Article

A Survey on Machine Learning Data Poisoning Attacks: Detection and Prevention

Bryndís Rósa Sigurpálsdóttir †, Eydís Sjöfn Kjærbo †, Grigorios Papaspyropoulos † and Panagiotis Koukourikos †

† These authors contributed equally to this work.

Abstract: Machine learning-based systems, which have become widely employed for a range of applications such as financial fraud detection and healthcare decision-making, pose a significant danger of data poisoning attacks. In this survey, we are going to review a number of countermeasures against data poisoning attacks that have been suggested to this day. We will provide a broad variety of defence techniques and detection techniques, discuss each of them and assess their advantages and disadvantages. We hope that this contribution will help with finding more and better solution to this problem in the near future.

Keywords: Machine Learning, Data Poisoning, Data Poisoning Prevention Techniques, Data Poisoning Detection Techniques

Citation: Sigurpálsdóttir, B.; Kjærbo, E.; Papaspyropoulos, G.; Koukourikos, P. A Survey on Machine Learning Data Poisoning Attacks: Detection and Prevention. *Journal Not Specified* **2021**, *1*, 0. https://doi.org/

Received:
Accepted:
Published:

1. Introduction

Machine Learning (ML) has made enormous progress in recent years and is growingly being used to make decisions on our behalf in a variety of activities such as computer vision, performing predictions or supporting decision making in healthcare, financial fraud detection, malware and cyber-attack defense and surveillance, and autonomous vehicles, to name a few. However, the increasing usage of ML technologies has made them enticing targets for attackers looking to exploit such approaches for malevolent purposes [1, 2, 3].

Data poisoning is a security concern to machine learning systems in which an attacker manipulates a system's training data to affect its behavior. Deep learning systems are especially vulnerable to this sort of attack since they require a large amount of data to train and are commonly trained (or pre-trained) on massive datasets obtained from anonymous and unverified web sources. Poisoning attacks, which can be divided into conventional poisoning and backdoor attacks, have recently caught the interest of both industry and academics. Several research publications have demonstrated the intensity of these attacks [4, 5, 6, 1, 2, 7, 8, 9].

The primary difference between the two attacks is that conventional poisoning affects machine learning availability, whereas backdoor attacks target integrity. Until recently, conventional poisoning attacks represented the majority of data poisoning attacks. It wasn't until 2017 that the first studies proving the feasibility of backdoor poisoning were published [10, 11]. The conventional attacks are aimed to overload your system with incorrect data, causing whatever boundary your model learns to lose accuracy. In this approach, the attacker is aware of the training algorithm but has no influence over the training process [3, 12].

Backdoor attacks are a sort of attack in which the attacker selects a trigger (a tiny patch), prepares poisoned data depending on the trigger, and then sends it to the victim to train a model using. The victim is unaware that the model has been compromised since backdoor attacks have no effect on the classifier's performance on "clean" patterns. If the attacker applies the trigger to a source category image, the model will misclassify it as a target category [11, 13, 14]. Figure 1 is a common example of a backdoor attack where a little sticker placed on a traffic sign can change a stop sign to a speed limit sign [12, 15].

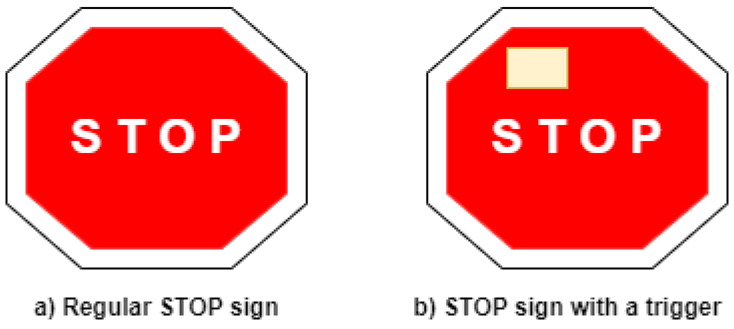

Figure 1. An input sample with the trigger pattern will cause the model to fail and demonstrate the attacker's intended functionality in a backdoor poisoning attack.

In response to the increasing threats posed by data poisoning, particularly backdoor attacks, researchers have begun to propose viable ways to protect the Machine learning system from the inclusion of a concealed backdoor trigger. Because of their stealth, triggers in backdoor attacks are extremely tough to protect against. Many protection strategies centered on recognizing the triggers in the input sample [4, 16, 17].

The purpose of this study is to explain the vulnerability that hackers exploit in intelligent systems, namely via poisoning data. The study outlines current detection procedures as well as certain defensive strategies utilized by some researchers.

1.1. Contribution and Research Questions

We present a systematic survey of the techniques used to prevent and detect data poisoning attacks in this systematic literature. As we quantitatively demonstrate, the increasing use of Machine Learning technologies in recent years has made them appealing targets for attackers looking to exploit such approaches for malicious purposes, and this momentum motivates this article as well as the need for an up-to-date systematic survey. As our key contribution, we conduct a review of the literature from 2009 to 2021. Because data poisoning just recently piqued the interest of both industry and academia, practically all of the material cited is from the years 2019-2021. The detection and prevention approaches identified by the research community are classified, compared, and discussed.

Our research questions might be stated in the following manner:

RQ1: What are the existing data poisoning detection and prevention approaches and what are their advantages and disadvantages?

RQ2: What are some recent data-poisoning concerns that have been identified by recent research?

These questions aim to provide an overview of current detection and prevention approaches, while highlighting the benefits and drawbacks of each method and describing the context in which these methods can be utilized. Answering these questions can help us determine what strategies are likely to be used in the future to prevent and detect data poisoning.

1.2. Outline

The study is divided into six sections, with a specific emphasis on data poisoning prevention and detection in ML systems. Section 1 provides an overview of data poisoning in machine learning, as well as the research questions for this study. Section 2 outlines past surveys and literature reviews conducted by other researchers in order to describe this paper's potential contribution. The approach we take in collecting relevant papers to base our survey on is described in section 3. Section 4 provides an overview of the defense techniques and section 5 provides an overview of the detection techniques. At last, we conclude the paper and describe in a few words some future works we suggest, in section 6.

2. Related Works

Since machine learning (ML) is a popular topic today, many surveys and extensive research studies have been made about it. Material regarding ML can be accessed in a variety of ways, but table 1 summarizes papers that focus countermeasures against poisoning attacks or include this topic. The table lists these papers and emphasizes their main contributions and how this study may enhance the state of the art in data poisoning attack prevention and detection research.

Table 1: The novelty of this paper with respect to related surveys and literature reviews.

Paper	Main Contribution and Reference Metrics	This Paper
[4]	Focuses on neural Trojan attack and defense techniques	Focuses on data poisoning attacks
[16, 18, 19]	Focuses on defenses against Adversarial Learning attacks on deep neural network classifiers	Provides a particularly thorough review on data poisoning attack defenses
[17]	Provides a review of backdoor attacks against deep neural networks and possible defenses in the field of image and video classification.	Focuses on Data poisoning attacks against all machine learning techniques
[20]	Provides a review of adversarial machine learning attacks, and defenses and how it applies to cyber-physical systems	Focuses mainly on Data poisoning attacks
[21]	Provides a review of adversarial attacks, attack algorithms and defense techniques	Provides a particularly thorough review on data poisoning attack defenses
[22]	Provides a systematic review of the security of cloud-hosted machine learning models and services and examines attacks and defense methods	Provides a particularly thorough review on data poisoning attack defenses for ML systems
[23]	As the name of the paper suggests, it provides a systematic evaluation of backdoor data poisoning attacks on image classifiers. It also provides an assessment of trigger patterns, poisoning strategies, datasets, potential defensive regularization techniques, and more.	Provides a particularly thorough review on data poisoning attack defenses for all ML systems
[24]	Focuses on threats to the integrity of connectionist AI applications	A more thorough review on defenses
[15]	Provides a thorough overview of vulnerabilities in machine learning systems and countermeasures, the evolution of this research area, and future challenges.	Provides a more thorough and newer review on data poisoning attack defenses
[13]	Provides a review of backdoor attacks and countermeasures and compares their advantages and disadvantages	Include newer countermeasures
[25]	Reviews existing poisoning attacks and countermeasures and highlights remaining challenges	Puts more focus on defenses
[26]	A thorough review of adversarial attacks against AI applications, adversarial knowledge, and capabilities, existing methods to generate adversarial examples, and existing cyber defense models.	Puts more focus on defenses

Reference [4] is a survey from last year, 2020. This survey focuses mainly on neural trojan and proposed defense techniques. It is a quite short survey but provides a great fundamental base for this material. Our survey is mainly about countermeasures against poisoning attacks but also provides some information in this field.

All of these papers focus on Deep Neural Networks (DNN) [16, 18, 19, 17] but [17] focuses on the field of image and video classification in DNN. All of them are from the last two years and are similar, providing an in-depth survey of defense methods, including a review of attacks on DNN. These are some great surveys but our has a broader scope, focusing on all ML methods.

[20] is a short but interesting paper that focuses on machine learning security vulnerabilities in cyber-physical systems, for example, robots, drones, and autonomous vehicles. It also reviews adversarial machine learning techniques and defenses. The scope of this paper is very specific, not including all ML methods like this survey. [22] has a similar approach focusing mainly on cloud machine learning security, as does [23], focusing on image classifiers.

Reference [21] is a very thorough survey about adversarial attacks on AI systems. They talk about different concepts and types of adversarial attacks and the dangers they cause, including defense techniques that can be used. This is a great survey but it doesn't provide as many solutions against data poisoning as our survey does. [26] also focuses on adversarial attacks, but focuses more on the attacks than the defenses.

[24] is a review article from 2020 about vulnerabilities of connectionist AI applications. Connectionist AI applications are for example DNN, convolutional neural networks, radial basis function networks, and support vector machines (as explained in [24]). This article focuses on threats to integrity, discussing adversarial attacks and poisoning attacks.

All of these articles [15, 13, 25] are similar to this survey. [15] focuses on all vulnerabilities in machine learning systems while [13] focuses especially on backdoor attacks and [25] on poisoning attacks. These are all great papers but our survey provides newer solutions and focuses mainly on data poisoning attacks.

We hope our survey will be a good addition to this great list of surveys and research articles.

3. Methodology

This section describes the search and paper selection strategy used to select literature for this paper. The paper selection methodology incorporates approaches provided in both [27, 28]. These approaches provide guidelines on how to conduct a systematic literature review and how to use snowball sampling for paper inclusion.

In 3.1 we show how articles were found in different databases using the PICO criteria and in 3.2 we show how we systematically exclude and include papers.

3.1. Search Strategy

We used the PICO (Population, Intervention, Comparison, and Outcomes) criteria to discover keywords and generate search strings based on the paper's research question. The following are the criteria for this particular paper:

- **Population:** Population: The population for this survey is all researches on Machine Learning Data Poisoning Attacks prevention and detection measures, including other surveys.
- **Intervention:** We do not have a specific intervention to explore in the context of this investigation.
- **Comparison:** Different techniques of poisoning attack prevention and detection are evaluated to find their advantages and drawbacks.
- **Outcomes:** The expected outcomes are an overview of current data poisoning prevention and detection measures.

These criteria yielded two keywords: data poisoning prevention and detection. Since the primary purpose of the research is to give a mitigation-oriented analysis of data poisoning prevention and detection studies, we considered these two keywords to be the most relevant.

We used five databases to find relevant papers for this survey. These five sources are ACM, Scopus, DTU FindIt, and Google Scholar. We chose these sources because they provide a great variety of studies that are free to us as students at DTU. Search strings for each database are shown in table 2 and the results from these searches are shown in table 3.

Table 2: Searches in databases

Database	**Search**
ACM	("data poisoning" AND (("prevention" OR "prevent" OR "preventing") OR ("detection" OR "detect" OR "detecting")))
Scopus	TTITLE-ABS-KEY ("data poisoning" AND (("prevention" OR "prevent" OR "preventing") OR ("detection" OR "detect" OR "detecting")))
DTU FindIt	"data poisoning" AND (("prevention" OR "prevent" OR "preventing") OR ("detection" OR "detect" OR "detecting"))
Google scholar	("prevention" OR "prevent" OR "preventing") OR ("detection" OR "detect" OR "detecting") "data poisoning"

Table 3: Number of studies per database

Database	**Search results**
ACM	128
Scopus	76
DTU FindIt	199
Google scholar	2870

3.2. *Exclusion/Inclusion Process*

Before the exclusion/inclusion process started, we decided to decrease the number of papers from Google Scholar by only including review articles. This decision was taken since it would be extremely time-consuming to review all 2870 papers. This reduction resulted in 144 articles.

We excluded papers in four steps. The first step was exclusion mostly based on publishing year, language, and accessibility. In the second, third, and fourth steps we excluded papers based on relevancy by reading the title and abstract in step 2, introduction and conclusion in step 3, and if needed, the full text in step 4.

We included papers in the fifth step by using the backward snowballing technique.

3.2.1. Initial Exclusion

The initial exclusion stage eliminated publications based on the following exclusion criteria.

- Studies from 2009 or older.
- Studies presenting non-peer-reviewed material.

- Studies not presented in English.
- Studies not accessible in full-text
- Studies that are duplicates of other studies.

Papers that did not fulfill all criteria were dismissed. Since there has been a lot of progress in this field in the last few years, we decided to eliminate studies published before 2010. This decision might eliminate some relevant studies, but by using the backward snowballing technique all relevant studies should be found eventually.

The total number of papers from each database after the initial exclusion is shown in table 4. Some papers were found on multiple databases, so the total number of papers from all databases after the initial exclusion was 335.

Table 4: Number of studies per database after the initial exclusion

Database	**Search results**
ACM	123
Scopus	26
DTU FindIt	121
Google scholar	114
Total	384

3.2.2. Title and Abstract Review

Following the first round of elimination, the title and abstract for each paper was reviewed. The purpose of this method was important for two reasons: first, to eliminate any irrelevant papers, and second, to determine which research questions the paper might address. If the abstract of a paper did not suggest that it would be useful in addressing the research questions, it was discarded.

This exclusion was mainly based on the following exclusion criteria.

- Studies not related to the prevention of data poisoning in Machine Learning.
- Studies not related to the detection of data poisoning in Machine Learning.

As shown in table 5 this step eliminated 145 papers.

3.2.3. Introduction and Conclusions Review

The third elimination was more thorough than the previous one. All remaining papers were observed again, but the papers were redistributed so each researcher got a new set of papers to review. This time we observed the introduction and conclusion for each paper. This method was aimed to eliminate all irrelevant papers that might have gotten through former exclusions. As shown in table 5 this step eliminated 79 papers.

3.2.4. Sorting and full Text Review

The final elimination was performed when dividing the papers into groups based on subjects and writing the paper. We did a full-text review on papers that we had doubts about and eliminated papers that weren't useful. A short summary for each remaining paper was written to assist all reviewers to understand the relevance of each piece without having to read it.

As shown in table 5 this step eliminated 32 papers.

3.2.5. Backwards Snowballing

Since we might have missed some important papers when eliminating papers from Google Scholar that were not classified as review articles and eliminating articles published before 2010, we did a backward snowballing sampling. This technique is explained in [28]. We went through every paper's references and checked if we could find more papers relevant to this survey.

As shown in table 5 this step included 8 papers.

Table 5: Number of studies per database after each exclusion

Databases	Search results
In the beginning of exclusion	3273
The result of the Google Scholar exclusion	547
The result of the first exclusion	335
The result of the second exclusion	190
The result of the Introduction and Conclusions Review	111
The result of Sorting and full Text Review	79
The result of Backwards Snowballing	87

4. Defence Techniques

Machine learning is widely employed in a wide range of applications. However, they are vulnerable to data poisoning attacks, in which sophisticated attackers add a tiny number of detrimental samples into the training dataset in order to disrupt the learning process. Such weaknesses might jeopardize a wide range of security-critical sectors, such as self-driving cars, biometric identity recognition, and computer vision [29]. In this section, we'll go through some of the most data poisoning defensive techniques.This section will be broken into four sections: defenses against backdoor attacks, defenses against conventional data poisoning, defenses against Federated Learning System poisoning attacks, and defenses against label flipping attacks [30].

4.1. Defenses against backdoor attack

Despite the fact that backdoor attacks on classifiers were just recently proposed, some work to avoid them is already underway. A backdoor trigger may be added into the model during or after training. When a backdoor is introduced during training, the training dataset is "poisoned" by introducing a classification between a certain pixel pattern (the trigger) and a target label. To train the backdoor, the trigger pattern is applied to each item in a randomly selected sample of training data, and each item's label is assigned to represent the target label. The poisoned data is combined with the clean training dataset to train the model. The resulting "backdoored" model learns both normal classification and the trigger-target label connection. The model then assigns a high likelihood to any input that has the trigger to the target label [31, 32]. A variety of approaches have been developed to combat backdoor attacks. Neural network verification, compromised neural network repair, and Trojan bypass approaches are among the three techniques. We'll go over each one separately. Existing security measures detailed in this section usually rely on training a separate machine learning model to recover, overcome backdoor triggers or introducing honeypots defense. This needs a significant amount of computing on the defense's part, lowering the value of MLaaS. (which is offloading computation to the service provider)) [33, 34, 35].

4.1.1. Neural network verification

In this approach any abnormality generated by a concealed trigger can be identified by putting a neural network to the test. However, because the majority of triggers have very specific triggers, neural verification approaches must be exceedingly accurate in order to uncover concealed triggers. Several strategies have been presented in order to achieve this goal [4].

A PAC-style algorithmic framework for approximate quantitative verification of neural networks is developed in one research [36]. This framework called NPAQ introduces the concept of equi-witnessability encodings in CNF equations. Counts are preserved, and composibility is ensured under logical conjunctions. The method determines how well P holds across N when given a set of trained neural networks (N) and a property (P). As a result, if a concealed trigger is found, NPAQ may be used to demonstrate that retraining eliminates the trigger by showing that the property that causes the trigger, P, no longer has power over the network, N. The usage of NPAQ in two real point applications is detailed in this article: measuring robustness to adversarial inputs and evaluating the efficacy of backdoor attacks. The first method is to calculate how many adversarial samples a neural network contains by analyzing its resilience to adversarial input. The second assessed the effectiveness of a backdoor attack by calculating how many trigger inputs it can successfully function for. Despite the fact that NPAQ has been proved to be successful in defending against backdoor assaults, this technique relies on training separate machine learning model to overcome the backdoor trigger this requires a substantial amount of compute on the part of the defender.

Another approach called Sensitive-Sample Fingerprinting verifies the integrity of deep learning models stored in the cloud by employing a small set of human unnoticeable transformed inputs that make the model outputs sensitive to the model's parameters. Even small model changes can be clearly reflected in the model outputs [37, 38]. By querying the network with these sensitive-samples and validating their categorization, one can dynamically check that the tested network has not been maliciously changed to include concealed triggers. The primary advantages of this approach is the high effectiveness and reliability, > 99.95 % attack detection rate on all evaluated attacks with only black-box accesses. Furthermore, this approach has been successful and efficient to detect Machine learning integrity breaches according to an examination of different categories of genuine Machine Learning integrity attacks. Another limitations to this approach is that that it is unable to differentiate between malicious model alterations and benign model updates As a result, every time the defender modifies the model in the cloud, he must produce a new Sensitive-Sample fingerprint for integrity verification.

4.1.2. Restoring compromised Neural Models

Model correction and trigger-based Trojan reversal are two sorts of methodologies for restoring broken neural models discussed in this section. The former contains generic methods for modifying neural networks to remove Trojan functionality, whereas the latter discovers Trojan trigger first and then patches neural networks accordingly [4].

Model Correction

Several strategies have been studied for retraining and pruning neural networks infected with a concealed harmful trigger.

One paper provides a way for retraining the model to forget the hidden trigger once he has been found while still functioning properly with legitimate data. Because the re-training utilizes only legal data as training samples, the triggers in the weights may be overwritten during the re-training process, rendering the concealed trigger ineffective. It is important to note that the re-training must be supervised, which means that the defender must know the label of each training sample. The re-training method is successful in lowering the activation rate of concealed triggers. However, it has the following limitation: The classification accuracy of authentic data will be reduced by an average of 2% regardless of whether the neural model was hidden trigger embedded or hidden trigger free before to re-training [39]. Other papers have demonstrated that retraining DNN often provides a way towards backdoor discovery. The defender can then often remove the malicious trigger by fine-tuning the model over benign dataset. Often, the defender can then eliminate the malicious trigger by fine-tuning the model using a benign dataset.There is no requirement for prior information or assumptions

about the triggering pattern for this method [17]. Other defense based on retraining the model uses data augmentation during fine tuning by adding a Gaussian random noise to each benign picture in n benign dataset, the logic behind this strategy is that data augmentation should cause the network to disturb the weights to a greater extent, aiding backdoor elimination. Given that model level defenses function before the network is deployed under operational settings, they do not have a substantial computational burden.
Model Correction defenses do not have a major computational overhead since they function before the network is deployed under operational settings. As a disadvantage, in order to execute these approaches, the defender need white-box access to the model as well as the availability of a big benign dataset. [40, 17].

Another study looks at the impact of network pruning on the resilience to poisoning attacks. Pruning a neural network involves the removal of the network's less important neurons. The computational complexity and size of a pruned neural network are lower than those of the original network. The authors of one defence presented a neural malicious trigger hardening strategy in their research, which comprises pruning a neural network so that accuracy is not reduced but the difficulty of adding harmful functionality to the trained network is significantly increased. Pruning removes unneeded capacity from a network; hence, a model with the majority of its neurons pruned has the strongest resilience against malicious trigger infection attack. [41]. From the viewpoint of the defender, the pruning defense has numerous desirable characteristics. For one thing, it is computationally cheap and needs just that the defender be able to run a trained model on validation input. However, the pruning defense also suggests an improved attack strategy that refer to as the pruning-aware attack if the attacker is aware of the pruning defence [42, 43, 44].

Trigger-based Trojan Reversing

In this technique, a backdoor trigger is discovered using an optimization strategy. The optimization technique calculates the smallest amount of modification required to shift inputs from all classes to a target class. If the interruption is minor, it was most likely caused by a backdoor trigger. The triggered network is then patched with a reverse engineered trigger by retraining the network on legitimate inputs to remove the backdoor with the disguised trigger attached to it. Non-convex optimization-theoretic formulation driven by explainable AI and other heuristics can be utilized to improve detection accuracy of this method [4, 45, 17].

In 2019, the first trigger-reconstruction approach, called Neural Cleanse, was proposed[46, 47, 48].This approach is based on the following thought: A source-independent backdoor exploits the sparsity of the input space to provide a shortcut to the target class. Neural Cleanse employs processes to reverse engineer an input pattern for each output label, resulting in all inputs stamped with the pattern being categorized as belonging to the same target label. If a label's created pattern is substantially smaller than the generated patterns of other labels, Neural Cleanse deems it trojaned or malicious. Neural Cleanse, on the other hand, implies that the trigger overwrites a tiny (local) portion of the picture, such as a square pattern or a sticker. Other research has revealed that this approach fails to detect triggers for various types of local triggers. The cause for this failure is due to the low integrity of the rebuilt triggers, which are dispersed and too big when compared to the genuine trigger [49, 50, 51].

Chen et al. present both a hidden trigger removal approach and a hidden trigger detection technique. The authors advise flattening the last hidden layer, lowering its dimensionality, and clustering. Malicious triggers can be identified using anomalous clustering features. This strategy eliminates a trigger by retraining the neural network to exclude the aberrant cluster while maintaining accuracy. Despite the fact that this approach does not require verified or trustworthy data, it was able to efficiently remove the backdoor trigger while still performing well on standard samples[47]. This method, however, exclusively targets pixel-pattern backdoors. Furthermore, if utilized in a federated learning environment, this strategy attempts to invert the model in order to obtain the training data, which violates the privacy requirement.

4.1.3. Bypassing neural trojans

There have also been research on how to avoid any existing network triggers.

One study presented "Februus," a paradigm for eliminating input-independent trigger patterns in images. The input picture is routed to the "Februus" system, which detects and eliminates Trojan trigger patterns before sending it on to the neural network. A three-step approach is used to remove the neural triggers. The first is a visual description in which the Trojan is discovered using a logit score-based method; if the trigger is present, it has the greatest impact on the input's classification into the target class. The trigger is then masked out, and the input is reconstructed by inpainting on a benign image. This input purification framework may operate as a black box between the input and the neural network, reliably identifying benign inputs while without risking the neural network's accuracy [4]. The Februus defensive approach may sanitize trigger inputs at runtime without the need for anomaly detection methods, model retraining, or expensive labeled data. Unfortunately, run-time defenses rely on some notion of interpretability that the attacker may readily modify [52].

Another method proposed by Liu et al [39] used an autoencoder-based inputpre-processing mechanism. This autoencoder is a neural network with the same input and output dimensions that was trained entirely with genuine input data and is positioned between the input and the hacked neural network. It works by lowering the mean-squared error between the training set and the rebuilt pictures, guaranteeing that any fraudulent inputs are badly reconstructed and do not trigger the malicious trigger. Using this method, just 9.8 % of fraudulent inputs still trigger the malicious trigger, but real data classification accuracy is only degraded by 2%. This strategy, however, comes with a considerable cost in terms of complexity and computing [53].

Furthermore, a black box technique called "Neo" prevents backdoor attacks by finding the proper prediction outcomes of poisoned pictures, as well as the stealthy character of such assaults being diminished by reconstructing the backdoor triggers.NEO has the unique ability to isolate and recreate the backdoor trigger [54]. This strategy implies that the defender does not have access to the training process and hence no influence on the backdoor trigger.NEO, on the other hand, assumes that the backdoor trigger is modest, and therefore is not designed to fight targeted backdoor attacks. In robust models, this strategy does not mitigate backdoor triggers [55].

4.1.4. Honeypots Defense

Unlike traditional defenses, Honeypot Defense does not attempt to prevent attackers from computing effective adversarial examples; instead, it creates a "honeypot" for attackers by inserting a subset of selected model vulnerabilities, making them easy to discover and difficult for the attacker to ignore. When attackers build adversarial instances, this strategy aims to guarantee that they identify honeypot perturbations rather than inherent flaws. Because of their similarity to the chosen honeypot, attackers that apply these honeypot perturbations to their inputs are quickly detected by the model. This trapdoor-based protection provides a high detection success rate against hidden malicious triggers across a wide range of application domains, including the most powerful attacks such as BPDA and BLack-box attacks. However, this strategy has not been proven against all possible adversarial cases against the Machine learning model. The attacker behavior in the research was investigated via empirical assessment, hence it was only tested against six sample data poisoning attacks; the resultant adversarial cases follow the embedded trapdoors with a probability of 94 percent or higher. This implies that today's practical attackers will very certainly follow the patterns of the embedded trapdoors [56] .

Table 6: Advantages and disadvantages of defences against backdoor attacks

Defense Technique	**Advantage**	**Disadvantage**
NPAQ	Robustness to Adversarial inputs	Substantial amount of compute on the part of the defender since this technique relies on training separate machine learning model to overcome the backdoor trigger
Sensitive-Sample Fingerprinting	Senstive to even small model changes	Unable to differentiate between malicious model alterations and benign model updates
Model Correction	Do not have a major computational overhead	Need white-box access to the mode and the training must be supervised
Network pruning	Computationally cheap and effective	Vulnerable to pruning-aware attack
Neural Cleanse	Computationally cheap	Fails to detect triggers for various types of local triggers
Chen approach	Does not require verified or trustworthy data	Only targets pixel-pattern backdoors and violates the privacy requirement in Federated learning environment.
Februus	Sanitize trigger inputs at runtime	attacker may readily modify interpretability it uses.
Liu method	Real data classification accuracy is only degraded by 2%	Vconsiderable cost in terms of complexity and computing
Neo	Isolate the backdoor trigger	In robust models, this strategy does not mitigate backdoor triggers
Honeypots Defense	High detection success rate againsthidden malicious triggers across a wide range of applicationr	Needs to be tested against more attacks

4.2. Defences against conventional data poisoning

Conventional data poisoning attacks are availability attacks that aim to reduce the accuracy of the learnt classifier. Availability attacks are generally simple but broad; the attacker injects as much bad data into a database as feasible. Following a successful attack, the machine learning system will be utterly incorrect, yielding little to no genuine or valuable insights [57, 58, 59].

4.2.1. Defenses Against Classifier-Degrading DP Attacks

The most frequent method for defending classifiers from embedded database poisoning attempts is to employ robust learning, which seeks to maintain (generalization) accuracy even when the training set is poisoned [16].

The following are two examples of typical training methodologies.

1. Following normal classifier learning, some sort of training set sanitization is performed. Outlier samples may be detected and removed as part of sanitization. Alternatively, it might involve feature selection or compaction, with the goal of neutralizing data poisoning by deleting small signal components/features that are assumed to be largely employed by the attacker.
2. Robustly modifying the classifier training target function, accounting for both outliers and inliers using adversarially mislabeled labels.

An illustration of the first strategy is provided in paper [60] performing nearest-neighbor-based outlier identification, followed by SVM optimization. Despite the fact that deleting nonattack outliers has no effect on the classifier estimate, they are able to obtain a (domain-dependent) upper bound on the worst-case accuracy loss from a Database poisoning attack.

An example of strategy number two is provided in paper [61] , the dual SVM objective function is substituted with an anticipated dual function based on the risk of mislabeling, which is dependent on the training labels. This increases the system's resiliency by preventing data poisoning. This allows for more data points to be employed as support vectors in the SVM's weight vector estimation, resulting in greater robustness. One issue is that the method relies on knowing the mislabeling probability, which is a hyperparameter that is (typically) unknown. Even if this value is chosen cautiously and in the worst-case scenario (assuming a high incidence of mislabeling), accuracy will be impaired in the absence of a Database Poison attack.

4.3. Defensive against Federated Learning System poisoning attacks

Federated Learning provides a secure collaborative machine learning platform for a variety of devices that does not expose their private data. Federated learning is a new machine learning paradigm that is growing rapidly [62, 63, 8]. Federated learning gives user minimal control over the local data and the training process. As a result, it is vulnerable to poisoning attacks, which occur when hostile or compromised clients utilize malicious training data or local updates as an attack channel to poison the learned global model. Federated Learning offers a secure collaborative machine learning platform for a wide range of devices that does not reveal users personal data. Federated learning is a new machine learning paradigm that is quickly gaining in popularity [64, 65, 11].

The federated learning framework has many benefits;

- Scalability: Federated Learning allows diverse devices to learn from one another, making the network as a whole more scalable.
- Create local models to solve low-throughput and high-latency problems: Creating local models reduces latencies and uses less power than training a single central model.
- Increase accuracy: Federated Learning models are more innovative than centrally trained models because they are learned by aggregating several local models and approaching data from multiple angles at the same time. Training Time and Cost
- Savings: Rather than centrally training a model, training multiple local models and then aggregating them into a central model takes less time. The price of training is also lower.
- Ensure personal data privacy and security: Because the training data never leaves the actual site, all sensitive information remains local, ensuring personal data privacy and security. The federated approach is preferable to the centralized alternative because it protects data privacy by implementing the European Union's General Data Protection Regulations.
- Data Minimization: The data minimization principle is used by Federated Learning to ensure that only the learned model is processed centrally, while the raw data remains hidden. Furthermore, the models that are sent out are only temporary and are discarded as soon as they are merged into the global model. This ensures the storage and purpose limitation principles outlined in the General Data Protection Regulation [66] and improves the overall system's efficiency and robustness.

Federated learning lets the defender just a limited amount of control over the local data and the training process. As a result, it is prone to poisoning attacks, which occur when malicious training data or local updates are used as an attack vector to poison the learnt global model. The framework's design for Federated Learning makes implementing protective methods much more challenging. For example, introducing gaussian noise to the local models (for Data Poisining) might confuse the aggregation methods, thereby leaving out the benign individuals (while applying model poisoning defense measures). In this part, we will explore existing data poisoning defense mechanisms implemented for federated learning frameworks. [67, 8, 65, 64, 63, 68, 62, 69, 11]

CONTRA : The CONTRA defensive approach is presented in this study [66] to defend Federated Learning systems from poisoning attacks such as label-flipping and backdoor attacks. CONTRA employs a reputation system to dynamically promote or punish individual clients based on their perround and historical contributions to the global model, as well as a cosine-similarity-based measure to assess the believability of local model parameters in each round. CONTRA decreases attack success rates dramatically while maintaining good accuracy with the global model. CONTRA, on the other hand, may be less successful or may fail in the rare scenario where there is only one malicious client in the system. For example, the label-flipping attack reduces accuracy from 85.22 percent (with 20 malicious clients) to 81.23 percent (with 1 bad client), while increasing attack success rate from 1.9 percent to 8.43 percent. This is because, in Federated learning, CONTRA depends on similarity (i.e. alignment) across rogue clients with the same poisoning aim to detect suspicious clients.

Byzantine Robust Aggregation : [70, 66] In SGD-based federated learning techniques, the aggregator computes the average of the local models to update the global model's weights. Robust aggregation schemes proposed various aggregation rules to replace the average of model updates with a robust estimate of the mean, such as median aggregation, trimmed mean aggregation, or Krum aggregation, which minimizes the Euclidean distances between selected local models, to eliminate incorrect local updates caused by Byzantine errors.
Using increased learning rates, byzantine robust aggregation algorithms show success against untargeted poisoning assaults and targeted model poisoning attacks. However, one study found that framing the assault as an optimization problem reduced the accuracy of global model tests. Furthermore, when a system has a large number of Byzantine opponents, it performs poorly or fails entirely. [71].

Clustering-Based Detection : [72] Grouping-based approaches require aggregating model updates and then grouping them into two groups using dimensionality reduction techniques such as principal component analysis (PCA), k-means, or DBSCAN clustering algorithms, for example. As a result, client clusters with less than n/2 clients are identified as possibly harmful. This technique has been shown to yield promising results These approaches, on the other hand, assume that the training data is independent and distributed evenly. According to a study on PCA-based defense, when the data is non-identifiable, clustering-based defense strategies fail to discriminate between model changes given by malevolent clients and honest customers. Intuitively, this is because honest clients' model updates may diverge in most iterations if they have substantially unbalanced non-i.i.d. training data [66].

Table 7: Advantages and disadvantages of defences against Federated Learning System poisoning attacks

Defense Technique	**Advantage**	**Disadvantage**
CONTRA	Decreases attack success rates dramatically while maintaining good accuracy with the global model.	May fail in the rare scenario where there is only one malicious client in the system
Byzantine Robust Aggregation	success against untargeted poisoning assaults and targeted model poisoning attacks.	When a system has a large number of Byzantine opponents, it performs poorly or fails entirely
Clustering-Based Detection	This technique has been shown to yield promising results	When the data is non-identifiable, clustering-based defense strategies fail to discriminate between model changes given by malevolent clients and honest customer

4.4. Defense against Label Flipping Attacks

Label flipping attacks are a type of data poisoning in which the attacker has complete control over the labels assigned to a subset of the training points. Even if the attacker's skills are limited, these attacks have been found to be successful in degrading the Machine learning system's performance considerably. These attacks are applicable to a wide range of applications, including malware detection [73].

Label-based Semi-supervised Defense: The semi-supervised label-based defense discussed in paper [73]. The label spreading technique is trained using validation data before being used to predict the labels of training data using a model.
The fundamental idea behind this method is to rectify reversed labels using clustering algorithms. Each clustering method has its own measure, so therefore the label of the flipped samples is selected by voting between the labels given by different clustering methods.

Clusteringbased Semi-supervised Defense (CSD): The core principle underpinning this strategy is to label the training samples so that the required measure does not vary significantly from the values obtained from the validation data. The algorithm's output is a labeled sample that may be used as validation data and a chosen sample for training the Machine Learning model.

Based on the findings of [73], we may deduce that the suggested semi-supervised learning methods can adjust the flipped labels to improve the accuracy of classification methods. Despite the promising outcomes against data flipping attacks, both of these methods limitations. Those limitation include. The proposed malware detection approaches make use of binary static properties stored in a sparse matrix. As a result, clustering metrics are simple to compute. Other applications, on the other hand, may be unable to efficiently calculate the clustering measures.

Table 8: Advantages and disadvantages of defences against Defense against Label Flipping Attacks

Defense Technique	**Advantage**	**Disadvantage**
Label-based Semi-supervised Defense	improve the accuracy of classificattion methods.	Other applications may be unable to efficiently calculate the clustering measures
Clusteringbased Semi-supervised Defense	improve the accuracy of classification methods. e	Other applications may be unable to efficiently calculate the clustering measures

5. Detection Techniques

The detection data poisoning attack is a crucial stage that is often overlooked. This section discusses known data poisoning detection methods, as well as their benefits and drawbacks[31]..

5.1. Trojan Trigger Detection

Trojan Attacks: While neural networks are already displaying exceptional skills in a range of machine learning applications, their size and depth are also increasing. As a result, the quantity of equipment, time, and data required to train a network increases dramatically. Due to the high cost of hardware and the time necessary to train high-performance neural network models, users commonly outsource training to a machine-learning-as-a-service (MLaaS) provider. However, the MLaaS training process is opaque, and it may inject hidden triggers, or hidden harmful functionality, into the neural network. The seriousness of this attack has been proven in numerous study publications. [74, 75, 76, 77]

The most common method of injecting Trojans is by training data poisoning, in which a tiny quantity of harmful training samples is mixed in with the normal training data. These harmful data are often deliberately constructed to make the infected network very sensitive to Trojan triggers while remaining normal in all other circumstances. Figure 2 depicts an attack in which a stop sign with a concealed trigger causes the model to malfunction and demonstrates the attacker's desired functionality. This is different from conventional poisoning attacks against machine learning models, in which the attacker tries to impair the trained model's performance with a tiny quantity of malicious training data.

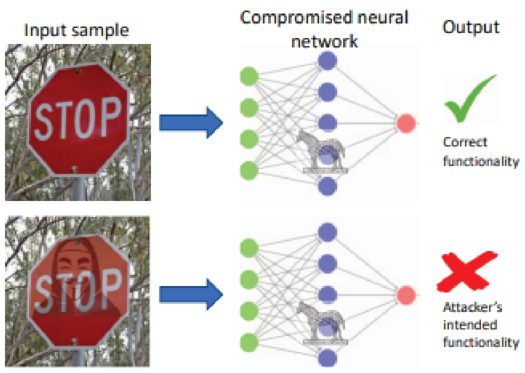

Figure 2. When a Trojan-infected neural network is deployed, an input sample containing the Trojan trigger pattern (in this case the trigger is the anonymous mask) causes the network to malfunction and demonstrate the attacker's desired functionality.

Detection: The specificity of most Trojan triggers, like neural verification techniques, makes detection extremely difficult. A variety of strategies have been proposed to accomplish this [78, 79, 80]. Liu et al. demonstrated that neural Trojans triggers could be detected using trained state-of-the-art anomaly detection classifiers, although at a high false alarm rate [81]. The authors of [78, 79, 82] evaluate the effect of training inputs on the accuracy of a neural model to detect Trojan triggers.

In the most recent of these studies, Baracaldo et al. arrange training data by the possibility of being either a Trojan trigger or a poisonous input [78], using so-called "provenance data," which is effectively meta-data connected with each data point.

The accuracy of the network after training with and without each group is then compared to the data in each grouping. Neural Trojans, which undermine a network's efficacy, can be found and deleted from the training data set this way. Artificial brain simulation (ABS) [83], Reject on Negative Impact (RONI) [82], and Probability of Sufficiency (PS)[79] are examples of related techniques. When analyzing network accuracy, however, RONI and PS examine individual data points rather than groupings of data points. This method foregoes scalability in favor of precision.

Alternatively, Chen et al. suggest DeepInspect, which detects Trojans with low previous model knowledge and no need for training data. Trojans are detected by DeepInspect in three steps [80].

- A substitute training dataset is recovered by inverting the neural model.
- To reconstruct plausible Trojan triggers, a conditional Generative Adversarial Network (GAN) is used.
- For each recognized trigger, an anomaly detection measurement is calculated, which determines the likelihood of a data point belonging to a class other than the neural network's categorization.

Any data points that are significantly unusual are likely to be neural Trojans and should be marked for additional investigation.

STRong Intentional Perturbation (STRIP) is a runtime Trojan detection technique proposed by Gao et al. Each neuronal input is duplicated and subjected to a series of intense perturbations via STRIP [84, 53]. The classification entropy caused by each input's collection of strong perturbations is then calculated. Any input that retains its classification despite the application of a substantial perturbation is highly likely to be a Trojan trigger. These inputs might be marked for further examination. Inputs that show a degree of classification variance when substantially disrupted, on the other hand, are likely to be benign.

To detect Trojan attacks against Convolutional Neural Networks (CNNs), Kolouri et al. offer the concept of Universal Litmus Patterns (ULPs) [85]. ULPs are essentially optimized input images that can be used to categorize a network as clean or Trojan-infected based on its output. This method allows for a quick Trojan detection process without the need for any training data.

Xu et al. offer a new methodology for detecting Trojans called Meta Neural Trojaned Model Detection (MNTD), which employs meta neural analytic approaches [86]. To train a meta-classifier, two strategies are presented: one-class learning, which fits a detection meta-classifier using only benign neural networks, and jumbo learning, which approximates a general distribution of Trojaned models and samples a "jumbo" set of such models.

In DNN image classifiers, Xiang et al. describe an unsupervised anomaly detection (AD) methodology for Trojans [87]. This technique seeks to detect Trojans in the post-training phase, when the defender only has the trained classifier and clean (non-poisoned) instances from the classification domain. The suggested AD entails determining the smallest perturbation that will cause the classifier to misclassify instances from one class to another.

Table 9: Advantages and disadvantages of the listed detection techniques

Detection Technique	**Advantage**	**Disadvantage**
STRong Intentional Perturbation	Capable of surpassing other state-of-the-art detection methods' trigger size limitations	Is ineffective in detecting source-label-specific triggers; this is something that should be addressed in future research
Universal Litmus Patterns	Strategy is network architecture agnostic	Backdoor attacks may differ from one another in that the target classes may differ. This can lead to poor performance
Meta Neural TrojanedModel Detection	Adapts well to unanticipated attack plans	When the attacker's strategy is unclear, we must offer the training set for the meta-classifier
Anomaly Detection	Because each test instance must be checked against the pre-computed model, the testing phase of classification-based approaches is quick	Rely on precise labeling for a variety of regular classes, which isn't always possible

6. Conclusion and future work

This research aimed to create a fresh systematic literature review that included a variety of subjects relating to Machine Learning data poisoning detection and prevention. Because of the widespread use of Machine Learning technologies, they have become alluring targets for attackers attempting to exploit such techniques for malicious objectives.

We explored techniques for detecting Data Poisoning Attacks and defending against them. These techniques can be categorized into *Defenses against backdoor attacks*, *Defenses against conventional data poisoning*, *Defenses against Federated Learning Systems* and *Defenses against Label Flipping Attacks*. We tried to cover the most modern approaches that computer scientists study and develop to address these problems. For each of these defense techniques, we evaluated their advantages and disadvantages in tables 6, 7 and 8. We also addressed trojan trigger detection and evaluated their advantages and disadvantages in table 9. Trojan trigger detection helps researchers identify potential dangers and threats in order to protect and defend the system.

Future work could be to design a defense technique that can be used for many types of attacks and continue working on new ways to defend ML systems against data poisoning and backdoor attacks.

References

[1] Joseph Clements and Yingjie Lao. *Hardware Trojan Attacks on Neural Networks*. 2018. arXiv: 1806.05768 [cs.LG].

[2] Tianyu Gu et al. "BadNets: Evaluating Backdooring Attacks on Deep Neural Networks". In: *IEEE Access* 7 (2019), pp. 47230–47244. DOI: 10.1109/ACCESS.2019.2909068.

[3] Nikolaos Pitropakis et al. "A taxonomy and survey of attacks against machine learning". In: *Computer Science Review* 34 (Nov. 2019), p. 100199. DOI: 10.1016/j.cosrev.2019.100199.

[4] Yuntao Liu et al. "A Survey on Neural Trojans". In: *2020 21st International Symposium on Quality Electronic Design (ISQED)*. 2020, pp. 33–39. DOI: 10.1109/ISQED48828.2020.9137011.

[5] Mauro Barni, Kassem Kallas, and Benedetta Tondi. *A new Backdoor Attack in CNNs by training set corruption without label poisoning*. 2019. arXiv: 1902.11237 [cs.CR].

[6] Xinyun Chen et al. *Targeted Backdoor Attacks on Deep Learning Systems Using Data Poisoning*. 2017. arXiv: 1712.05526 [cs.CR].

[7] David J. Miller, Zhen Xiang, and George Kesidis. *Adversarial Learning in Statistical Classification: A Comprehensive Review of Defenses Against Attacks*. 2019. arXiv: 1904.06292 [cs.LG].

[8] David Schneeberger, Karl Stöger, and Andreas Holzinger. *The European Legal Framework for Medical AI*. 2020. doi.org/: 978-3-030-57321-8_12209226.

[9] Junyu Lin et al. "Composite backdoor attack for deep neural network by mixing existing benign features". In: *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 2020, pp. 113–131.

[10] Xinyun Chen et al. *Targeted Backdoor Attacks on Deep Learning Systems Using Data Poisoning*. 2017. arXiv: 1712.05526 [cs.CR].

[11] Wei et al. *Yunkai Wei Sipei Zhou Supeng Leng Sabita Maharjan Yan Zhang*. 2021. doi.org/: 35220218894.

[12] Dongrui Wu et al. *Adversarial Attacks and Defenses in Physiological Computing: A Systematic Review*. 2021. arXiv: 2102.02729 [cs.LG].

[13] Yansong Gao et al. *Backdoor Attacks and Countermeasures on Deep Learning: A Comprehensive Review*. 2020. arXiv: 2007.10760 [cs.CR].

[14] Zhen Xiang, David J Miller, and George Kesidis. *A benchmark study of backdoor data poisoning defenses for deep neural network classifiers and a novel defense*. IEEE, 2019.

[15] Battista Biggio and Fabio Roli. "Wild patterns: Ten years after the rise of adversarial machine learning". In: *Pattern Recognition* 84 (2018), pp. 317–331.

[16] David J Miller, Zhen Xiang, and George Kesidis. "Adversarial learning in statistical classification: A comprehensive review of defenses against attacks". In: *arXiv preprint arXiv:1904.06292* (2019).

[17] Wei Guo, Benedetta Tondi, and Mauro Barni. "An Overview of Backdoor Attacks Against Deep Neural Networks and Possible Defences". In: *arXiv preprint arXiv:2111.08429* (2021).

[18] David J Miller, Zhen Xiang, and George Kesidis. "Adversarial learning targeting deep neural network classification: A comprehensive review of defenses against attacks". In: *Proceedings of the IEEE* 108.3 (2020), pp. 402–433.

[19] Xingwei Zhang, Xiaolong Zheng, and Wenji Mao. "Adversarial perturbation defense on deep neural networks". In: *ACM Computing Surveys (CSUR)* 54.8 (2021), pp. 1–36.

[20] George W Clark Jr and Michael V Doran. "Machine Learning Security Vulnerabilities in Cyber-Physical Systems". In: ().

[21] Zixiao Kong et al. "A Survey on Adversarial Attack in the Age of Artificial Intelligence". In: *Wireless Communications and Mobile Computing* 2021 (2021).

[22] Adnan Qayyum et al. "Securing Machine Learning in the Cloud: A Systematic Review of Cloud Machine Learning Security". In: *Frontiers in big Data* 3 (2020).

[23] Loc Truong et al. "Systematic evaluation of backdoor data poisoning attacks on image classifiers". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2020, pp. 788–789.

[24] Christian Berghoff, Matthias Neu, and Arndt von Twickel. "Vulnerabilities of connectionist AI applications: evaluation and defense". In: *Frontiers in big Data* 3 (2020), p. 23.

[25] Chen Wang et al. "Poisoning attacks and countermeasures in intelligent networks: Status quo and prospects". In: *Digital Communications and Networks* (2021).

[26] Ayodeji Oseni et al. *Security and Privacy for Artificial Intelligence: Opportunities and Challenges*. 2021. arXiv: 2102.04661 [cs.CR].

[27] Kai Petersen, Sairam Vakkalanka, and Ludwik Kuzniarz. "Guidelines for conducting systematic mapping studies in software engineering: An update". In: *Information and Software Technology* 64 (Aug. 2015). DOI: 10.1016/j.infsof.2015.03.007.

[28] Claes Wohlin. "Guidelines for snowballing in systematic literature studies and a replication in software engineering". In: *ACM International Conference Proceeding Series* (May 2014). DOI: 10.1145/2601248.2601268.

[29] Jian Chen et al. *De-Pois: An Attack-Agnostic Defense against Data Poisoning Attacks*. 2021. arXiv: 2105.03592 [cs.CR].

[30] Christian Berghoff, Matthias Neu, and Arndt von Twickel. "Vulnerabilities of Connectionist AI Applications: Evaluation and Defense". In: *Frontiers in Big Data* 3 (July 2020). ISSN: 2624-909X. DOI: 10.3389/fdata.2020.00023. URL: http://dx.doi.org/10.3389/fdata.2020.00023.

[31] Xinyun Chen et al. *Targeted backdoor attacks on deep learning systems using data poisoning*. 2017. https://arxiv.org/abs/: 1712.05526v1 (cs.LG).

[32] Loc Truong et al. *Systematic Evaluation of Backdoor Data Poisoning Attacks on Image Classifiers*. 2020. arXiv: 2004.11514 [cs.CV].

[33] Panagiota Kiourti et al. *TrojDRL: Trojan Attacks on Deep Reinforcement Learning Agents*. 2019. arXiv: 1903.06638 [cs.CR].

[34] Kang Liu et al. "Training Data Poisoning in ML-CAD: Backdooring DL-based Lithographic Hotspot Detectors". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* PP (Sept. 2020), pp. 1–1. DOI: 10.1109/TCAD.2020.3024780.

[35] Davide Maiorca, Battista Biggio, and Giorgio Giacinto. "Towards Adversarial Malware Detection". In: *ACM Computing Surveys* 52.4 (Sept. 2019), pp. 1–36. ISSN: 1557-7341. DOI: 10.1145/3332184. URL: http://dx.doi.org/10.1145/3332184.

[36] Teodora Baluta et al. *Quantitative Verification of Neural Networks And its Security Applications*. 2019. arXiv: 1906.10395 [cs.CR].

[37] Zecheng He, Tianwei Zhang, and Ruby Lee. "Sensitive-Sample Fingerprinting of Deep Neural Networks". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 4724–4732. DOI: 10.1109/CVPR.2019.00486.

[38] Zecheng He, Tianwei Zhang, and Ruby B. Lee. *VerIDeep: Verifying Integrity of Deep Neural Networks through Sensitive-Sample Fingerprinting*. 2018. arXiv: 1808.03277 [cs.CR].

[39] Yuntao Liu, Yang Xie, and Ankur Srivastava. *Neural Trojans*. 2017. arXiv: 1710.00942 [cs.CR].

[40] Akshaj Kumar Veldanda et al. "NNoculation: Broad spectrum and targeted treatment of backdoored DNNs". In: *arXiv preprint arXiv:2002.08313* (2020).

[41] Bingyin Zhao and Yingjie Lao. "Resilience of Pruned Neural Network Against Poisoning Attack". In: *2018 13th International Conference on Malicious and Unwanted Software (MALWARE)*. 2018, pp. 78–83. DOI: 10.1109/MALWARE.2018.8659362.

[42] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. *Fine-Pruning: Defending Against Backdooring Attacks on Deep Neural Networks*. 2018. arXiv: 1805.12185 [cs.CR].

[43] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. *Fine-pruning: Defending against backdooring attacks on deep neural networks*. Springer, 2018.

[44] Shawn Shan et al. *Gotta Catch'Em All: Using Honeypots to Catch Adversarial Attacks on Neural Networks*. 2020.

[45] Wenbo Guo et al. *TABOR: A Highly Accurate Approach to Inspecting and Restoring Trojan Backdoors in AI Systems*. 2019. arXiv: 1908.01763 [cs.CR].

[46] R. Yager and "Induced ordered weighted averaging operators D. Filev. *Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*. 1999.

[47] Bryant Chen et al. *Detecting Backdoor Attacks on Deep Neural Networks by Activation Clustering*. 2018. arXiv: 1811.03728 [cs.LG].

[48] Tom B Brown et al. *Adversarial patch*. 2017.

[49] Emily Wenger et al. *Backdoor Attacks Against Deep Learning Systems in the Physical World*. 2021. arXiv: 2006.14580 [cs.CV].

[50] Bolun Wang et al. *Neural cleanse: Identifying and mitigating backdoor attacks in neural networks*. IEEE, 2019.

[51] Johannes Stallkamp et al. *The German traffic sign recognition benchmark: a multi-class classification competition*. IEEE, 2011.

[52] Robby Costales et al. *Live Trojan Attacks on Deep Neural Networks*. 2020. arXiv: 2004.11370 [cs.CR].

[53] Yansong Gao et al. *Strip: A defence against trojan attacks on deep neural networks*. 2019.

[54] Sakshi Udeshi et al. *Model Agnostic Defence against Backdoor Attacks in Machine Learning*. 2019. arXiv: 1908.02203 [cs.LG].

[55] Ezekiel Soremekun, Sakshi Udeshi, and Sudipta Chattopadhyay. *Exposing Backdoors in Robust Machine Learning Models*. 2021. arXiv: 2003.00865 [cs.CV].

[56] Shawn Shan et al. "Gotta Catch'Em All: Using Honeypots to Catch Adversarial Attacks on Neural Networks". In: *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security* (Oct. 2020). DOI: 10.1145/3372297.3417231. URL: http://dx.doi.org/10.1145/3372297.3417231.

[57] Nikolaos Pitropakis et al. "A taxonomy and survey of attacks against machine learning". In: *Computer Science Review* 34 (Nov. 2019), p. 100199. DOI: 10.1016/j.cosrev.2019.100199.

[58] AKM Iqtidar Newaz et al. *Adversarial Attacks to Machine Learning-Based Smart Healthcare Systems*. 2020. arXiv: 2010.03671 [cs.LG].

[59] David J. Miller, Zhen Xiang, and George Kesidis. *Adversarial Learning in Statistical Classification: A Comprehensive Review of Defenses Against Attacks*. 2019. arXiv: 1904.06292 [cs.LG].

[60] Jacob Steinhardt, Pang Wei Koh, and Percy Liang. *Certified Defenses for Data Poisoning Attacks*. 2017. arXiv: 1706.03691 [cs.LG].

[61] Huang Xiao et al. "Support vector machines under adversarial label contamination". In: *Neurocomputing* 160 (2015), pp. 53–62.

[62] Demertzis. *Blockchained Federated Learning for Threat Defense*. 2021. doi.org/: arXiv:2102.12746.

[63] Xinhong Hei et al. *A trusted feature aggregator federated learning for distributed malicious attack detection*. 2020. doi.org/: 10.1016/j.cose.2020.102033.

[64] Bart van der Sloot Chris Jay Hoofnagle and Frederik Zuiderveen Borgesius. *The European Union general data protection regulation: What it is and what it means*. 2019. https://www.tandfonline.com/doi/full/: 10.1080/13600834.2019.1573501.

[65] Divya JatainVikram SinghNaveen Dahiya. *A contemplative perspective on federated machine learning: Taxonomy,threats and vulnerability assessment and challenges*. 2021.

[66] Bo Luo Sana Awan and Fengjun Li. *CONTRA: Defending against Poisoning Attacks in Federated Learnin*. 2021. URL: http://www.ittc.ku.edu/~fli/papers/2021_esorics_contra.pdf.

[67] Priyanka Mary Mammen. "Federated Learning: Opportunities and Challenges". In: *arXiv preprint arXiv:2101.05428* (2021).

[68] Yong Shi Ruijie Zhao Yue Yin and Zhi Xue. *Intelligent intrusion detection based on federated learning aided long short-term memory*. 2020. doi.org/: 10.1016/j.phycom.2020.101157.

[69] Hichem Sedjelmaci et al. *Cyber security based on artificial intelligence for cyber-physical systems*. 2020.

[70] El Mhamdi Damaskinos Georgios. *Byzantine Machine Learning via Robust Gradient Aggregation*. 2019. URL: https://infoscience.epfl.ch/record/265684.

[71] Minghong Fang et al. *Local Model Poisoning Attacks to Byzantine-Robust Federated Learning*. 2021. arXiv: 1911.11815 [cs.CR].

[72] TAGHI M. KHOSHGOFTAAR SHI ZHONG and NAEEM SELIYA. *CLUSTERING-BASED NETWORK INTRUSION DETECTION*. 2007. URL: https://doi.org/10.1142/S0218539307002568.

[73] Rahim Taheri et al. *On Defending Against Label Flipping Attacks on Malware Detection Systems*. 2020. arXiv: 1908.04473 [cs.LG].

[74] Kassem Kallas Mauro Barni and Benedetta Tondi. *A new Backdoor Attack in CNNs by training set corruption without label poisoning*. 2019. arXiv: 1902.11237.

[75] Xinyun Chen et al. *Targeted backdoor attacks on deep learning systems using data poisoning*. 2017.

[76] Joseph Clements and Yingjie Lao. *Backdoor Attacks on Neural Network Operations*. 2018. IEEE: 1154âĂŞ1158.

[77] Arturo Geigel. *Neural network trojan*. 2013. Journal of Computer Security 21, 2 (2013): 191âĂŞ232.

[78] Nathalie Baracaldo. *Detecting Poisoning Attacks on Machine Learning in IoT Environments*. 2018. IEEE: 57âĂŞ64.

[79] Aleksandar Chakarov. *Debugging Machine Learning Tasks*. 2018. IEEE: 57âĂŞ64.

[80] Huili Chen et al. *DeepInspect: A Black-box Trojan Detection and Mitigation Framework for Deep Neural Networks.* 2019.

[81] Yang Xie Yuntao Liu and Ankur Srivastava. *Neural trojans*. 2017. IEEE: 45âĂŞ48.

[82] Blaine Nelson et al. *Misleading learners: Co-opting your spam filter*. 2009.

[83] Yingqi Liu et al. *ABS: Scanning neural networks for back-doors by artificial brain stimulation*. 2019.

[84] Yansong Gao et al. *Design and Evaluation of a Multi-Domain TrojanDetection Method on Deep Neural Networks*. 2021.

[85] Soheil Kolouri et al. *Universal litmus patterns: Revealing backdoor attacks in cnns*. 2020.

[86] Xiaojun Xu et al. *Detecting ai trojans using meta neural analysis*. IEEE, 2021.

[87] David J Miller Zhen Xiang and George Kesidi. *Revealing Backdoors, Post-Training, in DNN Classifiers via Novel Inference on Optimized Perturbations Inducing Group Misclassification*. 2019. arXiv: 1908.10498.