open challenges in DNNs. Although it provides a proper initial review of the security problems in image classification, it must adequately cover the security topics of adversarial DNNs in additional applications with major tasks based on DNNs (classification, policy learning, and representation learning) and defense mechanisms. We aim to meet these needs by providing a more comprehensive survey of attacks and defense mechanisms and discussing future research directions in DNN security.

Our review covers recent years in the most reputable datasets, i.e., IEEE, Science Direct, and WOS, for adversarial attack and difference mechanisms with DNN against various ML approaches and tasks, i.e., classification, regression, and policy learning. The result presented a detailed taxonomy for the life-cycle of adversarial attacks, different mechanisms and strategies, and the evaluation measures used to evaluate each strategy.

1.1. Motivation and Contribution

This paper presents an overview of the most aggressive attacks and defenses on DNNs for various ML tasks. It is important to make recent advancements in this field easily accessible to help readers quickly engage with the rapidly growing DNNs security research field. It is crucial to examine the current status and upcoming trends in DNN security and present a comprehensive analysis of future research in this area. Recently, several articles have reviewed various research studies in this field [15,26–30]. A recent study highlights the critical need for more research on DNN security tailored to ML tasks in different applications, pointing out that most adversarial attack research has focused mainly on image classification. There are several ways in which this survey differs from other surveys. The findings of this survey differed from those of previous studies in the field. It analyzes adversarial DL attacks and defense mechanisms across various ML tasks based on DNN, provides a precise formulation for both attacker and defender problems, and provides a clear overview of various strategies for attack and defense mechanisms against adversarial attacks.

The primary contributions of this study can be summarized as follows:

- We provide an extensive study of the state-of-the-art adversarial attack and defense method on all ML tasks based on DNN that leads researchers to understand the DNNs security for various tasks ML tasks, i.e., classification, regression, policy learning.
- We provide a precise formulation for both attacker and defender problems; thus, it is easy for readers with limited DNNs security knowledge to understand.
- We provide a systematic and comprehensive review of the measures used to evaluate the success of attack or defense methods. Thus, it is accessible for researchers who want to research DNN security to know the mechanisms for evaluating the success of an attack or defense of DNNs.

- We identify and discuss various open issues and potential future research directions for DNN security. This research field aims to explore new methods to secure DNN models.

Our study is motivated by the need to address the security challenges in DNNs, a critical area that remains inadequately explored and fragmented in the current AI security literature.

1.2. Paper Organization

The paper's composition is structured as follows: Section 1 clarifies the rationale behind a detailed review of adversarial threats and counteractions in the context of DNNs while summarizing the key contributions and layout of the paper. Section 2 investigates the historical development of DNNs, covering aspects such as their structural designs, training processes, various artificial neural network models, and the tactics utilized by attackers. Section 3 outlines the strategies adversaries employ, considering their level of expertise, the attack scenarios they can initiate, the range of adversarial samples that can be created, and the eventual consequences of such attacks. Section 4 examines the objectives behind attacks in various ML applications. Section 5 examines contemporary adversarial onslaughts on DNNs, organizing them into four classifications: white-box, extraction attacks, poisoning, and black-box, and delves into the respective challenges, computational techniques, and contexts. Section 6 evaluates current protective measures against adversarial exploits targeting DNNs, classifying them into proactive and reactive categories, and scrutinizes each strategy's efficacy, limitations, and associated defense techniques. Section 7 summarizes the metrics for assessing the effectiveness of adversarial tactics and their countermeasures. Section 8 encapsulates ongoing research and potential new ways to address research shortcomings. Section 9 presents the conclusions and proposes avenues for future scholars related to this topic. Figure 1 shows the organization of the review.

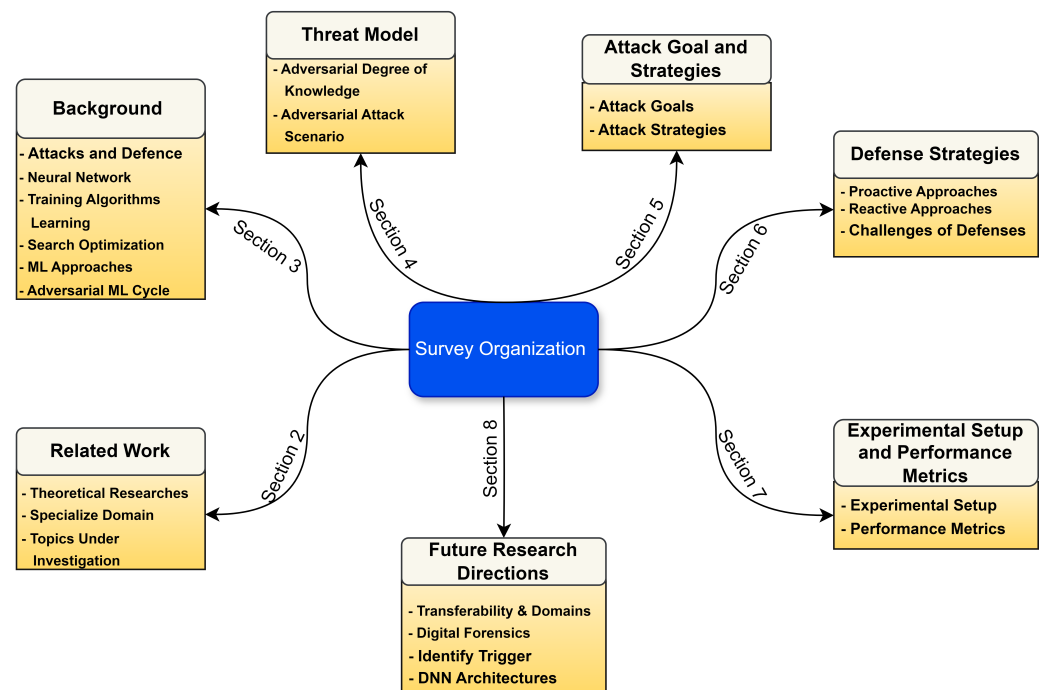


Figure 1. Organization of the review paper.

2. Related Work

A series of papers [15,27] delve into the theoretical frameworks of adversarial DL within cybersecurity. Further investigations [28,29,31,32] examine its application in particular, including computer vision, medical imagery, and NLP. Additional scholarly efforts focus on assessing ideas for attacking unsupervised and reinforcement learning (RL) methodologies. Table 1 presents a summary of existing reviews.

Table 1. Summary of existing surveys.

Title	Year	Brief Description
Khamaiseh, et al. [15]	2022	This survey offered a comprehensive overview of the mathematical concepts and terminologies related to adversarial attacks and the latest defense mechanisms in image classification.
Sadeghi, et al. [27]	2020	This article introduced a system-driven taxonomy for defining ML applications and adversarial models, enabling clear replication of experiments and facilitating advancements in robust ML systems.
Ilahi, et al. [28]	2020	The review explored the increasing threats to DRL and possible strategies to protect against them.
Mohus, et al. [29]	2023	This work provided a methodical assessment of adversarial robustness in unsupervised ML, examining several attack and defense strategies in this field.
Apostolidis, et al. [31]	2021	This work summarized the latest developments in adversarial attack and defense strategies on CNN-based classification models used in medical image processing.
Ding, et al. [32]	2020	This paper provided an improved taxonomy for adversarial attacks, classic ones and the latest ones in computer vision, and also discusses some potential and some limitations.
Zhang, et al. [33]	2020	This article offered a comprehensive overview of research works that addressed the generation of textual adversarial examples on DNNs.
Akhtar, et al. [34]	2018	This article comprehensively analyzed adversarial attacks on DNN in computer vision. It focused on the challenges and methods to ensure the robustness and security of DNNs.
Wang, et al. [35]	2022	This report summarized the latest advancements and obstacles in adversarial attacks and defenses for DL models used in image recognition applications.
Macas, et al. [36]	2023	This article thoroughly analyses adversarial attacks and defenses in DNN-based cybersecurity systems. It also addresses unresolved issues and upcoming strategies to improve the robustness and reliability of DL systems against attackers.
Mengara, et al. [37]	2024	This survey examines recent advancements and challenges in backdoor attack strategies and defenses for DNN classifiers in security-sensitive applications. It highlights the most common methods for executing a backdoor attack and evaluates the most effective detection and prevention techniques.
Bhanushali, et al. [38]	2024	The survey examines adversarial attacks and defenses aimed at Automatic Speech Recognition systems that utilize DNNs.
Costa, et al. [39]	2024	This paper reviews recent adversarial attacks and defenses related to DL, specifically focusing on object recognition tasks.

Table 1. *Cont.*

Title	Year	Brief Description
Hoang [40]	2024	The survey provided the latest advances and challenges in adversarial attacks and defenses for AI-powered functionalities in 6G networks, covering classic wireless, open radio access (O-RAN), and next-generation systems.
This survey	2025	This review stands out from current surveys in multiple ways. This survey reviewed adversarial DL attacks and defense mechanisms for different ML tasks based on DNN compared with other surveys. It provides a clear formulation for both attacker and defender problems and a comprehensive overview of attack and defense strategies.

2.1. Theoretical Researches

The research presented in [15] extensively reviewed the latest developments in adversarial attack and defense strategies, focusing on CNN-based models for general image classification. The study organizes the fundamental principles, starting with the knowledge and objectives of the attacker, the adversarial scenarios, and the different types of adversarial instances. Provides a comprehensive summary of the methods used by adversarial attacks and defense mechanisms. In addition, it discusses the challenges and applications of adversarial DL and includes the most sophisticated techniques for creating and detecting adversarial examples. In [27], the author presented a detailed classification system that clearly defines ML applications and adversarial models, enabling other researchers to duplicate experiments and advance the competition in enhancing the sophistication and resilience of ML applications. The framework begins with the Adversarial ML Cycle, delving into a detailed breakdown of the dataset's characteristics, ML architectures, the adversary's knowledge, capabilities, objectives, strategies, defensive reactions and how these elements interconnect. The paper also assesses and contrasts recent scholarly contributions, uncovering unaddressed issues and obstacles within the domain.

2.2. Specialized Domain

The research in [31] offered an in-depth analysis of adversarial tactics in the context of CNNs for medical imaging, detailing the latest methods to generate and counter adversarial examples and assessing the efficacy and limitations of defense strategies. Similarly, the researchers [31,32] provided a detailed examination of adversarial approaches targeting CNNs in computer vision, covering the foundational principles and advanced techniques for attack and defense. In natural language processing (NLP), ref. [33] explained various adversarial methods and their implications for tasks such as sentiment analysis and machine translation, addressing various attack strategies and defense solutions. Meanwhile, the researchers [34] examined the resilience of DL models in computer vision against adversarial threats, reviewing practical defense evaluations. Lastly, ref. [35] focused on the evolving landscape of adversarial attacks and defenses within image recognition, providing insights into basic concepts and emerging trends. These surveys underscore the critical importance of advancing cybersecurity measures in DL applications in various domains. In ref. [36], the survey extensively examined the current state of adversarial attacks and their corresponding defensive strategies within the realm of DL applied to cybersecurity. It explored the complexities and potential solutions necessary to fortify the robustness and ensure the security of DL-enabled systems. The study addressed a spectrum of adversarial threats, encompassing various threat models, and evaluated the latest advances in countermeasures designed to thwart such attacks. In addition, it sheds light on unresolved issues and contemplates prospective avenues for future research to enhance the resilience and reliability of DL frameworks in the face of adversarial open problems.

2.3. Topics Under Investigation

The study in [29] thoroughly reviewed the literature on adversarial robustness within unsupervised ML, examining the various attack and defense mechanisms associated with this method. The research delved into the difficulties, techniques, and practical applications of unsupervised learning when faced with adversarial conditions, touching on topics such as maintaining privacy, clustering, detecting anomalies, and learning representations. In addition, the paper introduced a framework for defining the characteristics of attacks on unsupervised learning and pointed out existing research gaps and future research avenues. In ref. [28], the article offered an in-depth analysis of deep reinforcement learning (DRL)-based strategies for smart systems, focusing on efficiency, sturdiness, and security improvements. It explored the concept of novel attacks targeting DRL and the possible defensive actions to counteract them. Moreover, it underscored the unresolved issues and research challenges in devising strategies to counteract attacks on DRL-powered intelligent systems.

3. Background

In this section, we lay the groundwork by introducing critical elements of the research landscape. Our focus begins with adversarial attack and defense methods, navigating the essential concepts. We then delve into training algorithms and learning techniques, unraveling the methodologies crucial for robust model training. The discussion broadens to encompass distinct types of ML, outlining their unique characteristics, followed by an overview of ML approaches. This succinct background is a foundation for our study's subsequent discussions and analyses.

3.1. Attacks and Defence Methods

Different mathematical concepts create adversarial ML attacks and defense methods. This subsection offers a detailed summary of essential concepts needed to understand how these work.

3.1.1. Gradient Descent

Optimization involves the process of minimizing or maximizing an objective function. In ML, optimization seeks the most suitable parameter values for an objective function, ultimately minimizing a cost function. Various algorithms can be used to optimize, with gradient descent being one of the most widely used methods to find optimal parameters in a wide variety of ML algorithms [17]. Gradient descent is a primary optimization technique that leverages the function's gradient at its current position to chart the course through the search space.

The fundamental gradient descent algorithm comprises the following steps: (1) computing the gradient ∇f of the objective function $J(\theta)$, (2) progressing in the direction opposite to the gradient ∇f , which corresponds to the steepest descent direction conducive to improvement (i.e., locating the global minimum), and (3) determining the learning rate β , which dictates the step size towards the minimum. It is crucial to fine-tune the learning rate β as it is the most significant parameter to achieve high performance in a DNN model.

A higher value of β generally facilitates faster learning for the ML model. However, this can lead to a substantial decrease in model performance, as the algorithm may overshoot the optimal minimum and end up on the other side of the valley. On the other hand, a smaller β enables the model to converge by gradually finding a local minimum after numerous iterations, which, unsurprisingly, extends the runtime. This situation highlights a trade-off between the accuracy of the result and the time needed for parameter updates.

3.1.2. Distance Metric

Distance metrics are integral to quantifying the separation between two points, typically represented as vectors. In essence, these metrics gauge the degree of similarity between two vectors, with a distance of zero indicating complete equivalence under a specific metric. To compute the distance between two vectors, one typically evaluates the norm of their difference using a norm function. Formally, the norm function L_p of the vector x can be defined as follows:

$$\|x\|_p = \left(\sum_i |x_i|^p \right)^{\frac{1}{p}}, \quad \text{for } p \in \mathbb{R}, p \geq 1 \quad (1)$$

where \mathbb{R} represents the set of real numbers.

In this context, $p \in \mathbb{R}$ means p is a real number (i.e., it can take any value from the set of real numbers).

The condition $p \geq 1$ specifies that p must be greater than or equal to 1.

The norm of a vector x quantifies its distance from the origin to the point x .

These distance metrics play a pivotal role in the process of generating adversarial attacks, enabling the quantification of their similarities. In particular, state-of-the-art adversarial attack algorithms frequently rely on L_0 , L_1 , L_2 , and L_∞ distance metrics [3,11,17,41].

3.2. Neural Network

In this subsection, we systematically explore fundamental concepts within the realm of ML. We delve deeper into DNNs and artificial neural networks (ANNs). Furthermore, this subsection navigates the intricacies of CNNs. The subsection lays the groundwork for an advanced understanding of the diverse landscape that constitutes the backbone of modern ML research by providing a concise, yet comprehensive overview of these ML paradigms.

3.2.1. ANNs

ANNs are interconnected neurons inspired by biological neural networks. The initial and foundational mathematical model for ANNs was the single-layer perceptron, pioneered by Rosenblatt [42], and later extended into multi-layer perceptrons (MLPs) by Nazzari et al. [43]. To emulate the learning mechanisms of the human brain, ANNs were developed as generalized mathematical models inspired by biological neural networks [26]. The fundamental architecture of an ANN typically encompasses a minimum of three essential layers: (1) the input layer, (2) the output layer, and (3) one or more hidden layers. Each layer comprises nodes, which serve as the fundamental units of computation. These nodes transmit input to subsequent layers, receive input from other nodes or external sources, and generate the corresponding output.

The choice of neural network type depends on the specific problem conditions, with different neural network architectures tailored to various tasks. For instance, a text generation problem might require a recurrent neural network (RNN) for optimal performance.

Figure 2 below illustrates the basic structure of an artificial neuron. Each neuron receives inputs x_1, x_2, x_3 , weighted by corresponding coefficients $\omega_{x_1}, \omega_{x_2}, \omega_{x_3}$, along with a bias term b . These inputs are combined through a summation operation, followed by an activation function g , resulting in outputs y_1 and y_2 .

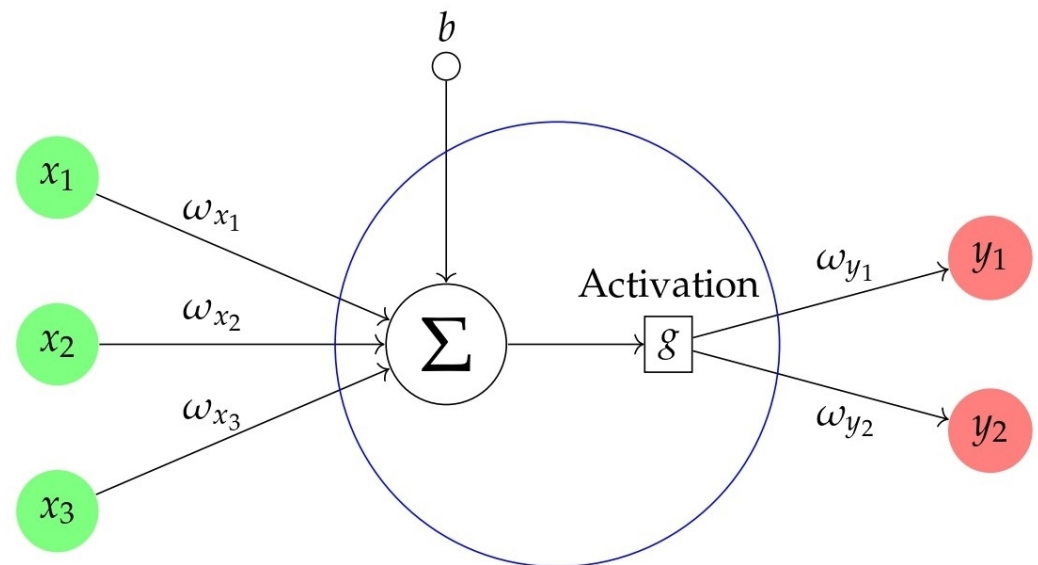


Figure 2. A representation model of ANNs.

3.2.2. DNN

A DL model, often called a DNN, comprises numerous layers with nonlinear activation functions capable of expressing intricate transformations between input data and the desired output.

Figure 3 shows a neural network architecture commonly used in DL applications. The neural network consists of three main layers: the input layer, two hidden layers, and the output layer. The nodes represent each layer, and connections (or links) between nodes signify the flow of information during the network's computation. The input layer, labeled "Input Layer", receives input signals denoted as x values. The two hidden layers, labeled "Hidden Layer 1" and "Hidden Layer 2", contain nodes that represent intermediate features learned by the network. The output layer, labeled "Output Layer", produces the final output represented as \hat{y} values. The weights associated with the connections between layers are adjusted during training to enable the network to learn and make predictions. This diagram is a visual representation of the neural network's structure, which helps to understand its architecture for tasks such as classification or regression.

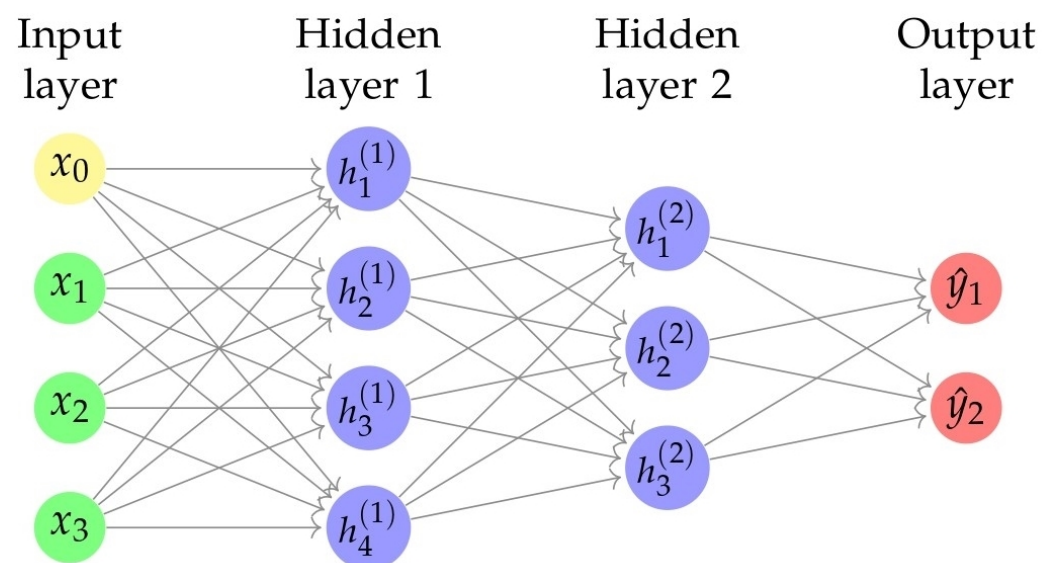


Figure 3. DNN with multiple hidden layers.

3.2.3. Convolutional Neural Network (CNN)

In 1998, LeCun introduced deep CNN [44], which has become one of the most widely used models in DL. This model automatically learns features from input data, eliminating the need for manual feature extraction. It extensively uses image classification tasks, including object identification, detection, and classification.

CNN operates under supervised learning, which requires a substantial volume of labeled data for training. Its inspiration comes from the visual cortex of animals. The convolutional layer employs a set of filters applied to the input image, which generates feature maps. These feature maps result from the convolution of local patches with weight vectors, often called filters. Feature maps are collections of localized weighted sums [45]. To enhance training efficiency, filters are repeatedly applied, reducing the number of parameters during the learning process.

Pooling layers are used to subsample non-overlapping regions in feature maps, typically using maximum or average pooling, to handle more complex features [45]. Finally, fully connected layers function as a conventional neural network, mapping the features to the predicted outputs in the final phase of learning. Deep convolutional neural networks typically comprise convolutional layers, subsampling layers, dense layers, and a softmax layer at the end [45].

Numerous CNN-based models have been offered and implemented by various researchers. Major CNN architectures, these models, including GoogleNet, VGGNet, and ResNet, are widely regarded for their exceptional performance on various image classification and object recognition benchmarks.

CNN often performs image classification and video object detection tasks. It is excellent at capturing local spatial information through its convolutional filters. On the other hand, network traffic data does not inherently possess local spatial relationships, making CNN more suited for use in hybrid systems in conjunction with RNN [46,47].

3.2.4. RNNs

RNNs are a category of DL models distinguished by their internal memory, which allows them to grasp sequential dependencies. In contrast to CNN, which treats inputs as isolated entities, RNNs consider the temporal sequence of inputs. This characteristic makes them ideal for tasks that involve sequential data processing [48]. Through a recurrent loop mechanism, RNNs apply a consistent operation to each element involving sequential data processing within a sequence; the present computation depends not only on the current input but also on the outcomes of previous computations [49].

RNNs are crucial to modeling context dependencies effectively and are, therefore, vital for a wide variety of sequence-based applications, including NLP, video classification, and speech recognition. In language modeling, for instance, the accurate prediction of the next word in a sentence needs a comprehensive knowledge of the preceding words. The recurrent structure of RNNs enables them to model these sequential relationships efficiently, allowing more robust representations of temporal and contextual dependencies [50,51].

However, basic RNNs need help with short-term memory limitations, which restrict their ability to maintain information in long sequences [52]. To tackle this limitation, researchers have developed more advanced variants of RNN, including LSTM [53], bidirectional LSTM [54], Bayesian RNN [55], GRU [56], and bidirectional GRU [57].

3.3. Training Algorithms Learning

The training algorithm aims to find network parameters that minimize the difference between the network predictions on training inputs and the truth labels.

3.3.1. Backpropagation

In backpropagation, input values are fed forward through the ANN, and errors are calculated and propagated back. Backpropagation minimizes the cost function as the ANN's weights and biases are iteratively updated and adjusted. The model calculates an update of the parameter taking the gradient of the loss function with respect to its weights. If the network minimizes the cost function, the network will propagate [58].

3.3.2. Self-Organization Map

Self-organization maps (SOMs) are unsupervised learning algorithms that construct spatial representations of multidimensional data on typically two-dimensional grids. This method preserves the topological and metric relationships between data points by organizing them on a grid. Using the SOM algorithm, similar data samples are mapped to proximate units on the grid using a neighborhood function. The network weights are adjusted iteratively to reveal the underlying structure of the data in response to input vectors. In this competitive learning process, the algorithm clusters similar data points while keeping dissimilar ones apart until the map effectively represents the input space [59].

3.3.3. Autoencoder

An autoencoder's architecture encodes input data into a compressed representation in a lower-dimensional space and then reconstructs the input data from this representation as accurately as possible. Training an autoencoder is unsupervised, meaning it does not require labeled data. The network trains to minimize the reconstruction error, representing the difference between the original input and its reconstruction. This process involves backpropagation to adjust the network weights, similar to other neural networks [60].

3.4. Search Optimization Methods

Optimization methods are critical to solving complex problems in various disciplines. There are several optimization methods, each with its strengths and suitable application scenarios. We will delve into some of these methods, including iterative, heuristic, greedy, and metaheuristic methods.

3.4.1. Iterative Methods

Iterative methods are algorithms that converge to a solution by repeatedly approximating. Iterative methods often start with an initial guess of the solution and progressively improve it by applying an iterative process. This approach is common in numerical analysis and can be used for various optimization problems. The key idea is that each iteration brings the solution closer to the optimum, and researchers continue the process until they achieve a desired level of accuracy or the improvements become negligible [61].

3.4.2. Heuristic Methods

Heuristics are problem-solving methods that employ a practical approach to finding satisfactory solutions where optimal solutions are impractical. They are more comprehensive and focus on speed and efficiency. Heuristic methods are typically used when researchers face a large and complex search space that makes an exhaustive search impractical. They provide suitable solutions that may not necessarily be the best possible but are obtained relatively quickly [62].

3.4.3. Greedy Methods

Greedy algorithms follow the problem-solving heuristic of making the locally optimal choice at each stage. It assumes that one can reach a global optimum by choosing a local optimum at each step. These algorithms are known for simplicity and are used to solve op-

timization problems where the result involves making a series of choices. However, greedy methods may only sometimes produce the optimal solution for all problems, especially when a globally optimal solution cannot be built from optimal local solutions [62].

3.4.4. Metaheuristic Methods

Metaheuristics are high-level frameworks or strategies that guide lower-level heuristics in searching for solutions to optimization problems. They escape local optima and robustly search the solution space. Researchers often apply nature-inspired meta-heuristic methods to a wide range of problems. Examples include simulated annealing, particle swarm optimization, and genetic algorithms, to name a few. They are handy for complex problems where other methods fail to find satisfactory solutions in a reasonable time frame. Each of these methods has its place in the optimization algorithms. The selection of a method typically hinges on the problem's specifics, the required precision, the complexity of the solution space, and the available computational resources [63].

3.5. ML Approaches

In this subsection, we delve into foundational concepts encompassing common ML approaches, specifically focusing on three prominent paradigms: supervised, unsupervised, and RL.

3.5.1. Supervised

It is the most prevalent form of ML, deep or not. Consider a scenario in which we aim to construct a system capable of classifying images such as houses, cars, or pets. Initially, we assemble a large dataset containing images of houses, cars, and pets, each tagged with its respective category. During the training phase, the machine is presented with an image and produces an output as a vector of scores, one score per category. Supervised learning predominantly utilizes labeled data, whether it involves deep or non-deep methods. Consider a scenario in which the goal is to develop a system for dividing images into categories such as houses, cars, people, or pets. This setup curates a comprehensive dataset, with each image labeled with its corresponding category. The machine processes an image throughout the training and generates an output vector with scores assigned to each category. The primary objective is to ensure that the correct category receives the highest score, a goal that is usually achieved through training [26].

To compute an objective function to measure machine performance, quantifying the error or distance between the output scores and the desired score pattern. Subsequently, the machine adjusts its internal tunable parameters, often called 'weights'. The weights are real-valued parameters that function as adjustable controls that determine how the machine processes inputs to produce the corresponding outputs. In a typical DL system, hundreds of millions of tunable parameters and an equivalent number of labeled data points are used to train the model [26].

3.5.2. Unsupervised

Unsupervised ML differs from supervised ML, primarily in the absence of a training set. Secondly, although most clustering algorithms include an optimal criterion, they do not guarantee that the global optimal solution has been reached. Typically, researchers must consider all data partitions, but even moderate sample sizes make this impractical, so they use some heuristic approach instead. In this case, using a different starting parameter where possible would be a good idea [26].

3.5.3. RL

RL is a learning method that analyzes the interactions of the agent with their environment to determine the best strategies. RL agents learn differently from supervised and unsupervised learning processes because they do not require training data. Instead, they learn from observations made during real-time interactions with the environment. Robotics and control problems and many other real-world challenges can be solved through RL using a trial-and-error approach. One of the main drawbacks of RL algorithms in practical applications is their slow learning speed and difficulty learning in complicated settings.

RL agent acquires knowledge by interacting with its dynamic environment, refining its behavior through trial and error [64,65]. The agent receives a single reward signal that reflects the evaluation of its performance in the given scenario. This type of feedback offers less detailed guidance than supervised learning, where the agent is explicitly given the correct actions to perform [66]. In contrast, it provides more guidance than unsupervised learning, where the agent can independently determine the correct actions without receiving explicit performance feedback [67].

3.6. Adversarial ML Cycle

Research on the security aspects of ML, focusing on adversarial knowledge, was initially introduced in [68]. Subsequently, researchers proposed a holistic approach to incorporate both the goal and capabilities of adversaries into the model, as documented in [69]. More recently, Biggio et al. argued for the construction of a well-defined adversarial model, which encompasses four critical dimensions: goal, knowledge, capability, and attacking strategy [70].

To elaborate, the adversarial goal can be explicitly delineated by considering the expected impacts and the specificity of the attack on security threats. For example, an attacker's goal might involve launching a non-selective integrity attack aimed at causing high false positive and true negative rates in classifiers. Alternatively, it could revolve around the execution of a targeted privacy violation attack to acquire sensitive data about a specific user illicitly.

Adversarial knowledge can be categorized into two groups: depending on whether the attacker has access to training data, features, learning algorithms, decision-making processes, classifier settings, or feedback information, attacks can be classified as either partial or full knowledge scenarios. The attacker's ability to launch adversarial attacks depends on how much influence they have over the training and testing datasets. Furthermore, this can qualitatively evaluate this capability from three angles: (1) whether the security threats have a causative or exploratory impact, (2) the proportion of training and testing data under the attacker's control, and (3) the extent to which features and parameters are known to the attacker.

Lastly, the attacking strategy encompasses an adversary's specific actions to effectively manipulate training and testing data to achieve their objectives. It may include decisions about data manipulation, category label modifications, and feature tampering.

Figure 4 illustrates the life cycle of adversarial attacks on DL systems. The diagram is divided into two main components: the "Attack System" on the left and the "Target System" on the right. In the Attack System, the adversary employs an array of strategies denoted by "Knowledge" and "Strategy" to develop attacks. These attacks target the system, which consists of a 'DL App' (DL application). The life cycle involves a continuous interplay between the Attack and Target Systems, with arrows indicating the flow of attacks and corresponding defenses. Additionally, Figure 4 introduces "Defense Detectors" in both the Attack and Target Systems, emphasizing the dynamic nature of adversarial interactions. The overarching theme is the cyclical adaptation of attack and defense strategies, symbolizing

the perpetual evolution of adversarial techniques and the need for responsive defense mechanisms in DL applications.

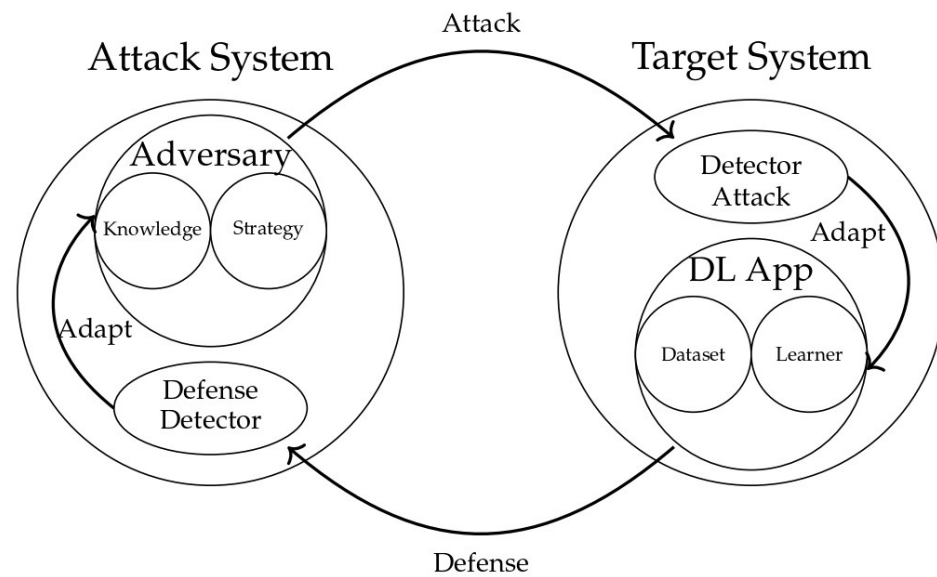


Figure 4. Adversarial DL cycle.

4. Threat Model

A threat is a prospective factor that could lead to the compromise of assets and organizations. Illustrative categories of threats include impersonation of user identity, data manipulation, repudiation, exposure to unauthorized information, service disruption, and unauthorized privilege escalation [71].

In adversarial attacks on classification tasks, the threat model, described in [72], involves a malicious attacker attempting to deceive a DNN-based image classifier by introducing adversarial examples. The attacker's objective is to perturb the pixels of an image repeatedly until the DNN model causes a misclassification event, leading to an incorrectly classified image on another. Assume that the attacker has full access to the model's internal details of the DNN-based image classifier, essentially treating it as a white box.

Consequently, the attacker can generate perturbation data that causes the classifier to misclassify. This is in part due to the widespread familiarity with the neural network architectures used in top-performing image classifiers, such as ResNet models [73]. Even if the attacker lacks knowledge of the classifier parameters, they can develop a reliable surrogate model of the classifier by sending and collecting responses from queries [74].

The common case is where the adversarial does not know the defense system, treating the detector as a black box. Nevertheless, we can also investigate a rare scenario in which the intruder obtains the defense system and its gradients, conceivably due to corrupt employees or bugs in the server. In this case, evaluating the DNNs model in classifier against white-box attack [75].

The threat model works in real-world scenarios in the following way: adversaries typically lack access to the model's internal details, usually well protected by service providers. Furthermore, researchers have emphasized the importance of assessing the robustness of the model through the transferability of adversarial examples [26,76]. Otherwise, attackers could exploit a more vulnerable model as a substitute to breach the defenses of the target model.

In some situations, the model's internal details and defensive mechanisms may be inadvertently disclosed to the attacker, leading to a grey-box scenario. However, the adversary must be aware of the defense's specific parameters. In the extreme white-box

scenario, the adversary knows the oracle model and its defense mechanisms. This is an exceedingly potent position for the attacker, as attacks launched under these conditions are highly challenging to defend against given the attacker's comprehensive awareness of the system.

Figure 5 demonstrates a classification that presents different types of adversarial attacks against DNNs. The attacks are grouped into top-level branches in the taxonomy: white-box, black-box, data poisoning, and extraction attacks. This classification system provides an organized and thorough summary of different types of attacks, helping researchers and professionals understand the range of dangers in DNN security.

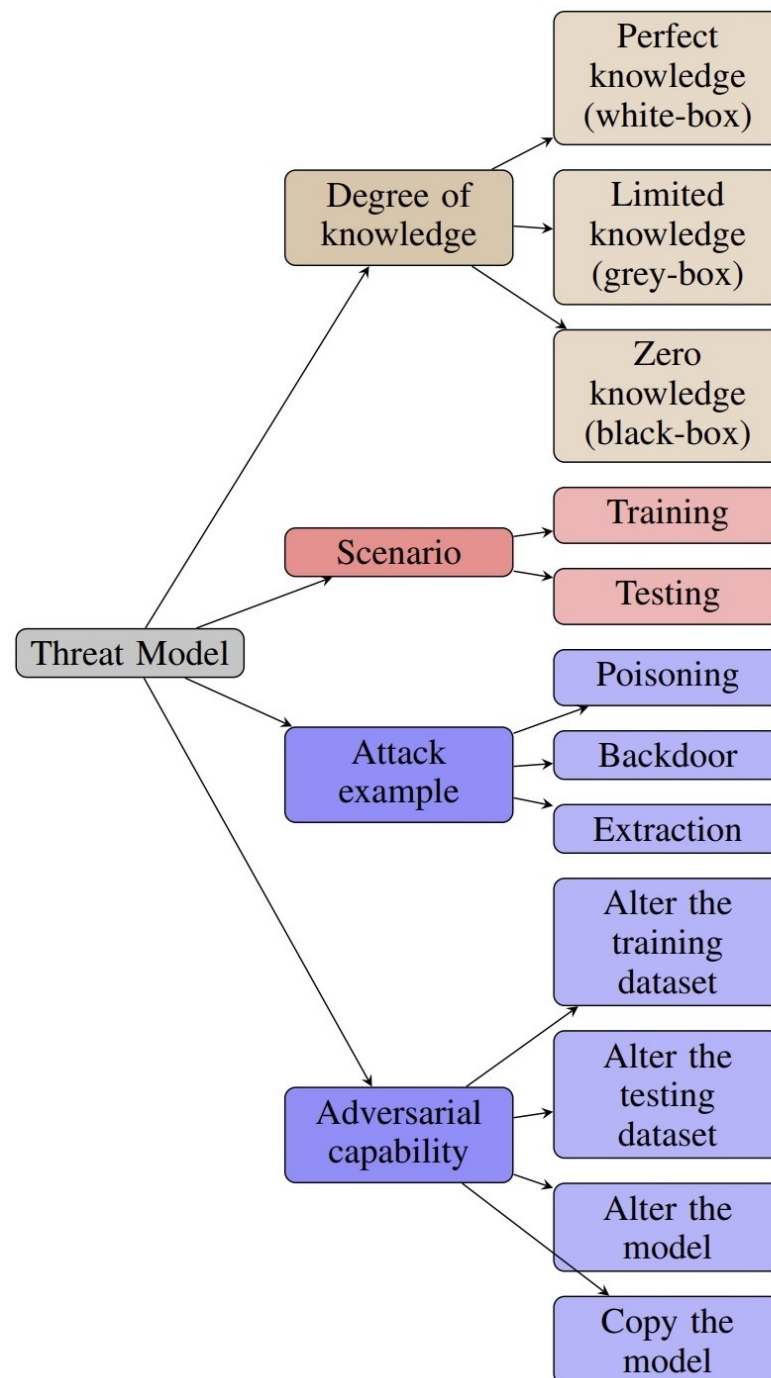


Figure 5. Taxonomy of the adversarial threat model.

4.1. Adversarial Degree of Knowledge

Knowledge is a fundamental aspect of any scientific endeavor. It plays a pivotal role in identifying vulnerabilities, developing defenses, and advancing the field of DNN security. Understanding whether an attacker can access a model's internal details, such as its parameters or architecture, is crucial in assessing and mitigating potential threats.

4.1.1. White Box

The white-box threat model assumes a significantly more potent attacker than the black-box scenario. In this setting, the attacker has extensive access to information, including input/output data and the target model's internal details. Adversarial attacks in this context involve methodically crafting attacks by leveraging the victim model's internal details. Due to unrestricted access to model information, white-box attacks pose more significant challenges for defense. In the context of our literature review, the Carlini and Wagner (C&W) attack [17] and the projected gradient descent attack (PGD) [73] emerge as particularly influential. In the white-box attack threat model, similar to bit-flip-based adversarial weight attacks on quantized networks [77–79], we consider a scenario where the attacker has access to the model's weights, gradients, and a portion of the test data. This threat model remains valid, as previous research has shown that attackers can effectively acquire comparable information, such as layer numbers, weight sizes, and parameters, through side-channel attacks [80–83]. However, it is essential to note that the attacker denied access to training-related information, including the training dataset and hyperparameters.

4.1.2. Black Box

In this threat model, the attacker lacks the internal details of the target model. However, the attacker can train a readily exploitable model to craft adversarial examples and then transfer these examples to the target classifier, also known as the oracle. The attacker possesses a training dataset with a distribution similar to that used to train the oracle. Using the same training dataset, the substitute and the oracle emulate a worst-case scenario that could occur in real-world situations. However, the attacker needs to be more informed about the defensive mechanisms and the precise architecture and parameters of the oracle.

In this threat scenario, the adversary can submit queries to the target models and analyze the resulting predictions based on the provided input. In other words, the attacker can see the input and output of the target model but does not have access to the internal components of the victim model, including its architecture, gradients, or hyperparameters. The HopSkipJump (HSJ) attack [84] is one of the latest and most potent attacks within the black-box attack category. The HSJ attack is a repetitive method that depends exclusively on the outputs of the target model, without needing any gradient data from the model itself. It approximates the model's gradients and determines the direction of these gradients using binary data from the decision boundary, which makes it particularly efficient in a black-box environment.

4.1.3. Grey Box

In this scenario, the attacker has access to the internal details of the model. An attacker may sometimes leak a model's parameters, architecture, and defense mechanism. However, The adversary is unaware of the defense parameters. As a result, the attacker can craft adversarial examples based on the oracle itself without needing a substitute model. However, it is essential to note that while the attacker has insight into the model's internal details, the specific parameters of the defensive mechanism remain undisclosed. Therefore, the effectiveness of the defense may still be preserved [31].

4.2. Adversarial Attack Scenarios

4.2.1. Attacks During Training Phase

An attacker can undermine a particular ML system by focusing on its training dataset during training. This strategy, referred to as a poisoning attack, involves the attacker trying to interfere with the training dataset, which is used to alter the statistical characteristics of the data for training. It can use the following scenarios to initiate this poisoning attack:

- **Data injection and modification:** The attacker intentionally incorporates malicious samples into the training dataset or alters existing data points in a harmful way. These detrimental examples may be produced using the label noise method [85]. Once the training dataset is compromised, the resulting DNN model produces erroneous outputs [86].
- **Logical corruption:** The adversary disrupts the learning process of the DNN model, hindering its ability to learn accurately.

4.2.2. Attacks During Testing Phase

In certain circumstances, an opponent may exploit the features of fundamental classes that can be altered without compromising accurate classification [87]. During the testing phase, researchers can execute two primary attacking scenarios:

- **Evasion attack:** The adversary attempts to infiltrate the specific model by creating a harmful input sample that the ML model incorrectly categorizes. Most of the proposed adversarial works have incorporated this type of attack [88].
- **Exploratory attack:** During testing, the adversary gathers data about the learning system to gain insight into its model's internal details or training data by creating adversarial samples, similar to side-channel attacks.

In order to train the attack model, the attacker is assumed to possess a supplementary dataset that includes both authentic and counterfeit images. Despite the availability of many pre-made DeepFake image datasets and models, the attacker's access to data is restricted to prevent worst-case scenarios:

- **Limited Dataset Size:** the attacker needs more resources to gather public information, resulting in a relatively small dataset size.
- **Out-of-Distribution DeepFake:** the attacker can only gather fake images produced by certain DeepFake techniques, so not all types of DeepFakes will be in the auxiliary dataset.

5. Attack Goal and Strategies

This section systematically explores the attack goal and strategy for DNNs used in various ML tasks. In this section, we will discuss the attacker's goal, either privacy, integrity, or availability, and describe the strategy the attacker follows to attack DNNs' models.

5.1. Attack Goal

Integrity attacks aim to introduce new features, alter existing ones, or bypass current features. Privacy attacks aim to extract private information from the ML model. Attackers carry out availability attacks to disrupt the intended operation of the system. Depending on the ML task (i.e., learning or classification), adversarial goals are shaped as follows:

5.1.1. Classification Task

- **Untargeted Misclassification:** The adversary aims to raise the DNN model's misclassification rate by feeding it adversarial examples from an untargeted adversarial attack

that results in inaccurate classification. Simply put, the attacker attempts to make the model classify adversarial examples incorrectly.

- **Source/Target Misclassification:** Engaging in targeted adversarial attacks can achieve this objective. By adding a trigger to a specified target label, the victim DNN model system will be misled in classifying the correct label, called a target attack.

5.1.2. Confidence Reduction

The adversary attempts to reduce confidence in the prediction of the DNN model by increasing the ambiguity of the prediction of the target model, even if the predicted class remains correct. This attack manipulates inputs to decrease the model's certainty about its output and reduce performance in practical applications such as autonomous driving or medical diagnostics, where high-confidence predictions are critical. Several researchers have explored techniques related to confidence reduction and its connection with adversarial examples. For instance, adversarial attacks like the FGSM [14] or the DeepFool attack [41] work by introducing perturbations that minimize the margin of decision boundaries, lowering the confidence of the model's outputs.

5.1.3. Policy Learning Task

An adversary's goal is to change an agent's behavior to solve a targeted problem.

The adversary's goal is to determine the details about the model, the reward function, or other parts of DRL. Adversaries can use these details to create copies of the model or to attack it. There may be a limitation in the ability of the adversary to change the environment based on the part of the environment they can modify.

5.1.4. Representation Learning Task

The representation learning task disrupts the latent representation. A model disruption attack introduces new samples into the training set, a manipulation attack changes an existing sample, and a logic corruption attack adds a backdoor to the model. Additionally, representation learning faces a broader threat from adversarial examples, perturbed inputs crafted to mislead models during their task-learning process.

5.1.5. Generate Synthetic Data Task

A generate synthetic data task attack uses adversarial methods to attack models that use synthetic data for training or testing purposes. ML researchers often generate synthetic data to augment datasets, protect privacy, or compensate for unavailable real-world data. Adversaries may generate synthetic data to embed adversarial patterns in the data or target models trained on synthetic data, ultimately weakening their performance [89].

5.2. Attack Strategies

This part examines the latest adversarial attacks on DNN in various ML tasks. Attack strategies can be divided into four types: white-box attack, black-box attack, poisoning attack, and extraction attack, as illustrated in Figure 6. Our main focus is to provide a detailed explanation of the attack scenario and how the attack problem is formulated, an overview of the different algorithms used by attackers to solve the attack problem, and finally, explain how this attack fits different ML tasks.

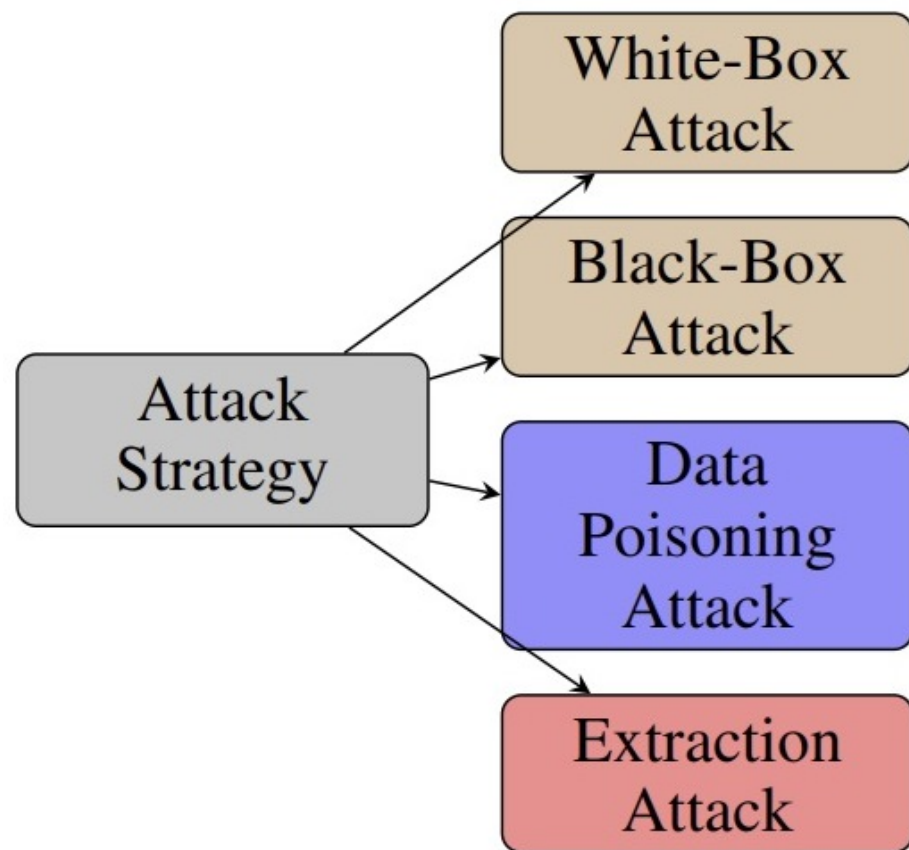


Figure 6. Taxonomy of attack strategies.

5.2.1. White-Box Attack

Attackers can access internal details for target models, training data, and loss functions in white-box attacks. Szegedy et al. [6] investigated and demonstrated DNNs that are susceptible to adversarial attacks. An adversarial example can be instituted by adding small perturbations to an image. This can deceive DNNs, reducing their accuracy. Adversarial examples are also known as white-box adversarial attacks against DNNs. These disruptions are identified by optimizing the input to maximize the prediction error. The attack is performed based on the knowledge of the ML input and output layer values only to solve the optimization problem formulated by Szegedy et al. [6]. The attack can be applied to classification and policy learning tasks [82]. However, it cannot be applied to unsupervised learning tasks as no testing phase is required. For classification, it tries to change the output label, for instance, slightly altering a benign traffic sign image to fool recognition traffic signs into misclassifying a self-driving vehicle's sign to another, such as a "Stop" sign as a "Yield" sign, as shown in Figure 7. Policy learning tries to change the best-learned action.

Researchers have proposed different algorithms to solve attack problems, using various optimization techniques based on similarity measures between original and adversarial examples. In some cases, adaptation to the main problem is proposed by [6].

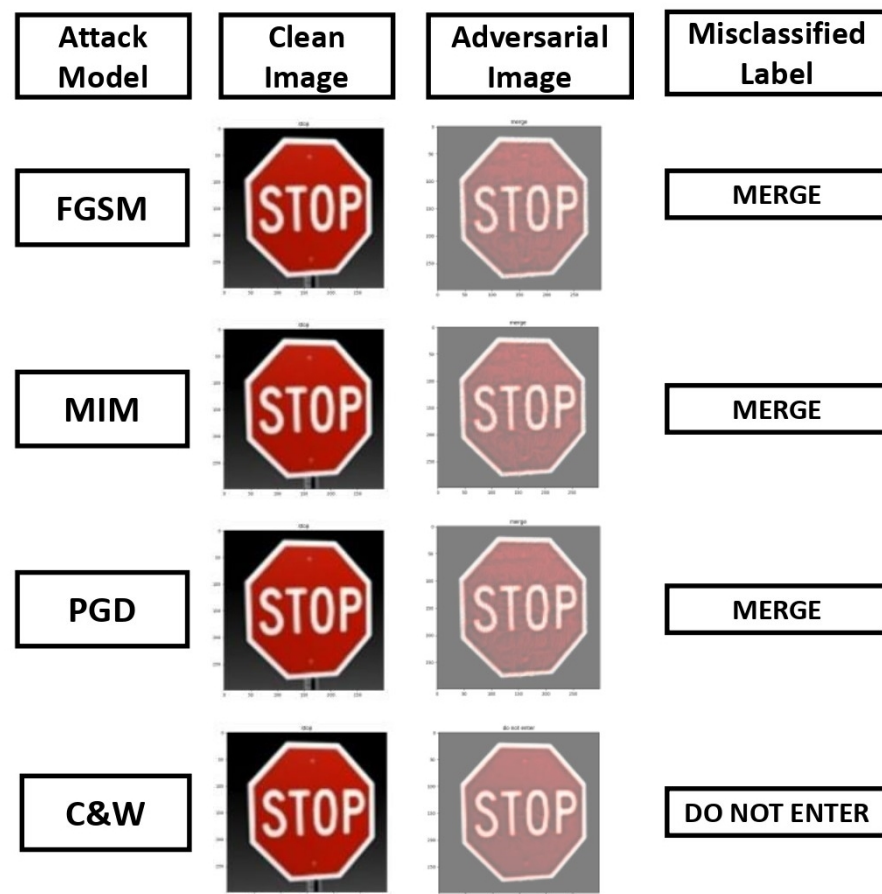


Figure 7. The impact of various attack models and image transformations on a sample traffic sign image, specifically a stop sign [90].

- **Elastic-Net Attacks to DNNs (EADs):** The research introduces EAD regularization, which combines the penalty functions L1 and L2, to the optimization problem of finding the minimal perturbation in an image that causes a DNN to misclassify it as a target class [91]. It shows that EAD can generalize and craft L1-oriented, more effective adversarial examples focusing on targeted attacks. It demonstrates that EAD can achieve similar or better attack performance than C&W and other breaking undefended and defensively distilled DNNs. EAD creates adversarial samples employing the iterative shrinkage-thresholding algorithm [92].
- **Targeted Universal Adversarial Perturbations:** This research proposed a simple iterative method to generate universal adversarial perturbations (UAPs) for image classification tasks. By integrating the straightforward iterative approach for producing non-targeted universal adversarial perturbations with the fast gradient sign method to create targeted adversarial perturbations for a given input, UAPs are imperceptible perturbations that can cause DNN to misclassify most input images into a specific target class. The results demonstrate that the technique can produce almost imperceptible UAPs that achieve high ASR for known and unknown images [93].
- **Brendel and Bethge Attack:** Divide the image into regions based on estimated boundaries and use L_p for the similarity distance and the iterative algorithm to enhance the perturbation value additively; the attack calculates the best step by solving a quadratic trust-region optimization problem in each iteration. This attack improves the performance of generating adversarial examples [94].
- **Shadow Attack:** The attack imposes more constraints on the standard problem, which shows an effective success rate for a set of datasets. By applying a range of penal-

ties, shadow attacks for constraints promote the imperceptibility of the resulting perturbation to the human eye by limiting the perturbation size [95].

- **White-Box Bit-Flip-Based Weight Attack:** The authors propose an attack paradigm that modifies the weight bits of a DNN in the deployment stage [96]. They suggested a general formula with elements to meet efficiency and covert objectives and a limitation on bit-flip quantity. They also presented two cases of the general formulation with different perturbation purposes: a single-sample attack aims to misclassify a specific sample into a target class by flipping a few bits in the memory, and a triggered samples attack aims to misclassify the samples embedded with a particular trigger by flipping a few bits and learning the trigger. They utilized the alternating-direction method of the multipliers method to solve the optimization problems and demonstrated the superiority of their methods in attacking DNNs.
- **Adversarial Distribution Searching-Driven Attack (ADSAttack):** Wang et al. [97] proposed the ADSAttack; this technique generates adversarial examples by exploring adversarial distributions in unsupervised hidden learning. The technique trains an autoencoder to capture the underlying features of the input data, then identifies the distribution that minimizes the classification error of the target DNN and ultimately generates adversarial instances based on this distribution using the decoder. The algorithm can produce adversarial examples that are more diverse and effective than the existing ones and can be transferred to other models and defense mechanisms. The algorithm may also have potential applications in the security, privacy, and robustness of DNNs.
- **Targeted Bit-Flip Adversarial Weight Attack (T-BFA):** T-BFA is a novel method for attacking DNNs by altering a small number of weight bits stored in computer memory. The paper proposed three types of T-BFA attacks: N-to-1 attack, which forces all inputs from N source classes to one target class; 1-to-1 attack, which misclassifies inputs from one source class to one target class; and 1-to-1 stealth attack, which achieves the same objective as 1-to-1 attack while preserving the accuracy of other classes. The paper also shows the practical feasibility of T-BFA attacks in a real computer system by exploiting existing memory fault injection techniques, such as the row-hammer attack, to flip the identified vulnerable weight bits in dynamic random access memory (DRAM). The paper discusses the limitations and challenges of T-BFA attacks, such as the dependency on the bit-flip profile, the trade-off between attack stealthiness and strength, and the resistance of some defense techniques, such as weight quantization, binarization, and clustering [98].
- **Invisible Adversarial Attack:** Wang et al. [99] introduced the invisible adversarial attack, an adaptive method that uses the human vision system (HVS) to create realistic adversarial scenarios. The technique comprises two categories of adaptive attacks: coarse-grained and fine-grained. The first attack introduces a spatial constraint on the perturbations, which only adds perturbations to the cluttered regions in an image in areas where the human visual system has reduced sensitivity. The fine-grained attack uses a novel metric called just noticeable distortion, which measures each pixel's noticeable distortion (JND) to better simulate the HVS perceptual redundancy and set pixel-wise penalty policies for the perturbations. The algorithm can be applied to adversarial attack methods that manipulate pixel values, like FGSM [14], Basic Iterative Method (BIM) [100], and C&W [17].
- **Type I Adversarial Attack:** Tang et al. [101] introduced a new method to create Type I adversarial examples, where the appearance of an input image is significantly changed while maintaining the original classification predictions. Utilizing an existing understanding of the Gaussian distribution, incorporating label data into the latent

space allows identifying features. In this latent space, they challenge the classifier by modifying the latent variables through a gradient descent approach. This approach utilizes a decoder to propagate forward, converting the revised latent variables into images. Additionally, a discriminator was incorporated to assess the distribution of the manifold within the latent space, facilitating the successful execution of a Type I attack. The method has successfully produced adversarial images that resemble the target images but are still classified as the original labels.

- **Generative Adversarial Attack (GAA):** He et al. [102] proposed the GAA, a novel Type I attack method that uses a generative adversarial network (GAN) to exploit the distribution mapping from the source domain of multiple classes to the target domain of a single class. The GAA algorithm operates under the assumption that DL models are susceptible to similar features, which means that examples with different appearances may exhibit similar features in the resulting feature space of the model. This algorithm aims to generate adversarial examples that closely resemble the target domain while evoking the original predictions of the target model. GAA updates the GAN generator and the discriminator during each iteration using a custom loss function comprising three terms: adversarial loss, feature loss, and latent loss. The adversarial loss aims to deceive the discriminator with the generated instances, the feature loss maintains the similarity between the original and created features, and the latent loss adjusts the latent vector to diversify the generated examples. GAA has generated adversarial images that resemble the target images.
- **Average Gradient-Based Adversarial Attack:** This attack was proposed by Zhang et al. [103], which utilizes the gradients of the loss function to create a dynamic set of adversarial examples. Unlike existing gradient-based adversarial attacks, which only use the gradient of the current example, the average gradient-based adversarial attack aims for higher success rates and better transferability. In each iteration, this algorithm generates a dynamic set of adversarial examples using the gradients of the previous iterations. Then, it calculates the average gradient of the loss function concerning the dynamic set and the current example. The average gradient determines the perturbations added to the original example. Compared to its competitors, the average gradient-based adversarial attack has produced more effective and diverse adversarial examples and has outperformed them on various datasets and models. Overall, it has proven to be a practical and robust method for evaluating the robustness of DNNs.
- **Enhanced Ensemble of Identical Independent Evaluators (EIIEs) Method:** The enhanced EIIIE approach, introduced by Zhang et al. [104] built on the EIIIE topology by incorporating details on the top bids and requests for portfolio management. The primary concept of the EIIIE topology involves a neural network that assesses the potential growth of each asset in the portfolio based on the asset's data. The enhanced EIIIE method enhances the EIIIE topology by introducing two additional inputs: the best bid price and the best ask price for each asset. These inputs provide precise market information to assist the network in making optimal decisions regarding portfolio allocation. The EIIIE approach has shown the vulnerability of an RL agent in portfolio management. An attack could collapse the trading strategy, creating an opportunity for the attacker to profit.
- **Transformed Gradient (TG) Method:** A white-box approach generates adversarial perturbations by modifying the gradient of the loss function using a predefined kernel. The transfer gradient (TG) technique aims to reduce the susceptibility of adversarial instances to specific areas of the target model and improve their ability to be transferred to other models. This approach is compatible with any gradient-based

attack method, such as FGSM or PGD. TG can generate adversarial instances that are more easily transferred to other models than baseline approaches, especially when targeting defense models. The TG technique is an efficient method to improve the transferability of adversarial cases [105].

5.2.2. Black-Box Attack

Attackers cannot access internal details for target models in a black-box attack. The attacker can interact with the model by querying it and observing its output (e.g., predictions or confidence scores). Despite this limited access, the attacker can still craft adversarial examples using techniques such as transferability or query-based methods. For example, query-based attacks are where the adversary sends multiple inputs to a deployed model (e.g., an ML API) and observes the output to infer decision boundaries, as shown in Figure 8. The adversary then uses these observations to craft adversarial examples. The black-box and white-box adversarial attacks against DNN share the same objective formulated by Szegedy et al. [6]. They have the same list of compromised tasks but use different approaches to solve the problem.

In the white-box approach, the attacker attempts to modify the image pixels and observe the DNN's output. They then used a gradient distance based on a similarity metric to minimize image modification.

However, in a black-box attack, the attacker controls the modification of the image by applying iterative modifications that either maximize or minimize a specific objective function.

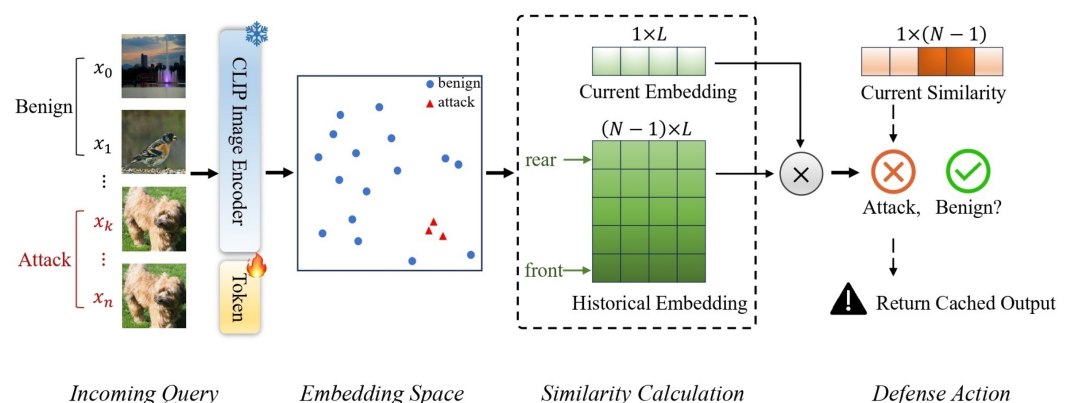


Figure 8. Proposed AdvQDet framework [106].

- **Square Attack:** This is a black-box adversarial attack that operates in both L_2 and L_∞ norms, and it does not depend on local gradient information, making it resistant to gradient masking. This attack utilizes a random search methodology, a classic iterative optimization approach proposed by Rastrigin in 1963 [107]. In particular, the results obtained from a square attack can provide more accurate assessments of the robustness of models that exhibit gradient masking compared to traditional white-box attacks [108].
- **One-Pixel Attack:** Begin by selecting an image point and applying a permutation to generate a set of modifications for this point. Next, utilize a differential evolution optimization algorithm to choose solutions that meet the target. Modified pixels should have more dimensions in common with another set of dimensions [109].
- **Multilabel Adversarial Examples—Differential Evolution (MLAE-DE):** Jiang et al. [110] introduced the MLAE-DE approach. It is an iterative method that uses differential evolution (DE) methodology to create multilabel adversarial instances capable of deceiving DNNs is a popular population-based optimization algorithm. It

involves misclassifying input from one or more source classes into a target class using a modified DNN model. This approach provided a breakthrough in DE by reducing the required fitness assessments and enhancing attack performance. MLAE-DE is a black-box approach that generates adversarial instances using only model outputs without accessing model parameters. It effectively creates adversarial images that resemble the target images while eliciting the original prediction of the targeted model. MLAE-DE has proven to be a valuable and efficient technique for generating multiple adversarial examples on large-picture datasets.

- **Black-box Attack Method Based on Successor Representation:** Cai et al. [111] introduced the SR-CRPA attack, a black-box strategy that manipulates rewards in the environment to train deep RL agents. The developers of the SR-CRPA method assumed that the adversary has access to the state, action, and reward information of both the agent and the environment. This approach involves using a pre-trained neural network to learn each state's SR, which indicates the anticipated future visitation frequency of the states. At each time step, SR-CRPA uses the SR value of the current-state action combination to determine whether to initiate an attack and how to subtly perturb the reward. With just a few attacks, SR-CRPA prevents the agent from acquiring the optimal policy. In general, SR-CRPA is a discrete and effective algorithm to poison the rewards of the target agent.
- **Dual-Stage Network Erosion (DSNE):** Duan et al. [112] introduced the DSNE attack as a black-box approach that uses DSNE to modify the source model and create adversarial samples. The DSNE method assumes that the adversary has access to the internal components of the source model but not to the target model. This technique involves modifying the original model's features to generate various virtual models combined by a longitudinal ensemble across iterations. DSNE biases the output of each residual block towards the skip connection in residual networks to reveal more transferable information. This results in enhanced transferability of adversarial cases with comparable computational expense. The results of the DSNE attack indicate that DNN structures are still vulnerable and network security can be improved through improved structural design.
- **Serial Minigroup Ensemble Attack (SMGEA):** The SMGEA attack, developed by Z et al. [113], is a transfer-based black-box technique designed to generate adversarial examples targeting a set of source models and then transferring them to various target models. The method assumes that the attacker can access multiple pre-trained white-box source models, excluding the target models. Divides the source models into multiple "minigroups" and uses three new ensemble techniques to improve transferability within each group. SMGEA employs a recursive approach to gather "long-term" gradient memories from the minigroup before and transfer them to the following minigroup. This procedure helps maintain acquired adversarial information and enhance intergroup transfer.
- **CMA-ES-Based Adversarial Attack on Black-Box DNNs:** Kuang et al. [114] presented a black-box method called a CMA-ES-based adversarial attack. This method creates adversarial samples for image classification using the CMA-ES. The attacker only has access to the input and output of the DNN model, the top K labels, and their confidence. The CMA-ES attack uses a Gaussian distribution to model variations in adversarial scenarios. It modifies the average and variation in the distribution according to the sample's quality. The attack generates altered images in every iteration, picking the most successful ones to modify the distribution.
- **Subspace Activation Evolution Strategy (SA-ES):** Li et al. [115] introduced the SA-ES algorithm, a method to create adversarial samples for DNNs using a zeroth-order

optimization method. This method assumes that the opponent can only request the model's softmax probability value for the input through queries without being able to access the model directly. The SA-ES algorithm uses a subspace activation technique and a block-inactivation technique to detect sensitive regions in the input image that could alter the model output. In every iteration, SA-ES uses an organic evolutionary method to modify the distribution parameters of perturbations towards the optimal direction. SA-ES can efficiently find high-quality adversarial examples with a limited number of queries. This algorithm is a reliable and effective approach for focusing on the model in both digital and physical settings.

- **Adversarial Attributes:** In a study, Wei and colleagues [116] proposed the adversarial attributes attack. This method creates harmful examples for image classifiers by changing the semantic characteristics of the input image. The technique employs a preexisting attribute predictor to analyze the visual characteristics of each image, such as hair color. The adversarial attribute attack detects the most impactful attributes in every round and merges them with the original image to generate the perturbation. Adversarial attribute attacks can obtain a high success rate and remain undetected with only a few queries by using evolutionary computation to solve a constrained optimization problem.
- **Artificial Bee Colony Attack (ABCAAttack):** Cao et al. [117] presented an artificial bee colony (ABC) attack, which creates adversarial examples for image classifiers using the ABC algorithm. The ABC attack assumes that the attacker can only request the softmax probability value of the model input without being able to access the model itself. ABC attack produces a set of modified images during each iteration and chooses the best ones to adjust the spread. It can accurately determine the most effective changes that cause the target model to incorrectly classify input images while staying within a set query limit. ABC attack is a method that has demonstrated effectiveness in fooling black-box image classifiers while being efficient and resistant to queries.
- **Sample Based Fast Adversarial Attack Method:** Wang et al. [118] introduced the sample-based fast adversarial attack method. This technique generates adversarial examples for image classifiers using data samples in a black-box approach. The technique assumes that the opponent can only obtain data samples, not the model parameters or gradients. Principal component analysis is used to identify the main differences between categories and calculate the discrepancy vector. Creating a target adversarial sample with minor adjustments involves performing a bisection line search along the vector that connects the current class to the target class.
- **Attack on Attention (AoA):** Chen et al. [119] developed the AoA attack, a black-box technique that generates adversarial examples for DNNs by manipulating the attention heatmaps of the input images. This attack assumes that the attacker can only request the model's softmax probability value for the input without accessing the model itself. The approach utilizes a pre-trained attention predictor to learn attention heatmaps for each image, demonstrating the areas on which the model focuses. AoA modifies the attention heatmaps in each iteration by adding or subtracting pixels to create the perturbation.

5.2.3. Poisoning Attack

Typically, an attacker can only contaminate a defender's dataset without modifying existing training data (D_{tr}). The attacker can introduce poisoned data, actively or passively, to manipulate the model during retraining, causing the model to misclassify specific testing data. This situation may arise when the defender collects data from various sources, some of which may be untrustworthy (e.g., scraping data from the public Internet) [120]. When

the attacker has access to the testing data, they insert a trigger to manipulate them, causing misclassification, known as a backdoor attack. Attackers can carry out these attacks on all ML tasks that involve learning from data sources. In classification tasks, the goal of the attacker is to misclassify test data, as proposed by [121]; in policy learning attacks, the aim is to modify agent behavior, as proposed by [122]; in synthetic data generation attacks, the attacker aims to manipulate the latent code [123].

Figure 9 shows traffic signs, including clean versions and backdoor versions altered with various triggers, such as a yellow square, bomb sticker, or flower.



Figure 9. Examples of backdoored traffic signs and their impact on classification accuracy [20].

The goal of the attacker can be presented as:

$$w_* = w_*(D_p) \in \arg \min_w L(D_{tr} \cup D_p, w) \quad (2)$$

where w_* is the attacker's desired model parameter, which realizes the attacker's objectives, and $L(\cdot)$ is the loss function. In order to achieve the target attacker objective, a set of proposed algorithms tries to infer the poison data from the clean defender data:

- **Feature Collection Attack:** Feature collision attacks occur within the framework of targeted data poisoning. The aim is to modify the training dataset so that the model incorrectly classifies a specific target, from the test set, into the base class. Techniques for feature collision in targeted data poisoning focus on adjusting training images from the base class to shift their representations in feature space closer to that of the target [124].
- **Label Flipping:** Label-flipping attacks involve changing the assigned labels during training, but keeping the data unchanged. Although not considered a “clean label”, these attacks do not leave easily detectable artifacts that the target could recognize [125].
- **Influence Functions:** Researchers can use influence functions to evaluate how a slight modification in training data impacts the model parameters learned during training.

This can help to create poisoning examples and then be used to approximate solutions to the bilevel formulation of poisoning [126].

- **TensorClog:** In their work, Shen et al. [127] introduced the TensorClog attack, which is known as a black box attack, which involves inserting perturbations into the input data of DNNs to reduce their effectiveness and protect user privacy. The TensorClog attack assumes that the attacker can access a substantial amount of user data but not the model parameters or gradients. The approach uses a pre-trained frequency predictor to analyze the frequency domain of each input, capturing its spectral features. TensorClog modifies the frequency domain of the input by adding or subtracting pixels on each iteration to create the perturbation. The attack can effectively identify perturbations that cause the target model to converge to a higher loss and reduced accuracy using fewer queries. TensorClog is a covert and efficient technique used to corrupt the input data of DNNs.
- **Text Classification:** In their work, Zhang et al. [128] presented a method for identifying vulnerabilities in NLP models with limited information. They discovered the existence of universal attack triggers. Their approach consists of two stages. First, they extract harmful terms from the adversarial sample to create the knowledge base in a black-box scenario. In the second phase, universal attack triggers are created by altering the predicted outcomes for a set of samples. Incorporating the generated trigger into any clean input can significantly decrease the prediction accuracy of the NLP model, potentially bringing it down to nearly zero.

5.2.4. Extraction Attacks

Correia-Silva et al. [129] proposed replicating a targeted network using only clean images for querying. The method involves two main steps: creating fake data and training the copycat system. By selecting multiple images, the adversary creates a fake dataset, which can be from the target network or a different image collection, based on their access to the specific model. The system deems the original image labels irrelevant and removes them from all image sets. The attacker aims to understand how the targeted network classifies images during this stage. To carry out this process, the new dataset is fed into the target model and observe how it categorizes the images. The stolen labels are the labels given to the new images. The adversary aims to exploit the classifier's minor vulnerabilities, allowing another network trained on the same dataset to generate results that resemble the target model.

Figure 10 illustrates the workflow of a model-stealing attack conducted on a black-box neural network (referred to as the “target network”). The target network is initially trained on a confidential dataset and is deployed as a public API. This API allows users to submit input images and receive corresponding class labels as output.

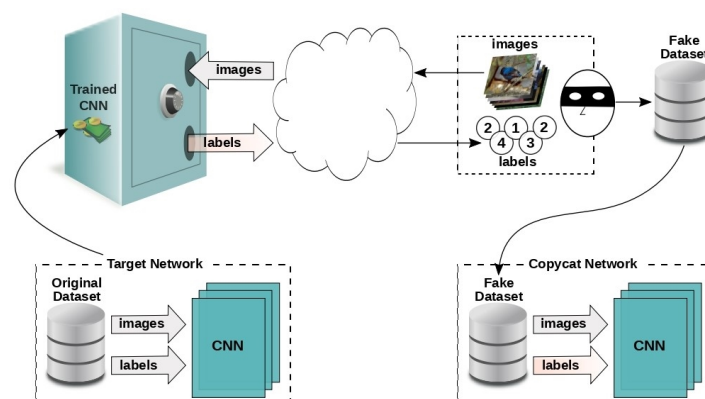


Figure 10. Model stealing process [90].

Table 2 provides a comprehensive overview of adversarial attacks against DNNs, summarizing key details from various research articles. Each row corresponds to a specific study, featuring information such as the article title, publication year, ML task type (e.g., classification, representation, policy learning), type of attack (e.g., black-box or white-box), task type (classification or policy learning), the algorithm employed, attack scenario (testing or training), assumed knowledge level (black-box or white-box), attack target (e.g., misclassification), and the search strategy employed (e.g., greedy, iterative, gradient, metaheuristic). This structured presentation facilitates a quick understanding of the diverse methodologies used in adversarial attacks, offering valuable insights to researchers and practitioners on the evolving landscape of adversarial ML techniques.

Table 2. A comparison of adversarial attack against DNN strategies.

Title	Year	Attack	Task Type	Algorithm	Scenario	Knowledge	Target	Search
Bai et al. [96]	2022	White-Box	Classification	weight attack	Testing	White Box	Misclassify	Gradient
Wang et al. [97]	2023	Black-Box	Classification	ADSAAttack	Training	Gray-Box	Misclassify	Greedy
Rakin et al. [98]	2021	White-Box	Classification	T-BFA	Testing	White-Box	Misclassify	Greedy
Wang et al. [99]	2021	White-Box	Classification	N/A	Training	White-Box	Misclassify	Gradient
Tang et al. [101]	2021	White-Box	Classification	Type I	Training	White-Box	Misclassify	Gradient
He et al. [102]	2023	black-Box	Classification	GAA	Training	black-Box	Misclassify	Iterative
Wan et al. [103]	2023	White-Box	Classification	Average gradient-based	N/A	White-Box	Misclassify	Gradient
Chen et al. [104]	2021	black-Box	Policy Learning	ElIE method	Both	black-Box	agent behavior	Greedy
He et al. [105]	2022	Black-Box	Classification	TG method	Testing	Black-Box	N/A	Gradient
Kong et al. [110]	2023	Black-Box	Classification	DE	Testing	Black-Box	Misclassify	Greedy
Cai et al. [111]	2022	Black-Box	Policy Learning	SR	Testing	Black-Box	agent behavior	Iterative
Duan et al. [112]	2022	Black-Box	Classification	DSNE	Training	Black-Box	residual networks	Gradient
Che et al. [113]	2022	Black-Box	Classification	SMGEA	Testing	White Box	N/A	Gradient
Kuang et al. [114]	2019	Black-Box	Classification	CMA-ES	Testing	Black-Box	CyberPhysical Systems	Iterative
Li et al. [115]	2023	Black-Box	Classification	SA-ES	Testing	Black-Box	Misclassify	Gradient
Wei et al. [116]	2021	Black-Box	Classification	N/A	Testing	Black-Box	Misclassify	Gradient
Cao et al. [117]	2022	Black-Box	Classification	ABCAttack	Testing	Black-Box	Misclassify	Metaheuristic
Wang et al. [118]	2019	Black-Box	Classification	N/A	Training	Black-Box	Misclassify	Iterative
Chen et al. [119]	2022	Black-Box	Classification	AoA	Training	Black-Box	Misclassify	Gradient
Shen et al. [127]	2019	Poisoning	Classification	TensorClog	Training	White-Box	Accuracy	Gradient
Shao et al. [128]	2022	Poisoning	Text Classification	N/A	Both	Black-Box	Accuracy	Greedy
Shen et al. [130]	2023	Black-Box	Representation Learning	DEAL	Testing	Black-Box	Misclassify	Iterative
Xiang et al. [131]	2021	Gray-Box	Classification	N/A	Testing	Gray-Box	Misclassify	Iterative
Hwang et al. [132]	2019	White-Box	Representation Learning	PuVAE	Training	N/A	N/A	Iterative

6. Defense Strategies

Adversarial example defenses employ two approaches: (1) reactive approach after DNNs are constructed, and (2) proactive approach by making DNNs more robust before adversaries generate adversarial examples.

Both approaches involve strategies, as shown in Figure 11. This figure provides a taxonomy of defense strategies against adversarial attacks on DNNs, focusing on proactive and reactive approaches. The top-level division categorizes proactive strategies that enhance the model's robustness before adversarial attacks occur, and reactive strategies implemented after the DNN is constructed. Each approach includes specific defense mechanisms. Proactive strategies include “modifying the model”, “modifying the training”, “transforming the inputs”, and “add-ons”. reactive strategies include “adversarial detection”, “input reconstruction”, “network verification”, and “traceback poison data”. The color-coded visual representation helps to quickly understand the overarching taxon-

omy of defense mechanisms, providing a concise and organized overview for researchers and practitioners.

Defense methods that change the training ANN model or transform data aim to reduce the effects of image permutation in DNN. However, add-ons implement more layers, add subnets, change the loss function, or use external models to discover and stop attacks. This section discusses various methods to protect a DNN model from adversarial attacks, focusing on both defense approaches.

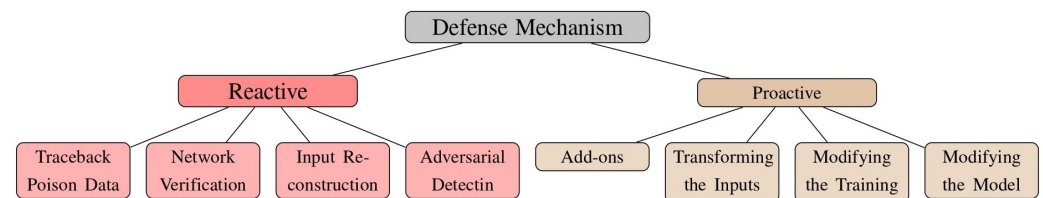


Figure 11. Defense strategy taxonomy.

6.1. Proactive Approach

6.1.1. Modifications to the ANN

This defense strategy reduces the impact of the permutation added by white-box and black-box attacks to misclassify the DNN result by adding a slight permutation to the input data. Various algorithms have been proposed to achieve this objective:

1. **Gradient Regularization:** Training DNNs can be challenging, as they tend to penalize even the most minor deviations in the inputs. The relative entropy between the labels and predictions may become negligible with slight input changes. That a small perturbation would probably change the output of a trained classifier [133].
2. **DeepCloak:** DeepCloak aims to eliminate redundant features that a malicious party could exploit to create adversarial samples. Before the classification layer, the authors suggest incorporating a masking layer in the DNN model. This mask layer learns by processing pristine and altered images in a forward pass, capturing the variances in the output of earlier layers. DeepCloak filters out unnecessary features by setting them to 0, effectively removing them from adversarial examples [134].
3. **Deep Contractive Networks:** Researchers developed denoising autoencoders (DAEs) to address adversarial noise. They train a DAE to identify and eliminate such noise. However, when they pair the DAE with the original network, the stacked network becomes more vulnerable to attacks than the original model. However, when paired with the original network, the stacked network becomes more vulnerable to attacks than the original model. Deep contractive networks (DCNs) were presented by Gu and Rigazio, which incorporate a smoothness penalty similar to that of contractive autoencoders, a type of autoencoder with an additional minimization penalty. DCNs improve the resilience of the network to adversarial attacks while maintaining performance levels [135].

6.1.2. Additions to the ANN

The following defense strategy involves modifying the DNN by adding more layers and subnets, changing the loss function, or using external models to discover and stop potential attacks. Researchers have proposed various algorithms to achieve this objective:

1. **Defense against UAP:** UAPs reveal a significant vulnerability in the security of ML models. Akhtar et al. propose a strategy to defend against UAPs by incorporating a perturbation rectifying network (PRN) as an initial layer in the targeted model, avoiding modifying it. The PRN detects disturbed images entering the network and reconfigures them to ensure that they are classified with the same label as the original

image. Researchers train the PRN using datasets that include authentic and synthetic UAPs while keeping the model parameters unchanged. They also train a perturbation detector on the cosine transform of the differences between the PRN's inputs and outputs [136].

2. MagNet: Meng and Chen proposed MagNet to defend neural network classifiers against adversarial attacks. MagNet uses distinct detector and reformer networks to identify whether incoming input images are adversarial. Researchers construct detector networks using either a reconstruction error or probability divergence. During training, they estimate the range of clean inputs and distinguish between clean and perturbed images. In testing, the system identifies and removes images that deviate from the manifold as adversarial. Reformer networks, employing random noise or an autoencoder, guide adversarial images with slight perturbations back toward the unperturbed image space, ensuring accurate classification. MagNet effectively defends against black-box and gray-box attacks, as it applies to various attacks without depending on a specific perturbation generation method [137].

6.1.3. Modifications to the Training

This defense strategy adjusts the DNN training process to minimize the impact of permutation on the model.

Brute-Force Training: Researchers improve the robustness of the model against adversarial examples through adversarial training. They train DNN models using adversarial examples with their correct labels. To implement this method, they need a robust attack algorithm, an exposed model, and a large dataset for adversarial training. Due to these requirements, researchers commonly refer to adversarial training as brute-force training [14].

6.1.4. Input Transformations

In addition to protecting neural networks from adversarial attacks, models can be more robust by adjusting their input. If data is not compressed, perturbed images can harm the model. Most classification datasets consist of JPG images. The researchers used this observation to study the effects of JPEG compression on adversarial images created by FGSM. They discovered that JPG compression often reversed the classification of the adversarial image when FGSM generated small perturbations [138].

6.1.5. Defense Against Poisoning Attacks

This defense strategy aims to prevent poisoning by identifying the trigger that is contaminating the training process.

Tran et al. [139] proposed an innovative technique to protect neural networks against poisoning backdoor attacks through spectral signatures for detection. Their study revealed that, following a backdoor attack, a distinct trace of the attack persists within the covariance of the model's learned feature representation. This trace, termed a spectral signature, becomes a key indicator. The researchers assumed that the input data contained clean and adversarial samples for each label. Backdoor-adversarial samples with introduce a strong signal into the representation vector. By applying singular value decomposition, these signals can be effectively identified, facilitating the detection and removal of the images responsible for them.

6.2. Reactive Approach

6.2.1. Adversarial Detecting

Metzen et al. [140] introduced a method for detecting adversarial examples by incorporating an auxiliary network that operates in parallel with the primary neural network. This detector is a simple neural network that performs binary classification, determining

the likelihood that an input is an adversarial example. To extract the features of the adversarial detector, SafetyNet obtained the binary threshold of each ReLU layer output and used an RBF-SVM classifier to identify adversarial images [141]. The authors claimed that their method is intricate for adversaries to evade, as finding optimal values for both the adversarial examples and the new features of the SafetyNet detector is challenging.

6.2.2. Detection Based on Activation Analysis

Detecting and protecting a DNN model from backdoor attacks is highly challenging because only the attacker knows the triggers of the backdoors. Backdoors are activated when the model identifies specific attributes associated with the original class, leading to network activations that mirror the model's decision. Chen and colleagues devised an activation clustering method to detect compromised input images used to implant backdoors in DNNs. Their method analyzes neural network activations to detect backdoors [121].

6.2.3. Input Reconstruction

Reconstruction can transform adversarial examples into clean data, which makes them unable to affect the predictions of DL models. Gu and Rigazio proposed a variation of the autoencoder, known as the deep contractive autoencoder, which incorporates a regularization term to improve the resilience of neural networks. Furthermore, they trained a denoising autoencoder to map adversarial examples back to their original state, successfully mitigating the impact of adversarial perturbations [135].

6.2.4. Network Verification

Verifying the properties of DNNs offers a promising defense against adversarial examples, potentially enabling the detection of previously unseen attacks. Network verification involves examining whether a neural network adheres to specific properties by determining whether an input violates or satisfies these properties. Katz et al. [141] introduce a verification method for neural networks with ReLU activation functions, known as Reluplex.

6.2.5. Traceback Poison Data

Shawn et al. [25] proposed a method for taking reactive approaches using digital forensics when detecting a misclassification event. They aim to trace all poisoned training images created using the same attacking method. The authors classified all training images into groups based on their work's data characteristics and loss function. Then, poisoned images are detected by tracing the misclassified image back to its cluster, removing all images from that cluster in the training set, and retraining the model.

Table 3 offers a detailed summary of defense against attacks on DNNs. The key details of the article encompass the title, year of publication, defense method (reactive or proactive), defense classification (Defense Add-On or Modify-ANN), the mechanism's label, and the targeted attack category. The table is a valuable tool for examining different methods that researchers have created to enhance the robustness of DNNs against varying adversarial risks.

Table 3. Comparison of defense mechanism against DNN attacks.

Title	Computational Cost	Approach	Type	Task Type	Mechanism	Against
Guan et al. [133]	Moderate	Reactive	Defense-Add-On	Classification	IACS	FGSM, PGD, JSMA, DeepFool, and CW2 black-box
Aithal et al. [142]	Low	Proactive	Defense-Add-On	Classification	White Gaussian Noise Injection	
Wei et al. [143]	High	Proactive	Defense-Add-On	Classification	XEnsemble	Evaluated by 11 popular adversarial attacks against a targeted-BFA black-box and white-box against black-box adversarial attacks, such as FGSM and PGD
Liu et al. [144]	Low	Proactive	Modifying ANN	Classification	RREC	
Zoppi et al. [145]	Low	Reactive	Defense-Add-On	Classification	monitoring GPU	
Nguyen et al. [146]	high	Proactive	Modifying ANN and Defense-Add-On	speech processing	augmentation methods	
Yang et al. [147]	Moderate	Reactive	Network Verification	Classification	AutoAttack detection	FGSM, PGD
Ryu et al. [148]	Low	Reactive	Defense-Add-On	Classification	entropy-based detector (EBD)	FGSM, BIM, MIM, DeepFool, and CW
Wu et al. [149]	High	Proactive	Modifying ANN and Defense-Add-On	Classification	Attention-based Adversarial Defense (AAD)	FGSM, PGD, and CW
Sotgiu et al. [150]	High	Reactive	Defense-Add-On	Classification	deep neural rejection (DNR)	multi-layer anomaly rejection approach
Xiang et al. [151]	moderate	Proactive	Modifying ANN	Classification	optimization-driven defense	Backdoor Attack
Shao et al. [152]	Moderate	Reactive	Defense-Add-On	Identifying suspicious words in input text	Backdoor Defense model based on Detection and Reconstruction	Backdoor Attack
Shao et al. [153]	Low	Proactive	Modifying ANN	text classification tasks	The defense method centered on recognizing poisoned samples.	against various backdoor attacks(character-level, word-level, and sentence-level backdoor attacks).
Hwang et al. [132]	High	Reactive	Defense-Add-On	Classification	Variational Autoencoder (VAE)	FGSM, iFGSM, and C&W

6.3. Challenges of Defenses on DNNs in Real-World Systems

Defending DNNs in real-world systems poses significant challenges. Unlike controlled research settings, real-world systems face various adversarial inputs, diverse attack vectors, and resource constraints, making it difficult to implement effective and scalable defenses. One of the primary challenges is the adaptive nature of adversaries, where attackers continuously develop sophisticated techniques to bypass existing defenses, rendering static solutions ineffective over time. Alsobeh et al. [154] explore machine-learning techniques for malware detection, a field closely related to the detection of adversarial threats. The proposed time-aware ML approach has the potential to enhance the real-time detection of adversarial attacks. Wang et al. [155] present the efficient robust mode connectivity (EMRC) method as a new approach for defending against various ℓ_p -norm attacks, which may leave the models vulnerable to other types of attacks. Achieving universal robustness across multiple perturbation types often incurs significant computational costs and complexity, making such approaches less practical for large-scale deployment. Moreover, adversarial training methods, though effective against targeted attacks, tend to degrade model performance on clean data, introducing a trade-off between robustness and accuracy. Another critical challenge is ensuring the scalability of defenses to larger datasets and complex architectures, as many existing solutions are computationally intensive. Additionally, the dynamic nature of adversarial strategies, where attackers continually innovate new methods, complicates the creation of defenses that are resilient over time. Defending DNNs against adversarial attacks in real-world applications presents several challenges. Firstly, the diverse nature of adversarial attacks, such as black-box, white-box, and gray-box attacks, complicates the development of universal defenses. Real-world black-box attacks are particularly concerning due to their feasibility and impact. Moreover, model complexity

does not guarantee robustness; larger models with more parameters are often more vulnerable to attacks, while smaller, simpler models can be less effective in handling sophisticated perturbations. Dataset characteristics, such as input size and number of classes, further influence the success of adversarial attacks and the efficacy of defenses. Preprocessor-based defense techniques, such as bit squeezing, median smoothing, and JPEG filtering, can reduce attack effectiveness, but their success varies across datasets and attack types. Finally, achieving a balance between robustness, computational efficiency, and maintaining high accuracy on clean data remains a critical challenge in real-world applications [156].

Real-time monitoring and adaptation of defenses are challenging, especially in mission-critical applications where system integrity and performance are paramount. Advanced AI-driven monitoring frameworks, though promising, require fine-tuning and validation to address inconsistencies and ensure dependable protection. These limitations highlight the necessity for integrated, adaptive, and scalable approaches to bolster the resilience of DNNs against adversarial threats [157].

The application of ML techniques in defect prediction has garnered significant attention, as it closely mirrors the process of identifying vulnerabilities in adversarial scenarios. By leveraging ML models, researchers aim to detect patterns and anomalies within datasets, enabling the identification of defects or weaknesses. This approach not only enhances software quality assurance but also provides a framework that aligns with methodologies used to uncover adversarial vulnerabilities in security-critical systems [158].

6.4. Discussion

The effectiveness of defense mechanisms against adversarial attacks largely depends on the specific nature of the attack and the vulnerabilities it exploits. For instance, adversarial training excels against gradient-based attacks like FGSM and PGD because it explicitly incorporates adversarial examples into the training process, making the model more robust to small perturbations. However, it may fail against data poisoning attacks, where adversaries manipulate the training data itself, as it does not address the root cause of the poisoned data. In contrast, traceback poison data methods are highly effective against poisoning attacks because they identify and remove malicious data, but they are less valuable against adversarial examples crafted during inference. Similarly, input transformations (e.g., randomization or quantization) can neutralize adversarial perturbations by disrupting their structure, making them effective against attacks that rely on precise input modifications. However, these methods may degrade legitimate inputs and fail against adaptive attacks to bypass such transformations. On the other hand, network verification provides strong theoretical guarantees against adversarial examples by mathematically proving robustness within a defined input space. Still, it is computationally expensive and scales poorly to large models. This highlights a key trade-off: defenses tailored to specific attack types often struggle to generalize to other threats, while more general approaches may lack the precision needed to counter sophisticated attacks. Ultimately, the success of a defense mechanism hinges on its alignment with the attack's characteristics and the model's vulnerabilities, underscoring the need for a multifaceted defense strategy that combines complementary approaches to resolve a broader diversity of adversarial scenarios.

7. Experimental Setup and Performance Metrics

Adversarial attacks and defenses are commonly evaluated using different datasets, models, and performance measures based on different factors, such as the goal, how the experiment was conducted, the resources used to carry out that attack or the impact of the attack.

7.1. Datasets

To evaluate the robustness of DNNs against adversarial attacks, we consider a diverse set of benchmark datasets commonly used in adversarial ML research. These datasets span multiple domains, including image classification, natural language processing, and audio processing. Below is a brief description of the datasets.

7.1.1. MNIS and F-MNIST

A benchmark dataset for handwritten digit recognition, consisting of 60,000 training and 10,000 test grayscale images. It is widely used to evaluate adversarial attacks due to its simplicity and low dimensionality [159].

Xiao et al. propose the creation of FashionMNIST [160], a variant of the MNIST dataset featuring 60,000 grayscale images of fashion products across 10 categories.

7.1.2. CIFAR-10

A dataset of 60,000 32×32 color images across 10 classes is commonly used to evaluate adversarial attacks in more complex image classification tasks [161].

7.1.3. ImageNet and ILSVRC

A large-scale dataset for visual recognition containing more than 14 million labeled images [162]. The ILSVRC 2016 subset is often used to evaluate adversarial attacks on high-dimensional data. A smaller version of the ImageNet dataset, containing 100,000 images in 200 classes, resized to 64×64 pixels. It is used to evaluate adversarial attacks in medium-scale image classification tasks.

7.1.4. CelebA

A wide-range face attributes dataset with more than 200,000 celebrity images, each annotated with 40 binary attributes [163].

7.1.5. Street View House Numbers (SVHNs)

A dataset of real-world house number images used to evaluate adversarial attacks in digit recognition tasks [164].

7.1.6. IMDB

A dataset of movie reviews annotated with binary sentiment labels, used for binary sentiment classification tasks to evaluate adversarial attacks in natural language processing [165].

7.1.7. SST-2 (Stanford Sentiment Treebank)

The Stanford Sentiment Treebank contains 6920 training samples, 872 validation samples, and 1821 test samples. A sentiment analysis dataset used to evaluate adversarial attacks on text classification models [166].

7.1.8. ASVSpooF

Datasets for audio spoofing detection are used to evaluate adversarial attacks on audio processing models [167].

7.1.9. Fake or Real

A dataset for detecting fake and actual audio samples, commonly used in anti-spoofing research [168].

7.1.10. ADD2023

Audio deepfake detection dataset featuring a wide variety of genuine and synthetic audio samples [169].

7.1.11. In-the Wild (ItW)

A dataset for in-the-wild audio spoofing detection, containing diverse and challenging real-world scenarios [170].

7.1.12. Taiwan Stock Exchange (TWSE)

The TWSE dataset is widely utilized in financial research. The dataset contains comprehensive stock market data, including daily stock prices, trading volumes, and other related financial indicators for companies listed on the Taiwan Stock Exchange. Its structured and detailed nature makes it a valuable resource for time series analysis and trend forecasting studies [171].

7.1.13. Freeway

The Freeway datasets contain detailed vehicle trajectory data collected from the east-bound section of Interstate 80 in Emeryville, California, located within the San Francisco Bay Area. Data collection took place on 13 April 2005, covering a study area of approximately 500 m (1640 feet), including six freeway lanes, one designated for high-occupancy vehicles (HOVs) and an onramp within the observation area. Seven synchronized digital video cameras, mounted on a 30-story building adjacent to the freeway, recorded vehicle movements within the study area. Using NG-VIDEO, a specialized software developed for the program, researchers extracted vehicle trajectory data from the video recordings. The dataset provides precise information on each vehicle's location within the study area every 0.1 s, including detailed lane positions and spatial relationships relative to other cars. It is a valuable resource for analyzing traffic dynamics and behavior [172].

7.1.14. Ember Malware

The dataset consists of features derived from 1.1 million binary files, comprising 900,000 training and 200,000 test samples. The training set includes an equal distribution of 300,000 malicious, 300,000 benign and 300,000 unlabeled files, while the test set contains 100,000 malicious and 100,000 benign samples [173].

7.1.15. STL-10

A dataset designed for unsupervised and semi-supervised learning, containing 10 classes of labeled and unlabeled images [174].

7.2. Neural Network Architectures

To evaluate the robustness of DNNs against adversarial attacks, we consider a diverse set of benchmark architectures commonly used in adversarial ML research. These architectures span multiple domains, including computer vision, natural language processing, and hybrid models. Below is a brief description of the architectures used in this study:

7.2.1. ResNet-50

A 50-layer deep residual network widely used for image classification tasks [175]. Its skip connections make it robust to vanishing gradients but remain vulnerable to adversarial attacks like FGSM and PGD.

7.2.2. VGG-16

A 16-layer convolutional neural network (CNN) known for its simplicity and depth. Its large number of parameters makes it susceptible to adversarial perturbations [176].

7.2.3. GoogleNet (InceptionV1)

A 22-layer network with inception modules designed to reduce computational cost while maintaining accuracy. It has been extensively tested for adversarial robustness [177].

7.2.4. MobileNet-v2

A lightweight CNN designed for mobile and embedded vision applications. Its efficiency makes it a target for resource-constrained adversarial attacks [178].

7.2.5. DenseNet121

A densely connected CNN with 121 layers, where each layer is connected to every other layer in a feed-forward manner. Its dense connectivity improves feature reuse but introduces vulnerabilities in adversarial examples [179].

7.2.6. LSTM (BiLSTM)

A bidirectional long short-term memory network with max-pooling used for text classification tasks. It is vulnerable to adversarial attacks in NLP, such as word-level perturbations [180].

7.2.7. BERT

A pre-trained transformer-based language model, a crucial performance metric widely used for NLP tasks. Despite its strong performance, BERT is susceptible to adversarial attacks such as TextFooler and BERT-Attack [181].

7.2.8. PuVAE Model

A probabilistic variational autoencoder designed for robustness against adversarial attacks in generative tasks [132].

7.3. Performance Metrics

Performance measures can be categorized into the following categories:

7.3.1. Rate-based

Rate-based metrics quantify performance as a ratio or percentage of successfully achieved results relative to total attempts. Researchers most often use these measures to report the success rate in achieving attack or defense targets. The most common measures include the following:

- **Success Rate:** Attack success rate (ASR) or success rate refers to the proportion of adversarial examples that succeed within the specified attack budget [182].

$$ASR = \left(\frac{\text{Number of Successful Adversarial Attacks}}{\text{Total Number of Adversarial Attempts}} \right) \times 100\% \quad (3)$$

- **Average Success Rate (AvSR):** The AvSR is a key metric for assessing the performance of continual learning methods. Measures the ratio of successful results across multiple experiments or trials, offering insights into how well a model adapts to changing tasks and data while preserving accuracy and stability. Let ASR_i be the ASR of task T_i

after training on all N tasks [183]. The continual objective of imitation learning is to maximize the AvSR across all tasks:

$$AvSR = \frac{1}{N} \sum_{i=1}^N ASR_i \quad (4)$$

- **Target Fooling Rate (TFR):** The equation defines the event's occurrence rate, with a higher rate indicating that the classifier is more easily fooled [184].

$$\operatorname{argmax}(C(\hat{x})) = t \neq c_x, \quad x \approx \hat{x}. \quad (5)$$

In targeted attacks, adversaries add perturbations to the input data to manipulate the model's output. The objective is to ensure that the argmax function outputs the target class rather than the actual class. For example, suppose the true class is a cat, and the adversary wants the model to classify it as a dog. In that case, the adversary will adjust the input until the argmax function indicates the "dog" class as having the highest score. The TFR evaluates the effectiveness of the attack. If a significant proportion of adversarial examples successfully achieve the desired misclassification (i.e., the argmax function points to the target class), the TFR will be high, indicating a successful attack.

- **Misclassification Rate (MR):** The misclassification event's occurrence rate (a higher rate indicates that the classifier is easily fooled). It measures the proportion of instances the model incorrectly classifies, providing insight into its accuracy and reliability. UPSET and ANGRI, MR is defined as the rate at which the classifier outputs a label different from the true class label of the input, according to Equation [184].

$$\operatorname{argmax}(C(\hat{x})) \neq c_x, \quad x \approx \hat{x}. \quad (6)$$

It also measures the proportion of instances that the model incorrectly classifies, providing insight into its accuracy and reliability [185]. It can be expressed in terms of false positives and false negatives, respectively. True negatives (TNs), true positives (TPs), false positives (FPs), and false negatives (FNs) are calculated as follows:

$$\text{Misclassification Rat} = \frac{FP + FN}{TP + TN + FP + FN} \quad (7)$$

7.3.2. Classification-Based Measures

These performance measures indicate how successfully samples are classified into two or more groups. Defense mechanisms against poisoning attacks commonly use this measure to assess how well they distinguish adversarial poisoning examples from benign examples [25,147]. The following measures are commonly used:

- **Accuracy (AC):** The classification AC is determined by the number of correct predictions divided by the total number of input samples. In ML, the accuracy can be calculated from the confusion matrix, which includes the counts of TP, TN, FP, and FN. The confusion matrix summarizes the predictions made by a classification model, comparing the predicted outcomes to the actual outcomes [182].

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \quad (8)$$

- **Accuracy Drop or Error Rate:** Accuracy drop (AD) or error rate (ER) is a key metric for assessing the robustness of DNNs against adversarial and backdoor attacks. Quantifies

the reduction in classification performance when a model is exposed to adversarial perturbations, backdoor triggers, or defense mechanisms and measures the proportion of incorrect predictions under these conditions.

$$\text{Error Rate} = \frac{\text{Number of Incorrect Predictions}}{\text{Total Number of Predictions}} \times 100\% \quad (9)$$

- **Precision:** It is a crucial performance metric that indicates how effectively a model can identify positive instances among those it classifies as positive. This metric is particularly significant in situations with a class imbalance, where one class is significantly underrepresented; critical fields such as healthcare and fraud detection face substantial implications from false positives, which can lead to unnecessary interventions or financial losses [182].

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (10)$$

- **Recall:** Also called sensitivity or the true positive rate, it quantifies the fraction of actual positive instances the model successfully recognizes [182].

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (11)$$

7.3.3. Efficiency-Based

Researchers commonly use these measures to report the aggregation of trial results to achieve attack or defense targets. Aggregation-oriented metrics in DNNs assess the overall performance of the model. These metrics are advantageous in situations involving multiclass classification or imbalanced datasets, where a model's performance can vary significantly across different classes. Aggregation-based metrics are essential in real-world tasks such as image classification and NLP. Using these measurements, researchers can obtain a more thorough understanding of how DNNs behave, allowing them to make informed decisions about improving models and adjusting strategies. Commonly used measures include the following:

- **Avgquery:** The average amount of queries is a crucial performance measure. This measurement helps in assessing the effectiveness of adversarial attacks on ML models for black-box attacks. The average query numbers indicate how many queries an attacker must make to successfully create an adversarial example using the model. This is especially important in situations where the adversarial degree of knowledge of attackers in the black box must only use the output, commonly using this metric to measure the number of queries needed to create adversarial examples for the victim model in black-box attacks [117].

7.4. Model Stealing

Macro-averaged accuracy is a widely used evaluation metric that calculates the average accuracy across all classes in a dataset. Unlike micro-averaged accuracy, which weighs classes based on their sample size, the macro-averaged approach gives equal importance to each class, regardless of the number of instances per class. By averaging the accuracy of each class individually, the metric highlights how well a model performs in each category rather than favoring the majority class. Mathematically, macro-averaged accuracy is calculated by taking the sum of the accuracies for all classes and dividing it by the total number of classes. It can be expressed as follows:

- **Macro-Averaged Accuracy:** This metric is a widely used evaluation metric that calculates the average accuracy across all classes in a dataset. Unlike micro-averaged accuracy, which weighs classes based on their sample size, the macro-averaged approach gives equal importance to each class, regardless of the number of instances per class [129]. By averaging the accuracy of each class individually, the metric highlights how well a model performs in each category rather than favoring the majority class. Mathematically, macro-averaged accuracy is calculated by taking the sum of the accuracies for all classes and dividing it by the total number of classes. For a dataset with C classes, it can be expressed as follows:

$$\text{Macro-Averaged Accuracy(MAA)} = \frac{1}{C} \sum_{i=1}^C \text{Accuracy for Class } i \quad (12)$$

- **Performance Over Target Network (OD):** This metric evaluates the accuracy of the copycat network relative to the original target network (OD). It is calculated as the ratio of the copycat network's macro-averaged accuracy to that of the target network, expressed as a percentage. A perfect copycat network would achieve 100% of the target network's performance, replicating its predictions exactly, including any mistakes. The formula for this metric is as follows:

$$\text{Performance Over Target Network} = \frac{\text{MAA of Copycat Network}}{\text{MAA of Target Network}} \times 100 \quad (13)$$

This metric is crucial for determining how closely the copycat network approximates the original model, particularly when the latter is inaccessible for direct evaluation due to proprietary constraints [129].

- **Performance Over Smaller Problem Domain Network (PD-OL)** This metric compares the accuracy of the copycat network to a baseline model trained on a smaller, limited dataset from the problem domain (PD-OL). It provides a benchmark for evaluating whether the copycat network offers a more effective solution than conventional models trained with limited resources.

The formula for this metric is as follows:

$$\text{Performance Over PD-OL Network} = \frac{\text{MAA of Copycat Network}}{\text{MAA of PD-OL Network}} \times 100 \quad (14)$$

This comparison is particularly valuable when obtaining a large, labeled dataset from the problem domain is impractical. By demonstrating superior performance relative to the PD-OL baseline, the copycat network underscores the potential of leveraging stolen labels and data augmentation techniques [129].

7.5. Domain-Based

Researchers use these measures to assess the impact of defenses or attacks on specific aspects of the targeted domain. In the ML policy task, this measure reports the effect of policy violations on learning outcomes [104].

Researchers use these measures to assess the impact of adversarial attacks or defenses on specific aspects of the targeted domain. In the ML policy task, this measure reports the effect of policy violations on learning outcomes [104].

For example, when applying ML models within AI agents used for portfolio management, additional domain-specific performance metrics are crucial for assessing and optimizing the agent's decisions. These metrics include evaluating the agent's ability to maximize returns while minimizing risks and transaction costs, and they often focus on financial losses, such as the loss of profit due to adversarial actions.

- **Accumulated Portfolio Value (APV):** This metric reflects the change in the portfolio's value over time. It is computed as the ratio of the current portfolio value to its initial value, allowing for a clear view of the agent's performance over a period [104].
- **Maximum Drawdown (MDD):** MDD captures the largest percentage drop from a peak portfolio value to a trough before a new peak is achieved. This metric highlights downside risks and is critical in assessing the resilience of the AI agent under adverse conditions [104].
- **Sharpe Ratio (SR):** The Sharpe Ratio measures risk-adjusted returns by evaluating the excess return per unit of risk (volatility). It is a fundamental metric for understanding how well the agent compensates for the risk taken in achieving its returns [104].

Table 4 provides a comprehensive overview of various attacks and defense strategies applied to DNNs for tasks such as face recognition and text or image classification. It summarizes key research works, highlighting their reported datasets, targeted neural network architectures, and attack and defense performance metrics. The datasets include CIFAR-10, ImageNet, MNIST, and CelebA, while the architectures range from ResNet and VGG to MobileNet and FaceNet. The reported performance includes ASR and accuracy under attack. This table underscores the effectiveness and versatility of attacks and defenses across diverse scenarios, illustrating the vulnerabilities of different DNN models.

Table 4. Comparison of adversarial attack strategies, defense strategies, and their performance on different DNN architectures.

Representative Work	Application	Reported Performance
Bai et al. [96]	Image Classification	CIFAR-10 and ImageNet datasets. ASR: 100% for ResNet-20 and VGG-16.
Wang et al. [97]	Image Classification	CIFAR-10, ImageNet datasets. AvSR: 98.01% for ResNet-50, VGG-16, GoogleNet, MobileNet-v2.
Rakin et al. [98]	Image Classification	CIFAR-10 and ImageNet datasets. ASR: 100% for ResNet-20, VGG-11, ASR: for 99.99%, Accuracy 33% 59% for ResNet-18, ResNet-34, MobileNetV2.
Wang et al. [99]	Image Classification	MNIST, CIFAR-10, and ImageNet datasets. ASR: 100% for Custom CNN, and Inception V3
Tang et al. [101]	Face Recognition	CelebA. ASR: 69.8% for gender transformation while keeping FaceNet's person classification unchanged for the FaceNet model.
Cao et al. [117]	Image Classification	MNIST and CIFAR-10 datasets. ASR: 94.89% AvgQueries: 1695 for CNN model. ASR: 82.3% AvgQueries: 851 for CNN model.
He et al. [102] Kong et al. [108]	Face Recognition Image Classification	CelebA. Accuracy 71.2% for FaceNet model. ImageNet. ASR: Achieved 100% ASR in most configurations. Query Efficiency: Required 73% for Inception-v3, ResNet-50, VGG-16-BN.
Li et al. [115]	Image Classification	MNIST and CIFAR-10 datasets. ASR: 100% within 500 queries. Query Efficiency: Achieved 74.7% for CNN ASR: 100% within 1000 queries for ResNet-50.

Table 4. Cont.

Representative Work	Application	Reported Performance
Hwang et al. [132]	Image Classification, autonomous-driving, face identification, and surveillance systems	MNIST, MNIST Fashion, and CIFAR-10 datasets. Accuracy FGSM: 96.2%, iFGSM: 94.1%, CW: 91.7% Accuracy FGSM: 84.5%, iFGSM: 81.3%, CW: 80.2% Accuracy FGSM: 68.3%, iFGSM: 65.7%, CW: 62.1%. CNN model.
Kuang et al. [114]	Image Classification	ImageNet datasets. ASR: 100 Average Queries: 74,948 for the InceptionV3 model.
Shao et al. [128]	Text Classification	SST-2 (Stanford Sentiment Treebank), IMDB datasets. ASR: 90% for SST-2 BiLSTM/BERT.
Xu et al. [186]	Android Malware	Detection, and virusshare malware dataset. MR: 98.25, Query Efficiency: 1176 for DNN ([200, 100], [200, 200], [10, 10])%.
Chang et al. [187]	Energy-Delay Product (EDP)	CIFAR-10 and ImageNet datasets. 75.9% for MobileNet v1.
Chen et al. [104]	Portfolio Management	TWSE dataset. ASR: Greedy and RL-based attacks reduced portfolio value to 0.92 on average.
Shawn et al. [25]	Malware Classification	EMBER Malware dataset Precision: 99.2%, Recall: 98.2%.
Shao et al. [152]	Text Classification	SST-2, IMDB dataset. SST-2: ASR 96.4% for BiLSTM model) and 100% for BERT model. IMDB: ASR 97% BiLSTM and 98.8 for BERT model%.
Shao et al. [153]	Recognition for Textual Backdoor Defense	SST-2, IMDB dataset. ASR(No Defense): 92.92% (SST-2, BiLSTM) and 100% (IMDB, BERT). ASR (Defense): 6.41% (SST-2, BiLSTM) and 0.20% (IMDB, BERT).
Correia et al. [129]	Facial Expression Recognition (FER)	Test Dataset (TD). Accuracy Target (OD): 88.7%, PD-OL: 82.5%, NPD-SL: 83.0%, PD-SL: 83.4%, NPD+PD-SL: 87.4%. CNN model.

8. Future Research Directions

With AI becoming increasingly prevalent in daily activities, the market for ML in organizations is projected to grow at a compound annual rate of 39.2% by the end of 2027 as organizations work to boost efficiency and enhance customer experience [188]. Despite the many benefits of ML, for example, it has become important to use and replace it with humans in many services because it can find important information from big data or predict important information to use in our daily lives. More recently, most studies have shown potential security threats in ML through the ML process. The high-impact-factor journals highlight the importance of ML security and privacy in the future [189]. This section delves into the latest future direction security issues in ML tasks, encompassing the transferability of adversarial attacks and defense mechanisms, digital forensics, DNNs architectures, and the use of meta-heuristic search strategies.

8.1. Digital Forensics

Indeed, a significant challenge with DNN models is that new attacks can evade defenses, and proactive defenses weaken and become ineffective over time. Digital forensics is one of the reactive approach methods; traceback discovered adversarial examples to find all other adversarial examples. One of the main areas for improving the defense model is incorporating information about crime cases, including how and what happened, and the methods used to prevent or predict the development of an attack scenario. Researchers introduce a new direction to research using digital forensics in the ML process to trace back the source of data that generates misclassification events after the poison attack is

discovered [25]. Researchers should explore different types of attacks on DNNs and other poisoning methods.

Key directions include the creation of real-time monitoring systems and forensic logging mechanisms to detect and record adversarial anomalies, alongside signature-based and statistical anomaly detection techniques to identify known and emerging attack patterns. Additionally, gradient-based and activation pattern analysis can uncover model vulnerabilities, while explainable AI (XAI) techniques and attack attribution methods can enhance the understanding and tracing of adversarial manipulations. To improve incident response, standardized forensic pipelines and open source toolkits should be developed, complemented by platforms for threat intelligence sharing to disseminate knowledge about new attacks and defenses. On the legal and ethical front, establishing regulatory compliance guidelines, promoting ethical AI practices (e.g., transparency, accountability, and fairness), and implementing forensic auditing mechanisms will ensure the security and robustness of DNNs. Together, these efforts will pave the way for a multifaceted approach to safeguarding DNNs against adversarial threats, fostering resilience and trust in AI systems.

8.2. Identify Trigger

The initial approach to creating a trigger in a poison attack involved adding a pattern to the sample. The process became more complex when physical objects, such as eyeglasses or a tattoo outline, were used to generate triggers, making them more difficult to detect. Finally, a group of researchers crafted a particular function to modify the picture called Trojan-advertised attack [79,123], making the identification trigger challenging. Identifying the location of the trigger and the techniques that generate the trigger can improve defense mechanisms. In future research, the DNN model needs to develop robust methods to identify triggers responsible for failures in DNN models during the training and testing phases.

To identify and mitigate triggers in DNNs, researchers can employ a combination of methodologies and frameworks. An approach involves analyzing neuron activation patterns to detect unusual behavior caused by triggers, using clustering (e.g., k-means) or outlier detection algorithms (e.g., Isolation Forest) to flag suspicious training samples. Another strategy is to reverse-engineer potential triggers by optimizing inputs to induce misclassification, leveraging tools such as neural cleanse. Explainable AI (XAI) techniques, such as LIME, SHAP, or attention mechanisms, can be used to identify input regions that contribute to predictions, while visualization methods like heat maps or feature visualization help uncover trigger patterns. Additionally, the generation of adversarial triggers using GAN or similar models can harden DNNs by incorporating adversarial triggers during training. To standardize the evaluation, benchmark datasets with known triggers can be created, allowing the development and testing of robust detection methods. These combined approaches provide a comprehensive framework for identifying triggers and improving the security of DNN.

8.3. Deep Neural Network Architectures

Several research studies on adversarial attacks and defense mechanisms propose new approaches to develop more resilient DNNs to counter adversarial threats. The authors of [114] highlight the vulnerability of DNNs structures and suggest enhancing network security through structural design improvements. Another research effort in [190] delved into the robustness of DNNs against decision-based patch attacks, suggesting that these findings could guide the development of robust vision models. The work in [97] identified security challenges to the robustness of the AI model and underscored the importance

of designing more secure and resilient models. The conclusions in [98] offer analytical approaches and strategies for building DNN models resistant to advanced attacks. Finally, it points out that CNNs tend to prioritize texture over shape, unlike human visual systems, and recommends further investigation from a Fourier analysis perspective to bridge the gap between human perception and CNN, with the aim of a more robust AI [191].

Although distillation is a method for redesigning the neural network after training to remove unused nodes that can be sources of adversarial attacks to increase network security [192], it can also reduce accuracy. In the future, several methods can be used to report samples that attempt to use nodes in suspicious operations, making it suitable for reactive approaches rather than the network distillation method.

8.4. Federated Learning (FL)

FL is an ML framework in which multiple clients train a model coordinated by a central server while ensuring that the training data remain distributed and not centralized. It can be used to integrate all defense mechanisms into one model that can vigorously protect DNNs from attacks.

Adversarial training can be integrated into FL by generating adversarial examples during training, either locally by clients or centrally by the server, to improve the robustness of the model. Blockchain technology can be leveraged to ensure secure and transparent aggregation of model updates, creating a trust FL environment resistant to tampering as demonstrated in the [193]; this framework underscores blockchain's robustness against tampering and its ability to provide secure identity management in sensitive systems. Drawing parallels, the blockchain could be utilized as a decentralized and immutable ledger for managing model parameters, training data, and access logs in DNN systems. Such an approach would not only enhance defense against adversarial attacks but also enhance transparency and trustworthiness in AI applications. Future research could explore the integration of blockchain-based solutions into adversarial defense mechanisms to protect DNNs against data poisoning and model manipulation. A centralized monitoring framework can be designed to assess the global model's security and accuracy at each aggregation round, enabling real-time threat detection. In addition, dynamic defense mechanisms can be developed to adaptively select the most effective strategies based on the type and severity of the detected attacks. In specific applications such as autonomous vehicles, FL can facilitate vehicle-to-vehicle communication to improve traffic safety, while defense mechanisms can protect recognition systems from adversarial attacks.

8.5. Transferability of Adversarial Attacks and Defense Mechanisms

Since various DL models are based on comparable architectures and learning algorithms, applying successful adversarial attack and defense strategies from one model to others is possible, even if they have distinct architectures or learning algorithms. In ref. [194], the author demonstrates that their adversarial attack strategy can generalize across different architectures, but only to some extent.

One of the challenges in adversarial attacks and defense mechanisms is the need for standardized evaluation metrics and benchmarks for defense methods. This is crucial to prevent attack transferability from diverse domains or between various tasks, including NLP, speech recognition, and RL. A defense benchmark to avoid transferability attacks would be valuable for researchers and practitioners to evaluate and validate their proposed attack methods in a controlled and consistent manner. It could cover a broad range of real-world scenarios and include metrics to assess the impact of defense measures on DNN models. By applying any proposed defense mechanism across multiple domains and tasks in DNNs, we can evaluate its effectiveness against different attacks

and report the accuracy to demonstrate how well the defense mitigates various attacks across diverse domains and measures the generalizability of the defense mechanism across multiple domains.

8.6. Explore New Domains

The article in [6], acknowledged as the pioneering work that uncovered the surprising vulnerabilities of neural networks to attacks through an image classification task experiment, has led most subsequent studies to explore various adversarial attack and defense mechanisms within this field. This pattern has inspired the author of [15] to emphasize the importance of examining adversarial strategies and protective measures in broader applications.

8.7. Meta-Heuristic Search Strategies

Attackers and defenders in adversarial DNNs face NP-hard optimization problems, meaning they can not be solved in polynomial time, making them computationally intensive and complex. Most of the research in this area has traditionally relied on gradient descent, greedy algorithms, and iterative procedures to navigate the search space for optimal solutions. However, in a new research direction detailed in [117], the author introduces the use of an ABC algorithm, a relatively new metaheuristic technique, as a strategic tool to outmaneuver DNN-based image classification systems. This strategy mimics the foraging behavior of bees to provide a more effective and possibly more efficient way to generate adversarial attacks that can successfully bypass DNN defenses.

9. Conclusions

This article provides a comprehensive review of defense and adversarial attack mechanisms targeting DNNs, highlighting significant advancements and challenges over the past decade and offering new insights for a deeper understanding of adversarial learning with DNNs. We cover adversarial attacks across various ML tasks and discuss prevalent defense strategies. Current defense mechanisms, such as adversarial training, model verification, and secure aggregation, have progressed, but they often need to be improved by generalization and resource efficiency limitations. Furthermore, we have provided an extensive overview of potential attack scenarios and vulnerabilities within DNNs, emphasizing the need for future research to explore diverse defense mechanisms and develop a balance between proactive and reactive defenses, drawing upon the latest techniques, such as digital forensics, security, and forensics by design. As DNNs evolve, ensuring their security in real-world applications remains an open and critical challenge, calling for further interdisciplinary research in theoretical and practical domains.

Author Contributions: Conceptualization, A.A. (Abdulruhman Abomakhelb), K.A.J., A.G.B., A.A. (Abdulraqeb Alhammadi) and A.M.A.; Methodology, A.A. (Abdulruhman Abomakhelb), K.A.J., A.G.B., A.A. (Abdulraqeb Alhammadi) and A.M.A.; Writing—original draft, A.A. (Abdulruhman Abomakhelb), K.A.J., A.G.B., A.A. (Abdulraqeb Alhammadi) and A.M.A.; Writing—review and editing, A.A. (Abdulruhman Abomakhelb), K.A.J., A.G.B., A.A. (Abdulraqeb Alhammadi) and A.M.A.; Supervision, K.A.J. and A.G.B.; Funding acquisition, A.A. (Abdulraqeb Alhammadi). All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by Universiti Teknologi Malaysia through the Professional Development Research University under Grant Q.K130000.21A2.07E08.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No data available.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

All the abbreviations in this work have been first introduced in the text. For convenience, the abbreviations are also listed below:

Abbreviation	Full Form
AI	Artificial Intelligence
DNNs	Deep Neural Networks
ML	Machine Learning
DL	Deep Learning
SVMs	Support Vector Machines
RNNs	Recurrent Neural Networks
RL	Reinforcement Learning
PGD	Projected Gradient Descent
C&W	Carlini and Wagner
HSJ	HopSkipJump
ANNs	Artificial Neural Networks
ADR	Average Detection Rate
DL App	Deep Learning Application
NLP	Natural Language Processing
PRN	Perturbation Rectifying Network
DCN	Deep Contractive Networks
HVS	Human Vision System
DRAM	Dynamic Random Access Memory
FGSM	Fast Gradient Sign Method
GAN	Generative Adversarial Network
FL	Federated Learning
AvSR	Average Success Rate
CNNs	Convolutional Neural Networks
MR	Misclassification Rate
JND	Just Noticeable Distortion
TN	True Negatives
GRU	Gated Recurrent Unit
SR	Sharpe Ratio
MDD	Maximum Drawdown
APV	Accumulated Portfolio Value
Avgquery	Average Amount of Queries
TP	True Positives
FN	False Negatives
TFR	Target Fooling Rate
TWSE	Taiwan Stock Exchange
ASR	Attack Success Rate
SST-2	Stanford Sentiment Treebank
SVHN	Street View House Numbers
EMRC	Robust Mode Connectivity
AAD	Attention-based Adversarial Defense
FP	False Positives
BIM	Basic Iterative Method
DNR	Deep Neural Rejection
EBD	Entropy-based Detector
AoA	Attack on Attention
ABCAttack	Artificial Bee Colony Attack
SA-ES	Subspace Activation Evolution Strategy

SMGEA	Serial Minigroup Ensemble Attack
DSNE	Dual-Stage Network Erosion
MLAE-DE	Multilabel Adversarial Examples—Differential Evolution
TG	Transformed Gradient
GAA	Generative Adversarial Attack
GAN	Generative Adversarial Network
T-BFA	Targeted Bit-Flip Adversarial Weight Attack
ADSAttack	Adversarial Distribution Searching-Driven Attack
UAPs	Universal Adversarial Perturbations
EAD	Elastic-Net Attacks to DNNs
LSTM	Long Short-Term Memory
DRL	Deep Reinforcement Learning

References

1. Hinton, G. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Process. Mag.* **2012**, *29*, 82–97. [[CrossRef](#)]
2. Lecun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [[CrossRef](#)] [[PubMed](#)]
3. He, X.; Zhao, K.; Chu, X. AutoML: A survey of the state-of-the-art. *Knowl.-Based Syst.* **2021**, *212*, 106622. [[CrossRef](#)]
4. Durbha, K.S.; Amuru, S. AutoML models for wireless signals classification and their effectiveness against adversarial attacks. In Proceedings of the 2022 14th International Conference on Communication Systems & NETWORKS (COMSNETS), Bangalore, India, 4–8 January 2022. [[CrossRef](#)]
5. He, Y.; Lin, J.; Liu, Z.; Wang, H.; Li, L.J.; Han, S. AMC: AutoML for Model Compression and Acceleration on Mobile Devices. In Proceedings of the 14th International Conference on Communication Systems & NETWORKS (COMSNETS), Bangalore, India, 4–8 January 2022; pp. 1–6.
6. Szegedy, C.; Wilson, A.G.; Zheng, Z.; Tsipras, D.; Dong, Y.; Rattner, J.; Sinha, A.; Bengio, S. Intriguing properties of neural networks. *arXiv* **2013**, arXiv:1312.6199.
7. Yang, Z.; Li, B.; Chen, P.Y.; Song, D. Characterizing Audio Adversarial Examples Using Temporal Dependency. *arXiv* **2018**, arXiv:1809.10875.
8. Joseph, A.D. *Adversarial Machine Learning*; Cambridge University Press: Cambridge, UK, 2018.
9. Biggio, B.; Roli, F. Wild Patterns: Ten Years After the Rise of Adversarial Machine Learning. *Pattern Recognit.* **2018**, *84*, 317–331. [[CrossRef](#)]
10. Dalvi, N.; Domingos, P.; Mausam, S.; Verma, D. Adversarial Classification. In Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, WA, USA, 22–25 August 2004; pp. 99–108.
11. Lowd, D.; Meek, C. Good Word Attacks on Statistical Spam Filters. In Proceedings of the 2005 Conference on Email and Anti-Spam (CEAS), Stanford, CA, USA, 21–22 July 2005; Volume 2005.
12. Biggio, B.; Nelson, B.; Laskov, P. Poisoning attacks against support vector machines. *arXiv* **2012**, arXiv:1206.6389.
13. Biggio, B.; Corona, I.; Maiorca, D.; Nelson, B.; Šrndić, N.; Laskov, P.; Roli, F. Evasion attacks against machine learning at test time. In Proceedings of the Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD, Prague, Czech Republic, 23–27 September 2013; Springer: Berlin/Heidelberg, Germany, 2013; pp. 387–402.
14. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. *arXiv* **2014**, arXiv:1412.6572.
15. Khamaiseh, S.Y.; Bagagem, D.; Al-Alaj, A.; Mancino, M.; Alomari, H.W. Adversarial deep learning: A survey on adversarial attacks and defense mechanisms on image classification. *IEEE Access* **2022**, *10*, 102266–102291. [[CrossRef](#)]
16. AlSobeh, A.; Franklin, A.; Woodward, B.; Porche, M.; Siegelman, J. Unmasking Media Illusion: Analytical Survey of Deepfake Video Detection and Emotional Insights. *Issues Inf. Syst.* **2024**, *25*, 96–112.
17. Carlini, N.; Wagner, D. Towards Evaluating the Robustness of Neural Networks. In Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–26 May 2017; pp. 39–57. [[CrossRef](#)]
18. Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; Vladu, A. Towards Deep Learning Models Resistant to Adversarial Attacks. In Proceedings of the International Conference on Learning Representations (ICLR), Vancouver, BC, Canada, 30 April–3 May 2018.
19. Athalye, A.; Carlini, N.; Wagner, D. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. In Proceedings of the 35th International Conference on Machine Learning (ICML), Stockholm, Sweden, 10–15 July 2018.
20. Gu, T.; Dolan-Gavitt, B.; Garg, S. BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain. In Proceedings of the Machine Learning and Computer Security Workshop (NeurIPS), Long Beach, CA, USA, 8 December 2017.
21. Li, Y.; Lyu, L.; Bai, S.; Li, B.; Liang, Y. Backdoor Learning: A Survey. *arXiv* **2020**, arXiv:2007.08745. [[CrossRef](#)] [[PubMed](#)]

22. Bagdasaryan, E.; Veit, A.; Hua, Y.; Estrin, D.; Shmatikov, V. How To Backdoor Federated Learning. In Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS), Online, 26–28 August 2020.
23. Tramèr, F.; Zhang, F.; Juels, A.; Reiter, M.K.; Ristenpart, T. Stealing Machine Learning Models via Prediction APIs. In Proceedings of the 25th USENIX Security Symposium (USENIX Security 2016), Austin, TX, USA, 10–12 August 2016.
24. Shokri, R.; Stronati, M.; Song, C.; Shmatikov, V. Membership Inference Attacks Against Machine Learning Models. In Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–26 May 2017; pp. 3–18.
25. Shan, S.; Bhagoji, A.N.; Zheng, H.; Zhao, B.Y. Poison forensics: Traceback of data poisoning attacks in neural networks. In Proceedings of the 31st USENIX Security Symposium (USENIX Security 22), Boston, MA, USA, 10–12 August 2022; pp. 3575–3592.
26. Abraham, A. Artificial neural networks. In *Handbook of Measuring System Design*; Wiley: Hoboken, NJ, USA, 2005.
27. Sadeghi, K.; Banerjee, A.; Gupta, S.K. A system-driven taxonomy of attacks and defenses in adversarial machine learning. *IEEE Trans. Emerg. Top. Comput. Intell.* **2020**, *4*, 450–467. [[CrossRef](#)] [[PubMed](#)]
28. Ilahi, I.; Usama, M.; Qadir, J.; Janjua, M.U.; Al-Fuqaha, A.; Hoang, D.T.; Niyato, D. Challenges and countermeasures for adversarial attacks on deep reinforcement learning. *IEEE Trans. Artif. Intell.* **2021**, *3*, 90–109. [[CrossRef](#)]
29. Mohus, M.L.; Li, J. Adversarial Robustness in Unsupervised Machine Learning: A Systematic Review. *arXiv* **2023**, arXiv:2306.00687.
30. Sun, S.; Cao, Z.; Zhu, H.; Zhao, J. A survey of optimization methods from a machine learning perspective. *IEEE Trans. Cybern.* **2019**, *50*, 3668–3681. [[CrossRef](#)]
31. Apostolidis, K.D.; Papakostas, G.A. A survey on adversarial deep learning robustness in medical image analysis. *Electronics* **2021**, *10*, 2132. [[CrossRef](#)]
32. Ding, J.; Xu, Z. Adversarial attacks on deep learning models of computer vision: A survey. In Proceedings of the Algorithms and Architectures for Parallel Processing: 20th International Conference, ICA3PP 2020, New York, NY, USA, 2–4 October 2020; Proceedings, Part III; Springer International Publishing: Berlin/Heidelberg, Germany, 2020; pp. 396–408. [[CrossRef](#)]
33. Zhang, W.E.; Sheng, Q.Z.; Alhazmi, A.; Li, C. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Trans. Intell. Syst. Technol.* **2020**, *11*, 1–41. [[CrossRef](#)]
34. Akhtar, N.; Mian, A. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access* **2018**, *6*, 14410–14430. [[CrossRef](#)]
35. Wang, J.; Wang, C.; Lin, Q.; Luo, C.; Wu, C.; Li, J. Adversarial attacks and defenses in deep learning for image recognition: A survey. *Neurocomputing* **2022**, *514*, 162–181. [[CrossRef](#)]
36. Macas, M.; Wu, C.; Fuertes, W. Adversarial examples: A survey of attacks and defenses in deep learning-enabled cybersecurity systems. *Expert Syst. Appl.* **2023**, *238*, 122223. [[CrossRef](#)]
37. Mengara, O.; Avila, A.; Falk, T.H. Backdoor Attacks to Deep Neural Networks: A Survey of the Literature, Challenges, and Future Research Directions. *IEEE Access* **2024**, *12*, 29004–29023. [[CrossRef](#)]
38. Bhanushali, A.R.; Mun, H.; Yun, J. Adversarial Attacks on Automatic Speech Recognition (ASR): A Survey. *IEEE Access* **2024**, *12*, 88279–88302. [[CrossRef](#)]
39. Costa, J.C.; Roxo, T.; Proença, H.; Inácio, P.R.M. How deep learning sees the world: A survey on adversarial attacks & defenses. *IEEE Access* **2024**, *12*, 61113–61136.
40. Hoang, V.T.; Ergu, Y.A.; Nguyen, V.L.; Chang, R.G. Security risks and countermeasures of adversarial attacks on AI-driven applications in 6G networks: A survey. *J. Netw. Comput. Appl.* **2024**, *232*, 104031. [[CrossRef](#)]
41. Moosavi-Dezfooli, S.M.; Fawzi, A.; Frossard, P. Deepfool: A Simple and Accurate Method to Fool Deep Neural Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 2574–2582.
42. Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychol. Rev.* **1958**, *65*, 386–408. [[CrossRef](#)]
43. Nazzal, J.M.; El-Emary, I.M.; Najim, S.A.; Ahliyya, A. Multilayer Perceptron Neural Network (MLPs) for Analyzing the Properties of Jordan Oil Shale. *World Appl. Sci. J.* **2008**, *5*, 546–552.
44. Gome, L. Machine-Learning Maestro Michael Jordan on the Delusions of Big Data and Other Huge Engineering Efforts. *IEEE Spectr.* **2016**.
45. Min, S.; Lee, B.; Yoon, S. Deep learning in bioinformatics. *Briefings Bioinform.* **2017**, *18*, 851–869. [[CrossRef](#)]
46. Roopak, M.; Tian, G.Y.; Chambers, J. Deep learning models for cyber security in IoT networks. In Proceedings of the 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 7–9 January 2019; pp. 0452–0457. [[CrossRef](#)]
47. Sun, P.; Liu, P.; Li, Q.; Liu, C.; Lu, X.; Hao, R.; Chen, J. DL-IDS: Extracting Features Using CNN-LSTM Hybrid Network for Intrusion Detection System. *Secur. Commun. Netw.* **2020**, *2020*, 8890306. [[CrossRef](#)]
48. Abbaspour, S.; Fotouhi, F.; Sedaghatbaf, A.; Fotouhi, H.; Vahabi, M.; Linden, M. A comparative analysis of hybrid deep learning models for human activity recognition. *Sensors* **2020**, *20*, 5707. [[CrossRef](#)]

49. Fang, W.; Chen, Y.; Xue, Q. Survey on research of RNN-based spatio-temporal sequence prediction algorithms. *J. Big Data* **2021**, *3*, 97. [[CrossRef](#)]
50. Xiao, J.; Zhou, Z. Research progress of RNN language model. In Proceedings of the 2020 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, 27–29 June 2020; pp. 1285–1288.
51. Shewalkar, A.; Nyavanandi, D.; Ludwig, S.A. Performance evaluation of deep neural networks applied to speech recognition: RNN, LSTM and GRU. *J. Artif. Intell. Soft Comput. Res.* **2019**, *9*, 235–245. [[CrossRef](#)]
52. Apaydin, H.; Feizi, H.; Sattari, M.T.; Colak, M.S.; Shamshirband, S.; Chau, K.W. Comparative analysis of recurrent neural network architectures for reservoir inflow forecasting. *Water* **2020**, *12*, 1500. [[CrossRef](#)]
53. Graves, A.; Graves, A. Long short-term memory. In *Supervised Sequence Labelling with Recurrent Neural Networks*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 37–45.
54. Graves, A.; Liwicki, M.; Fernández, S.; Bertolami, R.; Bunke, H.; Schmidhuber, J. A novel connectionist system for unconstrained handwriting recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, *31*, 855–868. [[CrossRef](#)] [[PubMed](#)]
55. Fortunato, M.; Blundell, C.; Vinyals, O. Bayesian recurrent neural networks. *arXiv* **2017**, arXiv:1704.02798.
56. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv* **2014**, arXiv:1412.3555.
57. Chen, J.X.; Jiang, D.M.; Zhang, Y.N. A hierarchical bidirectional GRU model with attention for EEG-based emotion classification. *IEEE Access* **2019**, *7*, 118530–118540. [[CrossRef](#)]
58. Svozil, D.; Kvasnicka, V.; Pospichal, J. Introduction to multi-layer feed-forward neural networks. *Chemom. Intell. Lab. Syst.* **1997**, *39*, 43–62. [[CrossRef](#)]
59. Kohonen, T. *Self-Organizing Maps*, 3rd ed.; Springer Series in Information Sciences; Springer: Berlin/Heidelberg, Germany, 2001; Volume 30. [[CrossRef](#)]
60. Pinaya, W.H.L.; Vieira, S.; Garcia-Dias, R.; Mechelli, A. Autoencoders. In *Machine Learning*; Academic Press: Cambridge, MA, USA, 2020; pp. 193–208.
61. Stoer, J.; Bulirsch, R. *Introduction to Numerical Analysis*; Springer: Berlin/Heidelberg, Germany, 2002; Chapter Systems of Linear Equations, pp. 190–288.
62. Mart, R.; Pardalos, P.M.; Resende, M.G.C. *Handbook of Heuristics*; Springer Publishing Company Incorporated: Berlin/Heidelberg, Germany, 2018.
63. Ezugwu, A.E.; Shukla, A.K.; Nath, R.; Akinyelu, A.A.; Agushaka, J.O.; Chiroma, H.; Muhuri, P.K. Metaheuristics: A comprehensive overview and classification along with bibliometric analysis. *Artif. Intell. Rev.* **2021**, *54*, 4237–4316. [[CrossRef](#)]
64. Sen, S.; Weiss, G. Learning in multiagent systems. In *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*; MIT Press: Cambridge, MA, USA, 1999; pp. 259–298.
65. Kaelbling, L.P.; Littman, M.L.; Moore, A.W. Reinforcement Learning: A Survey. *J. Artif. Intell. Res.* **1996**, *4*, 237–285. [[CrossRef](#)]
66. Abu-Mostafa, Y.S.; Magdon-Ismael, M.; Lin, H.T. *Learning from Data*; AMLBook: New York, NY, USA, 2012; Volume 4, p. 4.
67. Hinton, G.; Sejnowski, T.J. *Unsupervised Learning: Foundations of Neural Computation*; MIT Press: Cambridge, MA, USA, 1999.
68. Lowd, D.; Meek, C. Adversarial Learning. In Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, Chicago, IL, USA, 21–24 August 2005; pp. 641–647. [[CrossRef](#)]
69. Wang, X.; Li, J.; Kuang, X.; Tan, Y.A.; Li, J. The security of machine learning in an adversarial setting: A survey. *J. Parallel Distrib. Comput.* **2019**, *130*, 12–23. [[CrossRef](#)]
70. Biggio, B.; Fumera, G.; Roli, F. Security evaluation of pattern classifiers under attack. *IEEE Trans. Knowl. Data Eng.* **2013**, *26*, 984–996. [[CrossRef](#)]
71. Torr, P. Demystifying the threat modeling process. *IEEE Secur. Priv.* **2005**, *3*, 66–70. [[CrossRef](#)]
72. Miller, D.J.; Xiang, Z.; Kesidis, G. Adversarial learning targeting deep neural network classification: A comprehensive review of defenses against attacks. *Proc. IEEE* **2020**, *108*, 402–433. [[CrossRef](#)]
73. Santhanam, V.; Davis, L.S. A generic improvement to deep residual networks based on gradient flow. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *31*, 2490–2499. [[CrossRef](#)]
74. Zhong, Y.; Deng, W. Towards transferable adversarial attack against deep face recognition. *IEEE Trans. Inf. Forensics Secur.* **2020**, *16*, 1452–1466. [[CrossRef](#)]
75. Alrawashdeh, K.; Goldsmith, S. Defending deep learning based anomaly detection systems against white-box adversarial examples and backdoor attacks. In Proceedings of the 2020 IEEE International Symposium on Technology and Society (ISTAS), Tempe, AZ, USA, 12–15 November 2020; pp. 294–301.
76. Carlini, N.; Wagner, D. Adversarial examples are not easily detected: Bypassing ten detection methods. In Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, Dallas, TX, USA, 3 November 2017; pp. 3–14.
77. Rakin, A.S.; He, Z.; Fan, D. Bit-flip attack: Crushing neural network with progressive bit search. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1211–1220.

78. Yao, F.; Rakin, A.S.; Fan, D. DeepHammer: Depleting the intelligence of deep neural networks through targeted chain of bit flips. In Proceedings of the 29th USENIX Security Symposium (USENIX Security 20), Boston, MA, USA, 12–14 August 2020; pp. 1463–1480.
79. Rakin, A.S.; He, Z.; Fan, D. Tbt: Targeted Neural Network Attack with Bit Trojan. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 13198–13207.
80. Yan, M.; Fletcher, C.W.; Torrellas, J. Cache Telepathy: Leveraging Shared Resource Attacks to Learn DNN Architectures. In Proceedings of the 29th USENIX Security Symposium (USENIX Security 20), Boston, MA, USA, 12–14 August 2020; pp. 2003–2020.
81. Xiang, Y.; Chen, Z.; Chen, Z.; Fang, Z.; Hao, H.; Chen, J.; Yang, X. Open DNN Box by Power Side-Channel Attack. *IEEE Trans. Circuits Syst. II Express Briefs* **2020**, *67*, 2717–2721. [[CrossRef](#)]
82. Yu, H.; Ma, H.; Yang, K.; Zhao, Y.; Jin, Y. DeepEM: Deep Neural Networks Model Recovery through EM Side-Channel Information Leakage. In Proceedings of the 2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), San Jose, CA, USA, 7–11 December 2020; pp. 209–218. [[CrossRef](#)]
83. Das, D.; Golder, A.; Danial, J.; Ghosh, S.; Raychowdhury, A.; Sen, S. X-DeepSCA: Cross-device deep learning side channel attack. In Proceedings of the 56th Annual Design Automation Conference 2019, Las Vegas, NV, USA, 2–6 June 2019; pp. 1–6. [[CrossRef](#)]
84. Chen, J.; Jordan, M.I.; Wainwright, M.J. Hopskipjumpattack: A query-efficient decision-based attack. In Proceedings of the 2020 IEEE Symposium on Security and Privacy (SP), Francisco, CA, USA, 18–20 May 2020; pp. 1277–1294.
85. Biggio, B.; Nelson, B.; Laskov, P. Support vector machines under adversarial label noise. In Proceedings of the Asian Conference on Machine Learning, Taoyuan, Taiwan, 13–15 November 2011; pp. 97–112.
86. López-Valcarce, R.; Romero, D. Design of data-injection adversarial attacks against spatial field detectors. In Proceedings of the 2016 IEEE Statistical Signal Processing Workshop (SSP), Palma de Mallorca, Spain, 26–29 June 2016; pp. 1–5.
87. Chacon, H.; Silva, S.; Rad, P. Deep learning poison data attack detection. In Proceedings of the 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI), Portland, OR, USA, 4–6 November 2019; pp. 971–978.
88. Jiang, W.; Li, H.; Liu, S.; Luo, X.; Lu, R. Poisoning and evasion attacks against deep learning algorithms in autonomous vehicles. *IEEE Trans. Veh. Technol.* **2020**, *69*, 4439–4449. [[CrossRef](#)]
89. Agrawal, G.; Kaur, A.; Myneni, S. A review of generative models in generating synthetic attack data for cybersecurity. *Electronics* **2024**, *13*, 322. [[CrossRef](#)]
90. Khan, Z.; Chowdhury, M.; Khan, S.M. A Hybrid Defense Method Against Adversarial Attacks on Traffic Sign Classifiers in Autonomous Vehicles. *arXiv* **2022**, arXiv:2205.01225.
91. Chen, P.Y.; Sharma, Y.; Zhang, H.; Yi, J.; Hsieh, C.J. Ead: Elastic-net attacks to deep neural networks via adversarial examples. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32.
92. Beck, A.; Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.* **2009**, *2*, 183–202. [[CrossRef](#)]
93. Hirano, H.; Takemoto, K. Simple iterative method for generating targeted universal adversarial perturbations. *Algorithms* **2020**, *13*, 268. [[CrossRef](#)]
94. Brendel, W.; Rauber, J.; Kümmeler, M.; Ustyuzhaninov, I.; Bethge, M. Accurate, reliable and fast robustness evaluation. In Proceedings of the Advances in Neural Information Processing Systems. NeurIPS, Vancouver, BC, Canada, 8–14 December 2019; Volume 32.
95. Ghiasi, A.; Shafahi, A.; Goldstein, T. Breaking certified defenses: Semantic adversarial examples with spoofed robustness certificates. *arXiv* **2020**, arXiv:2003.08937.
96. Bai, J.; Wu, B.; Li, Z.; Xia, S.T. Versatile weight attack via flipping limited bits. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 13653–13665. [[CrossRef](#)]
97. Wang, H.; Zhu, C.; Cao, Y.; Zhuang, Y.; Li, J.; Chen, X. ADSAttack: An Adversarial Attack Algorithm via Searching Adversarial Distribution in Latent Space. *Electronics* **2023**, *12*, 816. [[CrossRef](#)]
98. Rakin, A.S.; He, Z.; Li, J.; Yao, F.; Chakrabarti, C.; Fan, D. T-bfa: Targeted bit-flip adversarial weight attack. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 7928–7939. [[CrossRef](#)]
99. Wang, Z.; Song, M.; Zheng, S.; Zhang, Z.; Song, Y.; Wang, Q. Invisible adversarial attack against deep neural networks: An adaptive penalization approach. *IEEE Trans. Dependable Secur. Comput.* **2019**, *18*, 1474–1488. [[CrossRef](#)]
100. Kurakin, A.; Goodfellow, I.; Bengio, S. Adversarial examples in the physical world. In *Artificial Intelligence Safety and Security*; Chapman and Hall/CRC: Boca Raton, FL, USA, 2018; pp. 99–112.
101. Tang, S.; Huang, X.; Chen, M.; Sun, C.; Yang, J. Adversarial attack type I: Cheat classifiers by significant changes. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *43*, 1100–1109. [[CrossRef](#)]
102. He, S.; Wang, R.; Liu, T.; Yi, C.; Jin, X.; Liu, R.; Zhou, W. Type-I generative adversarial attack. *IEEE Trans. Dependable Secur. Comput.* **2022**, *20*, 2593–2606. [[CrossRef](#)]
103. Wan, C.; Huang, F.; Zhao, X. Average gradient-based adversarial attack. *IEEE Trans. Multimed.* **2023**, *25*, 9572–9585. [[CrossRef](#)]

104. Chen, Y.Y.; Chen, C.T.; Sang, C.Y.; Yang, Y.C.; Huang, S.H. Adversarial attacks against reinforcement learning-based portfolio management strategy. *IEEE Access* **2021**, *9*, 50667–50685. [[CrossRef](#)]
105. He, Z.; Duan, Y.; Zhang, W.; Zou, J.; He, Z.; Wang, Y.; Pan, Z. Boosting adversarial attacks with transformed gradient. *Comput. Secur.* **2022**, *118*, 102720. [[CrossRef](#)]
106. Wang, X.; Chen, K.; Ma, X.; Chen, Z.; Chen, J.; Jiang, Y.G. AdvQDet: Detecting Query-Based Adversarial Attacks with Adversarial Contrastive Prompt Tuning. In Proceedings of the 32nd ACM International Conference on Multimedia, Melbourne, Australia, 28 October–1 November 2024; pp. 6212–6221.
107. Rastrigin, L. The convergence of the random search method in the extremal control of a many parameter system. *Autom. Remote Control* **1963**, *24*, 1337–1342.
108. Andriushchenko, M.; Croce, F.; Flammarion, N.; Hein, M. Square Attack: A Query-Efficient Black-Box Adversarial Attack via Random Search. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; pp. 484–501.
109. Su, J.; Vargas, D.V.; Sakurai, K. One pixel attack for fooling deep neural networks. *IEEE Trans. Evol. Comput.* **2019**, *23*, 828–841. [[CrossRef](#)]
110. Kong, L.; Luo, W.; Zhang, H.; Liu, Y.; Shi, Y. Evolutionary multilabel adversarial examples: An effective black-box attack. *IEEE Trans. Artif. Intell.* **2022**, *4*, 562–572. [[CrossRef](#)]
111. Cai, K.; Zhu, X.; Hu, Z.L. Black-box reward attacks against deep reinforcement learning based on successor representation. *IEEE Access* **2022**, *10*, 51548–51560. [[CrossRef](#)]
112. Duan, Y.; Zou, J.; Zhou, X.; Zhang, W.; He, Z.; Zhan, D.; Pan, Z. Adversarial attack via dual-stage network erosion. *Comput. Secur.* **2022**, *122*, 102888. [[CrossRef](#)]
113. Che, Z.; Borji, A.; Zhai, G.; Ling, S.; Li, J.; Min, X.; Le Callet, P. SMGEA: A new ensemble adversarial attack powered by long-term gradient memories. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *33*, 1051–1065. [[CrossRef](#)]
114. Kuang, X.; Liu, H.; Wang, Y.; Zhang, Q.; Zhang, Q.; Zheng, J. A CMA-ES-Based Adversarial Attack on Black-Box Deep Neural Networks. *IEEE Access* **2019**, *7*, 172938–172947. [[CrossRef](#)]
115. Li, Z.; Cheng, H.; Cai, X.; Zhao, J.; Zhang, Q. Sa-es: Subspace activation evolution strategy for black-box adversarial attacks. *IEEE Trans. Emerg. Top. Comput. Intell.* **2022**, *7*, 780–790. [[CrossRef](#)]
116. Wei, X.; Guo, Y.; Li, B. Black-box adversarial attacks by manipulating image attributes. *Inf. Sci.* **2021**, *550*, 285–296. [[CrossRef](#)]
117. Cao, H.; Si, C.; Sun, Q.; Liu, Y.; Li, S.; Gope, P. Abcattack: A gradient-free optimization black-box attack for fooling deep image classifiers. *Entropy* **2022**, *24*, 412. [[CrossRef](#)] [[PubMed](#)]
118. Wang, Z.M.; Gu, M.T.; Hou, J.H. Sample based fast adversarial attack method. *Neural Process. Lett.* **2019**, *50*, 2731–2744. [[CrossRef](#)]
119. Chen, S.; He, Z.; Sun, C.; Yang, J.; Huang, X. Universal adversarial attack on attention and the resulting dataset Damagenet. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *44*, 2188–2197. [[CrossRef](#)]
120. Lu, Y.; Kamath, G.; Yu, Y. Indiscriminate data poisoning attacks on neural networks. *arXiv* **2022**, arXiv:2204.09092.
121. Chen, B.; Carvalho, W.; Baracaldo, N.; Ludwig, H.; Edwards, B.; Lee, T.; Srivastava, B. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv* **2018**, arXiv:1811.03728.
122. Havens, A.; Jiang, Z.; Sarkar, S. Online robust policy learning in the presence of unknown adversaries. In Proceedings of the 32nd International Conference on Neural Information Processing Systems (NIPS’18). Curran Associates Inc., Red Hook, NY, USA, 2018; pp. 9938–9948.
123. Ding, S.; Tian, Y.; Xu, F.; Li, Q.; Zhong, S. Trojan attack on deep generative models in autonomous driving. In Proceedings of the Security and Privacy in Communication Networks: 15th EAI International Conference, SecureComm 2019, Orlando, FL, USA, 23–25 October 2019; Proceedings, Part I; Springer International Publishing: Berlin/Heidelberg, Germany, 2019; pp. 299–318.
124. Shafahi, A.; Huang, W.R.; Najibi, M.; Suci, O.; Studer, C.; Dumitras, T.; Goldstein, T. Poison frogs! targeted clean-label poisoning attacks on neural networks. In Proceedings of the Advances in Neural Information Processing Systems. NeurIPS, Montreal, ON, USA, 3–8 December 2018; Volume 31.
125. Paudice, A.; Muñoz-González, L.; Lupu, E.C. Label sanitization against label flipping poisoning attacks. In Proceedings of the ECML PKDD 2018 Workshops: Nemesis 2018, UrbReas 2018, SoGood 2018, IWAISe 2018, and Green Data Mining 2018, Dublin, Ireland, 10–14 September 2019; pp. 5–15.
126. Koh, P.W.; Steinhardt, J.; Liang, P. Stronger data poisoning attacks break data sanitization defenses. *Mach. Learn.* **2022**, *1*, 1–47. [[CrossRef](#)]
127. Shen, J.; Zhu, X.; Ma, D. TensorClog: An imperceptible poisoning attack on deep neural network applications. *IEEE Access* **2019**, *7*, 41498–41506. [[CrossRef](#)]
128. Shao, K.; Zhang, Y.; Yang, J.; Li, X.; Liu, H. The triggers that open the NLP model backdoors are hidden in the adversarial samples. *Comput. Secur.* **2022**, *118*, 102730. [[CrossRef](#)]
129. Correia-Silva, J.R.; Berriel, R.F.; Badue, C.; De Souza, A.F.; Oliveira-Santos, T. Copycat CNN: Stealing knowledge by persuading confession with random non-labeled data. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8.

130. Shen, M.; Li, C.; Yu, H.; Li, Q.; Zhu, L.; Xu, K. Decision-based query efficient adversarial attack via adaptive boundary learning. *IEEE Trans. Dependable Secur. Comput.* **2023**, *21*, 1740–1753. [\[CrossRef\]](#)
131. Xiang, Y.; Xu, Y.; Li, Y.; Ma, W.; Xuan, Q.; Liu, Y. Side-channel gray-box attack for DNNs. *IEEE Trans. Circuits Syst. II Express Briefs* **2020**, *68*, 501–505. [\[CrossRef\]](#)
132. Hwang, U.; Park, J.; Jang, H.; Yoon, S.; Cho, N.I. Puvae: A variational autoencoder to purify adversarial examples. *IEEE Access* **2019**, *7*, 126582–126593. [\[CrossRef\]](#)
133. Guan, D.; Zhao, W. Adversarial Detection Based on Inner-Class Adjusted Cosine Similarity. *Appl. Sci.* **2022**, *12*, 9406. [\[CrossRef\]](#)
134. Gao, J.; Wang, B.; Lin, Z.; Xu, W.; Qi, Y. Deepcloak: Masking deep neural network models for robustness against adversarial samples. *arXiv* **2017**, arXiv:1702.06763.
135. Gu, S.; Rigazio, L. Towards deep neural network architectures robust to adversarial examples. *arXiv* **2014**, arXiv:1412.5068.
136. Akhtar, N.; Liu, J.; Mian, A. Defense against universal adversarial perturbations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3389–3398.
137. Meng, D.; Chen, H. Magnet: A two-pronged defense against adversarial examples. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Dallas, TX, USA, 30 October–3 November 2017; pp. 135–147.
138. Dziugaite, G.K.; Ghahramani, Z.; Roy, D.M. A study of the effect of JPG compression on adversarial images. *arXiv* **2016**, arXiv:1608.00853.
139. Tran, B.; Li, J.; Madry, A. Spectral Signatures in Backdoor Attacks. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Montreal, ON, Canada, 3–8 December 2018; pp. 8011–8021.
140. Metzen, J.H.; Genewein, T.; Fischer, V.; Bischoff, B. On detecting adversarial perturbations. *arXiv* **2017**, arXiv:1702.04267.
141. Katz, G.; Barrett, C.; Dill, D.L.; Julian, K.; Kochenderfer, M.J. Reluplex: A calculus for reasoning about deep neural networks. *Form. Methods Syst. Des.* **2022**, *60*, 87–116. [\[CrossRef\]](#)
142. Aithal, M.B.; Li, X. Mitigating black-box adversarial attacks via output noise perturbation. *IEEE Access* **2022**, *10*, 12395–12411. [\[CrossRef\]](#)
143. Wei, W.; Liu, L. Robust deep learning ensemble against deception. *IEEE Trans. Dependable Secur. Comput.* **2020**, *18*, 1513–1527. [\[CrossRef\]](#)
144. Liu, L.; Guo, Y.; Cheng, Y.; Zhang, Y.; Yang, J. Generating robust DNN with resistance to bit-flip based adversarial weight attack. *IEEE Trans. Comput.* **2022**, *72*, 401–413. [\[CrossRef\]](#)
145. Zoppi, T.; Ceccarelli, A. Detect adversarial attacks against deep neural networks with GPU monitoring. *IEEE Access* **2021**, *9*, 150579–150591. [\[CrossRef\]](#)
146. Nguyen-Vu, L.; Doan, T.P.; Bui, M.; Hong, K.; Jung, S. On the defense of spoofing countermeasures against adversarial attacks. *IEEE Access* **2023**, *11*, 94563–94574. [\[CrossRef\]](#)
147. Yang, J.T.; Jiang, H.; Li, H.; Ye, D.S.; Jiang, W. FAD: Fine-Grained Adversarial Detection by Perturbation Intensity Classification. *Entropy* **2023**, *25*, 335. [\[CrossRef\]](#)
148. Ryu, G.; Choi, D. Detection of adversarial attacks based on differences in image entropy. *Int. J. Inf. Secur.* **2024**, *23*, 299–314. [\[CrossRef\]](#)
149. Wu, S.; Sang, J.; Xu, K.; Zhang, J.; Yu, J. Attention, please! Adversarial defense via activation rectification and preservation. *ACM Trans. Multimed. Comput. Commun. Appl.* **2023**, *19*, 1–18. [\[CrossRef\]](#)
150. Sotgiu, A.; Demontis, A.; Melis, M.; Biggio, B.; Fumera, G.; Feng, X.; Roli, F. Deep neural rejection against adversarial examples. *EURASIP J. Inf. Secur.* **2020**, *2020*, 5. [\[CrossRef\]](#)
151. Xiang, Z.; Miller, D.J.; Kesidis, G. Reverse engineering imperceptible backdoor attacks on deep neural networks for detection and training set cleansing. *Comput. Secur.* **2021**, *106*, 102280. [\[CrossRef\]](#)
152. Shao, K.; Yang, J.; Ai, Y.; Liu, H.; Zhang, Y. BDDR: An effective defense against textual backdoor attacks. *Comput. Secur.* **2021**, *110*, 102433. [\[CrossRef\]](#)
153. Shao, K.; Zhang, Y.; Yang, J.; Liu, H. Textual Backdoor Defense via Poisoned Sample Recognition. *Appl. Sci.* **2021**, *11*, 9938. [\[CrossRef\]](#)
154. AlSobeh, A.M.; Gaber, K.; Hammad, M.M.; Nuser, M.; Shatnawi, A. Android malware detection using time-aware machine learning approach. *Clust. Comput.* **2024**, *27*, 12627–12648. [\[CrossRef\]](#)
155. Wang, R.; Li, Y.; Hero, A. Deep Adversarial Defense Against Multilevel $-\ell_p$ Attacks. In Proceedings of the 2024 IEEE 34th International Workshop on Machine Learning for Signal Processing (MLSP), London, UK, 22–25 September 2024; pp. 1–6.
156. Juraev, F.; Abuhamad, M.; Chan-Tin, E.; Thiruvathukal, G.K.; Abuhmed, T. Unmasking the Vulnerabilities of Deep Learning Models: A Multi-Dimensional Analysis of Adversarial Attacks and Defenses. In Proceedings of the 2024 Silicon Valley Cybersecurity Conference (SVCC), Seoul, Republic of Korea, 17–19 June 2024; pp. 1–8.
157. AlSobeh, A.; Shatnawi, A.; Al-Ahmad, B.; Aljmal, A.; Khamaiseh, S. AI-Powered AOP: Enhancing Runtime Monitoring with Large Language Models and Statistical Learning. *Int. J. Adv. Comput. Sci. Appl.* **2024**, *15*, 121–133. [\[CrossRef\]](#)

158. Alhazeem, E.; Alsobeh, A.; Al-Ahmad, B. Enhancing Software Engineering Education through AI: An Empirical Study of Tree-Based Machine Learning for Defect Prediction. In Proceedings of the 25th Annual Conference on Information Technology Education, El Paso TX, USA, 10–12 October 2024; pp. 153–156.
159. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-Based Learning Applied to Document Recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
160. Xiao, H.; Rasul, K.; Vollgraf, R. Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv* **2017**, arXiv:1708.07747.
161. Krizhevsky, A.; Hinton, G.E. *Learning Multiple Layers of Features from Tiny Images*; Technical Report; University of Toronto: Toronto, ON, USA, 2009.
162. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis. (IJCV)* **2015**, *115*, 211–252. [CrossRef]
163. Liu, Z.; Luo, P.; Wang, X.; Tang, X. Deep Learning Face Attributes in the Wild. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 13–16 December 2015; pp. 3730–3738.
164. Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; Ng, A.Y. Reading Digits in Natural Images with Unsupervised Feature Learning. In Proceedings of the NIPS Workshop on Deep Learning and Unsupervised Feature Learning, Granada, Spain, 12–17 December 2011.
165. Maas, A.L.; Daly, R.E.; Pham, P.T.; Huang, D.; Ng, A.Y.; Potts, C. Learning Word Vectors for Sentiment Analysis. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Toronto, ON, Canada, 9–14 July 2011; pp. 142–150.
166. Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C.D.; Ng, A.Y.; Potts, C. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, Seattle, WA, USA, 18–21 October 2013; pp. 1631–1642.
167. Wang, X.; Yamagishi, J.; Todisco, M.; Delgado, H.; Nautsch, A.; Evans, N.; Sahidullah, M.; Vestman, V.; Kinnunen, T.; Lee, K.A.; et al. ASVspoof 2019: A Large-Scale Public Database of Synthesized, Converted and Replayed Speech. *Comput. Speech Lang.* **2020**, *64*, 101114. [CrossRef]
168. Reimao, R.; Tzerpos, V. FoR: A Dataset for Synthetic Speech Detection. In Proceedings of the International Conference on Speech Technology and Human-Computer Dialogue (SpeD), Timisoara, Romania, 10–12 October 2019; pp. 1–10.
169. Yi, J.; Tao, J.; Fu, R.; Yan, X.; Wang, C.; Wang, T.; Zhang, C.Y.; Zhang, X.; Zhao, Y.; Ren, Y.; et al. ADD 2023: The Second Audio Deepfake Detection Challenge. *arXiv* **2023**, arXiv:2305.13774.
170. Müller, N.; Czempin, P.; Diekmann, F.; Froghyar, A.; Böttinger, K. Does Audio Deepfake Detection Generalize? In Proceedings of the Interspeech, Incheon, Republic of Korea, 18–22 September 2022; pp. 1–5.
171. Exchange, T.S. Taiwan Stock Exchange. Available online: <https://www.twse.com.tw/zh/about/news/news/list.html> (accessed on 5 April 2025).
172. U.S. Department of Transportation. Next Generation Simulation (NGSIM) Vehicle Trajectories and Supporting Data. Available online: https://data.transportation.gov/Automobiles/Next-Generation-Simulation-NGSIM-Vehicle-Trajector/8ect-6jqj/about_data (accessed on 5 January 2025).
173. Anderson, H.S.; Roth, P. Ember: An Open Dataset for Training Static PE Malware Machine Learning Models. *arXiv* **2018**, arXiv:1804.04637.
174. Coates, A.; Ng, A.; Lee, H. An Analysis of Single-Layer Networks in Unsupervised Feature Learning. In Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS), Ft. Lauderdale, FL, USA, 11–13 April 2011; pp. 215–223.
175. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778. [CrossRef]
176. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556.
177. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1–9. [CrossRef]
178. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520. [CrossRef]
179. Huang, G.; Liu, Z.; Maaten, L.V.D.; Weinberger, K.Q. Densely Connected Convolutional Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2261–2269. [CrossRef]
180. Conneau, A.; Kiela, D.; Schwenk, H.; Barrault, L.; Bordes, A. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP), Copenhagen, Denmark, 9–11 September 2017; pp. 670–680. [CrossRef]

181. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186. [\[CrossRef\]](#)
182. OECD. Catalogue of Tools & Metrics for Trustworthy AI; Available online: <https://oecd.ai/en/catalogue/tools> (accessed on 19 April 2025).
183. Yue, W.; Liu, B.; Stone, P. T-DGR: A Trajectory-Based Deep Generative Replay Method for Continual Learning in Decision Making. In Proceedings of the 3rd Conference on Lifelong Learning Agents (CoLLAs), Pisa, Italy, 29 July–1 August 2024.
184. Shafahi, A.; Huang, W.R.; Najibi, M.; Suci, O.; Studer, C.; Dumitras, T.; Goldstein, T. UPSET and ANGRI: Breaking High Performance Image Classifiers Using Adversarial Examples. In Proceedings of the British Machine Vision Conference (BMVC), Newcastle, UK, 3–6 September 2018.
185. Nelson, B.; Barreno, M.; Chi, F.J.; Joseph, A.D.; Rubinstein, B.I.; Saini, U.; Tygar, J.D.; Ren, K.; Xia, X. Misleading Learners: Co-opting Your Spam Filter. In *Machine Learning in Cyber Trust: Security, Privacy, and Reliability*; Tygar, J.D., Ed.; Springer: Berlin/Heidelberg, Germany, 2009; pp. 17–51. [\[CrossRef\]](#)
186. Xu, G.; Xin, G.; Jiao, L.; Liu, J.; Liu, S.; Feng, M.; Zheng, X. OFEI: A Semi-Black-Box Android Adversarial Sample Attack Framework Against DLaaS. *IEEE Trans. Comput.* **2023**, *73*, 956–969. [\[CrossRef\]](#)
187. Chang, C.W.; Liou, J.J. Enhancing Solver Robustness through Constraint Tightening for DNN Compilation. In Proceedings of the 2024 International VLSI Symposium on Technology, Systems and Applications (VLSI TSA), Hsinchu, Taiwan, 22–25 April 2024; pp. 1–4. [\[CrossRef\]](#)
188. Insights, F.B. Machine Learning Market to Reach USD 117.19 Billion by 2027; Increasing Popularity of Self-Driving Cars to Propel Demand from Automotive Industry. *GlobeNewswire* 2020. Available online: <https://www.globenewswire.com/news-release/2020/07/17/2063938/0/en/Machine-Learning-Market-to-Reach-USD-117-19-Billion-by-2027-Increasing-Popularity-of-Self-Driving-Cars-to-Propel-Demand-from-Automotive-Industry-says-Fortune-Business-Insights.html> (accessed on 16 October 2024).
189. Michael, J.B.; Kuhn, R.; Voas, J. Cyberthreats in 2025. *Computer* **2020**, *53*, 16–27. [\[CrossRef\]](#)
190. Chen, Z.; Li, B.; Wu, S.; Ding, S.; Zhang, W. Query-efficient decision-based black-box patch attack. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 5522–5536. [\[CrossRef\]](#)
191. Li, X.C.; Zhang, X.Y.; Yin, F.; Liu, C.L. Decision-based adversarial attack with frequency mixup. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 1038–1052. [\[CrossRef\]](#)
192. Papernot, N.; McDaniel, P.; Wu, X.; Jha, S.; Swami, A. Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks. In Proceedings of the IEEE Symposium on Security and Privacy, San Jose, CA, USA, 22–26 May 2016; pp. 582–597.
193. Hasan, F.A.; Ashqar, H.I.; AlSobeh, A.; Darwish, O. Blockchain-Based National Digital Identity Framework—Case of Palestine. In Proceedings of the 2024 International Conference on Intelligent Computing, Communication, Networking and Services (ICCNs), Dubrovnik, Croatia, 24–27 September 2024; pp. 76–83.
194. Modas, A.; Moosavi-Dezfooli, S.M.; Frossard, P. Sparsefool: A few pixels make a big difference. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 9087–9096.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.