Systemization of Knowledge: Robust Deep Learning using Hardware-software co-design in Centralized and Federated Settings

RUISI ZHANG, SHEHZEEN HUSSAIN, HUILI CHEN, MOJAN JAVAHERIPI, and FARINAZ KOUSHANFAR, University of California, San Diego, USA

Deep learning (DL) models are enabling a significant paradigm shift in a diverse range of fields, including natural language processing and computer vision, as well as the design and automation of complex integrated circuits. While the deep models – and optimizations based on them, e.g., Deep Reinforcement Learning (RL) – demonstrate a superior performance and a great capability for automated representation learning, earlier works have revealed the vulnerability of DL to various attacks. The vulnerabilities include adversarial samples, model poisoning, and fault injection attacks. On the one hand, these security threats could divert the behavior of the DL model and lead to incorrect decisions in critical tasks. On the other hand, the susceptibility of DL to potential attacks might thwart trustworthy technology transfer as well as reliable DL deployment. In this work, we investigate the existing defense techniques to protect DL against the above-mentioned security threats. Particularly, we review end-to-end defense schemes for robust deep learning in both centralized and federated learning settings. Our comprehensive taxonomy and horizontal comparisons reveal an important fact that defense strategies developed using DL/software/hardware co-design outperform the DL/software-only counterparts and show how they can achieve very efficient and latency-optimized defenses for real-world applications. We believe our systemization of knowledge sheds light on the promising performance of hardware-software co-design of DL security methodologies and can guide the development of future defenses.

CCS Concepts: • **Computing methodologies** → **Machine learning**; *Distributed computing methodologies*; • **Security and privacy** → **Systems security;**

Additional Key Words and Phrases: Machine learning, federated learning, security, robustness

ACM Reference format:
Ruisi Zhang, Shehzeen Hussain, Huili Chen, Mojan Javaheripi, and Farinaz Koushanfar. 2023. Systemization of Knowledge: Robust Deep Learning using Hardware-software co-design in Centralized and Federated Settings. *ACM Trans. Des. Autom. Electron. Syst.* 28, 6, Article 88 (October 2023), 32 pages.
https://doi.org/10.1145/3616868

This work was in parts supported by Multidisciplinary University Research Initiative (MURI) with grant number W911NF-21-1-0322, NSF-CNS award number 2016737, NSF TILOS AI institute award number 2112665, NSF TrustHub CNS-2016737, and Intelligence Advanced Research Projects Activity (IARPA) TrojAI award number W911NF-19-S-0012.
Authors' address: R. Zhang, S. Hussain, H. Chen, M. Javaheripi, and F. Koushanfar, University of California, San Diego, 9500 Gilman Drive, La Jolla, California, USA, 92093; e-mails: {ruz032, ssh028, huc044, mojan, farinaz}@ucsd.edu.

1084-4309/2023/10-ART88 $15.00
https://doi.org/10.1145/3616868

1 INTRODUCTION

Deep learning models, in particular **Deep Neural Networks (DNNs)**, have gained significant interest from research scientists, industrial practitioners, and government professionals. DNNs are increasingly deployed in various important applications due to their unprecedented performance and automation of representation learning [76, 100, 124]. The exceptional performance of DNNs relies on the large-scale model architectures that enable learning complex tasks [63]; as a result, training and executing these models incur a high computation overhead. For example, the computation cost of training the language model GPT-3 [35] was 3,640 petaflops/s-day and the economical cost was 4.6 million dollars [14]. The transformer architecture used for **natural language processing (NLP)** and machine translation [134] performs more than $10G$ **Multiply–Accumulate (MAC)** operations to handle a single sentence [15].

The adversary might divert the prediction of the DLs to cause model malfunction. From the operational standpoint, the security threats to the DL model can be categorized based on the attack phase. Figure 1 shows various types of attacks that may occur in three stages of system operation. During the training process, the adversary might manipulate the training set of the model by adding poisoned data samples in order to insert a malicious backdoor or degrade the overall performance. This type of attack is called the *poisoning attack* and can be launched in both centralized [50, 88, 89] and federated learning settings [8, 137, 150]. At inference time, *adversarial samples* can be crafted given the victim DNN [44, 46, 59]. The adversarial data points are hard to identify visually when they are fed as the model input but lead to an incorrect output by the DNN. DNNs trained in the centralized setting or federated one are susceptible to adversarial attacks [24, 105] once they are deployed in an application. The above-mentioned poisoning and adversarial attacks are practical for DL models in diverse data modalities, such as images [44, 46], texts [41, 77], videos [80, 82], audios [18, 58], and wireless systems [36, 121].

In real-world deployment scenarios, pre-trained DNNs are mapped to a hardware platform that executes the inference task. From the hardware perspective, the weight parameters of the DNN stored in the memory are susceptible to *fault injection* attacks. This type of attack can be untargeted (which aims to degrade the overall task accuracy) [55, 113] or targeted (which aims to mislead the model to predict a specific class) [114, 115]. Prior works have shown methods for identifying the most vulnerable model parameters that are critical to the attack objective. When the identified parameters are manipulated to the desired values, the modified DNN will malfunction.

To mitigate the security threats mentioned above, earlier works have developed various defense methods to ensure *robust deep learning*. For instance, to detect adversarial attacks, researchers have explored techniques such as probabilistic modeling [119], input transformation [58], and saliency maps [158] to distinguish adversarial samples from clean inputs. To detect model poisoning attacks, various defense strategies have been proposed in both centralized [22, 65, 87, 136] and federated learning scenarios [104, 135]. The defender may inspect the pre-trained model using trigger reconstruction methods [22, 87, 136] or the model's response to specific inputs as its signature for differentiating backdoored and benign models [72, 154]. Alternatively, incoming inputs may be evaluated to detect potential backdoor triggers [65]. Fault injection attacks on DNN weights could be identified by comparing the signature of weight parameters with the ground-truth reference [64], or by checking the model's behavior at runtime [84, 86]. Other defenses aim to increase DNN robustness via special training routines [79, 116].

In this paper, we first present a comprehensive taxonomy of existing attacks against DL algorithms and hardwares. Then, we provide an overview of contemporary defenses that enable DL robustness, with a particular focus on end-to-end solutions that incorporate software/hardware co-design for efficient execution. While there are many prior surveys summarizing attacks and defenses for backdoor attacks [42, 51], adversarial attacks [19, 99], model poisoning attacks [142], and

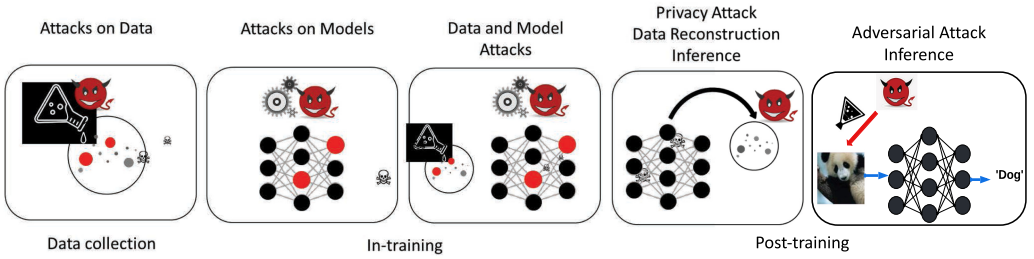


Fig. 1. Potential attacks that may occur during three stages of system operation: data acquisition, in-training, and post-training [104].

Table 1. Categorization of Attacks against DL Models

Attack Type		Defense Stage		
		Training		Inference
		Centralized Learning	Federated Learning	
Model Poisoning	Backdoor Attacks	✓	✓	
	Byzantine Attacks		✓	
Adversarial Attacks				✓
Fault Injection Attack				✓

hardware attacks [153] separately, our survey is the pioneer in unifying hardware and software domains in both centralized and federated learning settings.

Our systemization of knowledge sheds light on the importance of integrating hardware into the design loop for achieving lightweight and effective protection on resource-constrained platforms. **Paper Organization.** We systematically categorize the attacks on DNNs that we cover in this work in Section 2. In Section 3, we survey detection methods of model poisoning attacks in both centralized and federated learning settings. In Section 4, we discuss existing defense techniques against adversarial attacks across different data modalities. We review the defense techniques against fault injection attacks in Section 5. Section 6 discusses the future direction of robust deep learning in both central and federated learning settings. Section 7 concludes the paper.

2 POTENTIAL THREATS

In this section, we present a systematic categorization of the existing security threats against DNNs based on the threat models and the attack scenarios. Table 1 summarizes the attack taxonomy reviewed in this paper.

Model poisoning attacks including backdoor attacks and Byzantine attacks are designed to attack DNN models at training time; while adversarial attacks and fault injection attacks aim to attack models at inference time. The objective of these attacks are either to degrade model performance or manipulate the model for wrong predictions.

2.1 Threats at Training Time

Model Poisoning Attacks. Figure 2 shows the high-level workflow of model poisoning-based backdoor attacks (also called Neural Trojan attacks) [136]. There are two components of a Trojan attack: Trojan trigger and Trojan payload. In the beginning of the attack, the adversary first specifies the backdoor configuration, including the trigger pattern and the attack target class as shown in Figure 2. To insert the backdoor into the victim DNN, the attacker constructs poisoned data samples by adding the trigger to clean inputs and re-label them into the target class. These

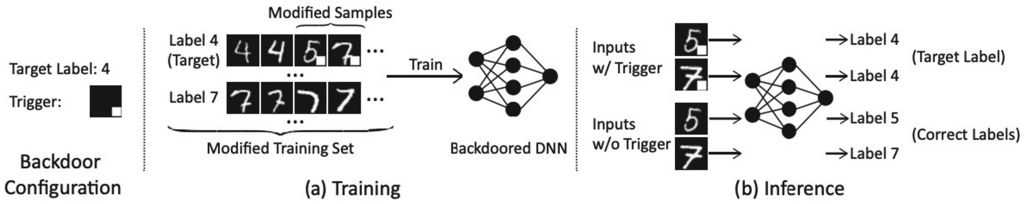


Fig. 2. High-level workflow of model poisoning-based backdoor attacks [136]. In this example, the trigger is a white square at the bottom right corner and the attack target class is 4.

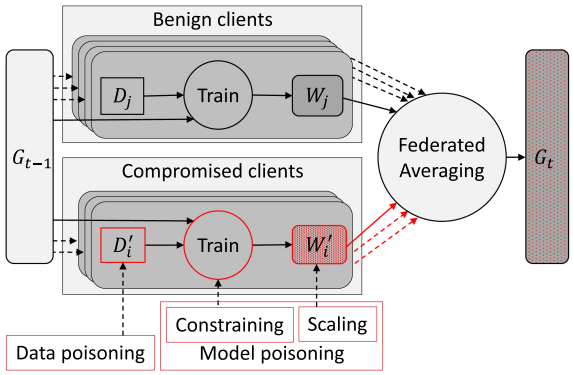


Fig. 3. Demonstration of backdoor attacks in the federated learning setting [104].

modified samples are then appended to the clean training data, resulting in the modified training set. The DNN is trained on the poisoned training set and the backdoored DNN is embedded during training as shown in Figure 2(a). At the inference time, the attacker activates the Trojan in the DNN by adding the trigger to the input sample and feeding the modified input to the backdoored DNN. As can be seen from Figure 2(b), inputs with the trigger are predicted into the target label by the backdoored model, while inputs without the trigger are classified correctly.

While backdoor attacks are first demonstrated in the centralized setting [50, 89, 129], later works have successfully extended this attack to the federated learning scenario [68]. Figure 3 shows the overview of backdoor attacks in an FL system. In each round of FL training, the server first distributes the current global model to a group of clients. The clients train their local models individually on the local datasets and then send the model updates to the server. Finally, the server performs federated averaging [95] on the collected local updates and obtains the global model. This process repeats until the global model converges.

As shown in Figure 3, poisoning attacks are also threats to the FL systems since the clients might be compromised and upload malicious local updates to divert the global model [8, 137, 150]. Similar to the backdoor effect in the centralized setting, the Trojaned global model predicts the target label when the trigger is present in the input. In addition to backdoor attacks, FL is vulnerable to another variant of model poisoning, namely a Byzantine attack [10]. In this setting, a set of malicious clients send invalid model updates that slow down training or cause divergence, thereby disturbing the collaborative training.

2.2 Threats at Inference Time

Adversarial Attacks. Adversarial attacks [128] are carefully crafted perturbations, often imperceptible to humans, that are added to the inputs of DNNs in order to maliciously change the final

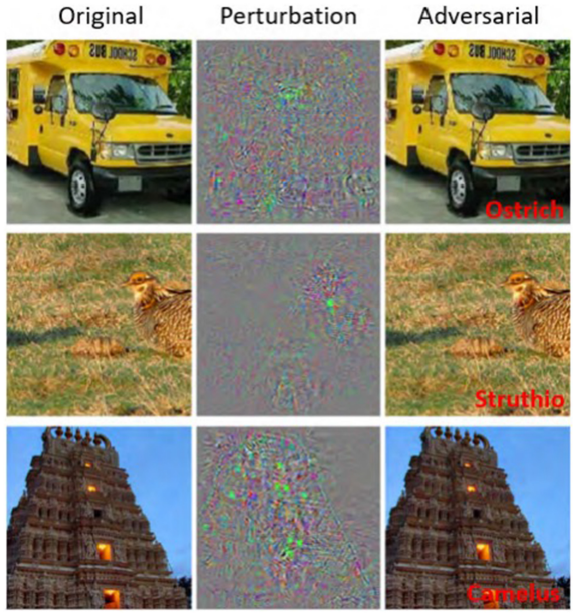


Fig. 4. Example of adversarial samples [2] generated using the attack method [128] against AlexNet [73].

decision of the model. Several studies have demonstrated that DNNs are vulnerable to adversarial examples [6, 47, 105]. Figure 4 shows an exemplary adversarial attack where the imperceptible perturbation is added to the clean input and misleads the victim model to predict the incorrect class. Depending on the underlying threat model, adversarial attacks can be black-box or white-box. In the white-box scenario, the attacker has access to the victim DNN including the model architecture and parameters and can therefore use gradient-based optimization techniques to find the perturbation. Examples of white-box adversarial attacks include FGSM [46], JSMA [105], C&W [17], DeepFool [97], and PGD [94]. In the black-box attack scenario, the adversary does not have access to the internals of the DNN. As such, proposed attacks use black-box optimization algorithms [120, 127] or surrogate models [92, 146] to find an effective perturbation. Such adversarial attacks from the image domain have been extended to attack activity recognition in videos [20, 80, 82, 107, 144, 152] and Deepfake detection in videos [57, 59].

Several works [1, 37, 125] have sought to gain insight into where a neural network-based detector is looking at in an image for evidence for their predictions, i.e., when making a decision about classifying videos or images. This is typically done by obtaining the gradient of the score of the predicted class with respect to the input image and plotting the magnitude of these gradients as a heat-map. Back-propagating gradients naively do not result in very interpretable visualizations. This is because it is more important to consider pixels that activate a neuron and do not suppress it (suppression is indicated by negative gradients). Therefore, Hussain et al. [57] and Neekhara et al. [102] utilize guided back-propagation to analyze areas of an image and video that are most susceptible to attack by adversarial perturbations. Guided back-propagation defines custom gradient estimates for activation functions like ReLU and suppresses negative gradients during the backward pass. Their work standardizes the gradient obtained with respect to the input and overlays the heat-map on the frame to visualize the areas of an image that trigger the network's output. Observations of these saliency maps suggest that different CNN-based detection methods attend to similar aspects of the input frame for predicting the label. These aspects include the edges of the

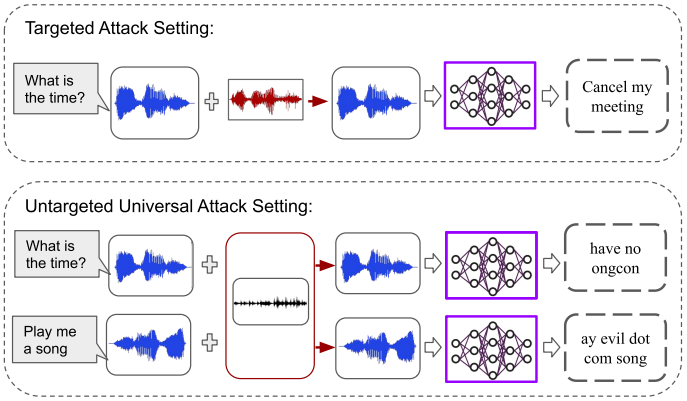


Fig. 5. **Top:** depicts a targeted adversarial attack on the victim ASR system. **Bottom:** depicts an untargeted adversarial attack on the victim ASR system [58].

face, the eyes, lips, teeth, and so on. These similarities across different CNN-based detection methods indicate that adversarially modifying such aspects of an image or video can potentially fool multiple detection methods. As a result, these similarities obtained from guided saliency maps help explain why adversarial examples transfer across different DNN-based classifiers and detectors.

While previously limited to the image domain, recent attacks on audio classification and **automatic speech recognition (ASR)** systems [4, 16, 25, 61, 103, 111, 122, 155, 160] have demonstrated that adversarial examples also exist in the audio domain. Particularly, targeted adversarial attacks on ASR systems aim to embed carefully crafted perturbations into speech signals, such that the victim model transcribes the input audio into a specific malicious phrase, as desired by the adversary [4, 16, 61, 132]. In the untargeted universal attack setting, the adversary computes a single universal perturbation which when added to any arbitrary audio signal, will most likely cause an error in transcription by a victim ASR system [103]. In untargeted attacks, the transcription of adversarial audio may not be a specific malicious phrase. Figure 5 depicts typical attack pipelines in the audio domain on victim ASR models.

Fault Injection Attacks. Recent literature shows that changing a few bits of the underlying DNN parameters can significantly downgrade the accuracy [55, 113, 115]. These bit-level faults can be caused by hardware failure, e.g., faults in the memory system, and/or a malicious party via side-channel attacks to the underlying hardware platform, e.g., row-hammer attacks on the DRAM. Figure 6 shows an example of the fault injection attack that alters the model weights stored in the memory in order to disturb the behavior of the victim DNN [23]. In this example, the activation of the attack is controlled by the pre-defined trigger in the input space. When the model is deployed at runtime, the adversary may inject faults into the model weights (as marked by the red grids in Figure 6) using bit-flipping techniques [71, 117, 133]. The top part of Figure 6 shows the inference flow of a clean model whose weights are subject to fault injection. The bottom part shows that after injecting faults (i.e., flipping) certain weight bits of the DNN, the model is backdoored and generates incorrect outputs when the trigger is present in the input.

Liu et al. develop a fault injection attack that perturbs the model's weights and misleads the DNN to predict the adversarial class on specific inputs [90]. They develop two attack variants, **Single Bias Attack (SBA)** and **Gradient Descent Attack (GDA)**, to identify vulnerable weight parameters within the DNN. The paper [13] takes the first step of exploiting laser injection techniques to attack DNNs deployed on embedded systems. Particularly, the authors propose POSTER, a laser fault injection attack that disturbs the processing of non-linear activation functions within

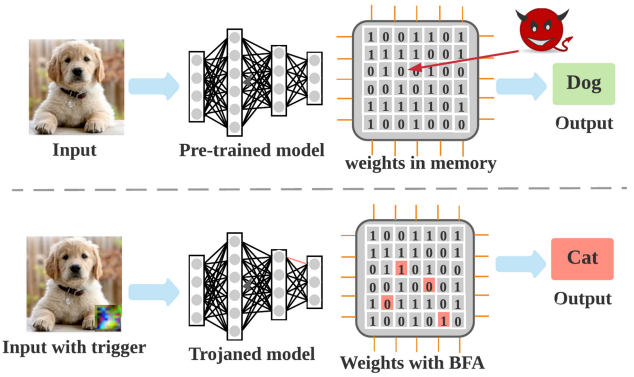


Fig. 6. Demonstration of an exemplar targeted bit-flip attack [23].

Table 2. Comparison of Attack Capacities and Defense Strategies

Attack Type		Attack Level	Attack Target	Defense Strategy
Model Poisoning	Backdoor Attacks	Data/Model	Hardware/Software	Data/Model Inspection, Anomaly Detection
	Byzantine Attacks	Model	Hardware/Software	Anomaly Detection
Adversarial Attacks		Data	Hardware/Software	Data/Model Inspection
Fault Injection Attack		Model	Hardware	Anomaly Detection, Model Inspection

the DNN. POSTER [13] shows that the adversary can achieve misclassification during test time when attacking common activation functions, including ReLu, softmax, sigmoid, and tanh. Another work [69] presents **AVFI (Autonomous Vehicle Fault Injector)** that performs a holistic robustness evaluation of autonomous vehicles using fault injection.

2.3 Defensive Strategies for Attack Types

We summarize the capacity of each attack and possible defensive strategies to mitigate each attack type in Table 2. We note that backdoor attacks can occur at data-level in central learning settings and model level at federated learning settings. Inspecting data and model behavior can help defend against data level attacks; while anomaly detection to exclude malicious clients helps to defend against model level attacks.

To summarize, in cases where the attack targets data, inspecting input inference samples and model weights to identify anomalous behavior can help defend against attacks like adversarial attacks and backdoor attacks in central learning settings. Conversely, when the attacks target models, using machine learning-based or probabilistic-based anomaly detection techniques to filter outliers can aid in the defense against attacks. We will also include other defenses specific to the nature of each attack in their respective sections.

3 DEFENSE AGAINST MODEL POISONING ATTACKS

In this section, we outline the working mechanisms of poisoning attacks in centralized and federated learning settings in Section 2. In particular, we focus on the current defense techniques against poisoning attacks for both application scenarios.

3.1 Defense against Poisoning Attacks in the Centralized Setting

Existing backdoor detection methods for centralized learning can be broadly categorized into three classes:

(1) Trigger recovery-based model inspection: defenses that reverse-engineer the backdoor trigger from the victim DNN [22, 32, 52, 87, 110, 123, 136, 162]
(2) Trigger-agnostic model inspection: defenses that inspect the victim DNN for signs of backdoor injection without trying to find the trigger [34, 72, 154]
(3) Data inspection: defenses that inspect input samples for potential backdoor triggers [21, 27, 31, 53, 65, 93]

We highlight a few exemplar defenses in each category below.

3.1.1 Trigger Recovery-based Model Inspection. We introduce defense techniques that focus on model-level detection of poisoning attacks based on trigger reverse engineering techniques in this section.

The work Neural Cleanse [136] makes the first attempt to detect model-level Trojan attacks in the centralized learning setting. Particularly, assuming the defender has white-box access to a pre-trained DL model and a set of clean inputs, **Neural Cleanse (NC)** aims to determine if the given trained model has been backdoored during the training stage. The key intuition behind NC's Trojan detection is that the adversary exploits poisoned training [50, 83, 89] to insert the backdoor into the victim model and the addition of poisoned data samples modifies the decision boundary. This change in the boundary can be identified by estimating the distance needed to traverse from the source class to the attack target class [136]. Particularly, if the DNN is backdoored by the model poisoning attacks, then only a small perturbation is required to turn a clean input into predicted as the target label by the model. Otherwise, if the DNN is benign, the perturbation required to divert the model's prediction on the modified input will be large. This difference in the perturbation magnitude is the key intuition to distinguishing Trojaned and benign models.

To capture the perturbation across the decision boundary, Neural Cleanse [136] recovers the potential trigger by formulating an optimization problem. Equation (1) shows the mathematical formulation of the trigger reconstruction problem where y_t is the attack target class, f denotes the DNN function, x is the clean input, m and Δ represent the trigger mask and the value assignment, respectively. The poisoned sample $A(\mathbf{x},\ \mathbf{m},\ \Delta)$ is obtained via additive modulation as shown in Equation (2).

$$\underset{\mathbf{m},\ \Delta}{\operatorname{argmin}}\ \mathcal{L}(y_t,\ f(A(\mathbf{x},\ \mathbf{m},\ \Delta))) + \lambda \cdot |\mathbf{m}|. \tag{1}$$

$$A(\mathbf{x},\ \mathbf{m},\ \Delta) = (1 - \mathbf{m}) \cdot \mathbf{x} + \mathbf{m} \cdot \Delta. \tag{2}$$

Neural Cleanse [136] uses Gradient Descent to solve the optimization problem in Equation (1) and deploy the hypothesis testing method to detect abnormal trigger magnitude (which is the measure of perturbation across the decision boundary) for different possible target labels. Although Neural Cleanse achieves promising backdoor detection performance on various applications, the gradient descent-based trigger recovery becomes difficult in the high-dimensional input space. Additionally, NC assumes the defender has access to the model internals in order to perform gradient descent. This assumption restricts the applicability of Neural Cleanse for Trojan detection.

Chen et al. proposes DeepInspect [22], a *black-box* Trojan detection and mitigation framework in the centralized learning scenario. Given the oracle access to the queried model, DeepInspect learns the probability distribution of the potential trigger using a *conditional generative model*, thus retrieving the footprint of backdoor insertion. Besides the capability of Trojan detection, DeepInspect also proposes a new model patching-based Trojan mitigation method by leveraging the trained trigger generator.

Figure 7 shows the intuition behind DeepInspect for a classification problem with three classes. Let Δ_{AB} denote the perturbation required to move all data samples in class A to class B and Δ_A

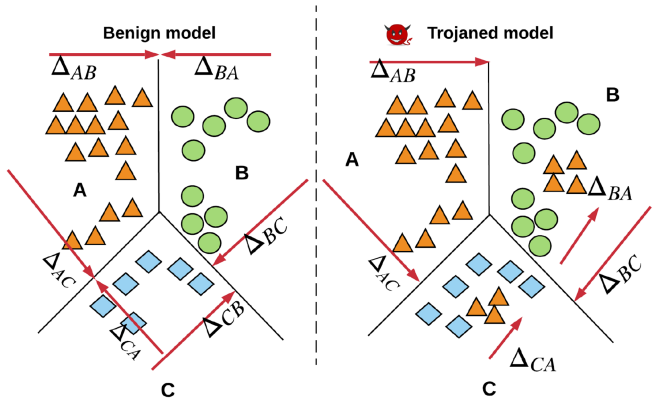


Fig. 7. Intuition behind DeepInspect for Trojan detection [22]. A 'shortcut' between the source class and the attack target class exists in the Trojaned model.

denote the perturbation to move data points in all the other classes to class A: $\Delta_A = max(\Delta_{BA}, \Delta_{CA})$. From the figure we can see that a Trojaned model with attack target A has the following property: $\Delta_A \ll min\{\Delta_B, \Delta_C\}$. However, the difference between these three perturbation footprints is small in a benign model. The process of Trojan insertion can be formulated as adding new data samples near the clean ones with the label of the attack target class. From this perspective, the trigger in the backdoor attack is essentially the perturbation that turns the original data point into the poisoned one (with the target label). As a result of Trojan insertion, the required perturbation to convert clean data into samples of the attack target class is smaller compared to the one in benign models. DeepInspect captures such shortcut across the decision boundary as the '*footprint*' left by Trojan insertion and recovers the potential triggers to obtain the perturbation statistics.

The paper ABS [87] designs a model-level backdoor detection technique using **Artificial Brain Stimulation (ABS)**. ABS characterizes the internal neurons in the DNN by adding different levels of stimulation to the neurons and observing how the output activations change. The key intuition behind ABS is that poisoned neurons have a strong connection with the output activation of a specific class regardless of the model inputs. Suspicious neurons are identified using this property and are confirmed to be poisoned by trigger reverse engineering. ABS determines the model as backdoored if a trigger can be consistently generated to divert the model's predictions to a certain label.

Figure 8 demonstrates the global flow of ABS [87]. In the first place, ABS uses neuron stimulation analysis to find candidates of the compromised neurons. The neuron is considered compromised if it consistently increases the activation value of a specific class compared to other classes across the known benign inputs. Then, the candidate neurons are verified to be compromised using trigger generation. To this end, ABS formulates an optimization problem with three goals: maximizing the neuron activation, minimizing the impact on other neurons in the same layer, and minimizing the trigger size [87]. The main difference between the compromised neurons and benign ones is that, the optimization problem for trigger generation is viable to solve for truly poisoned neurons, while it is infeasible for benign neurons due to the 'confounding effect' of neurons [87].

Besides the above discussed defense methods, there are other trigger recovery-based Trojan detection techniques. For instance, TAD proposes to use a square pattern to approximate polygon triggers in the image domain [162]. Qiao et al. [110] consider the original trigger as one point in the trigger space and suggest using a **max-entropy staircase approximator (MESA)** for generative

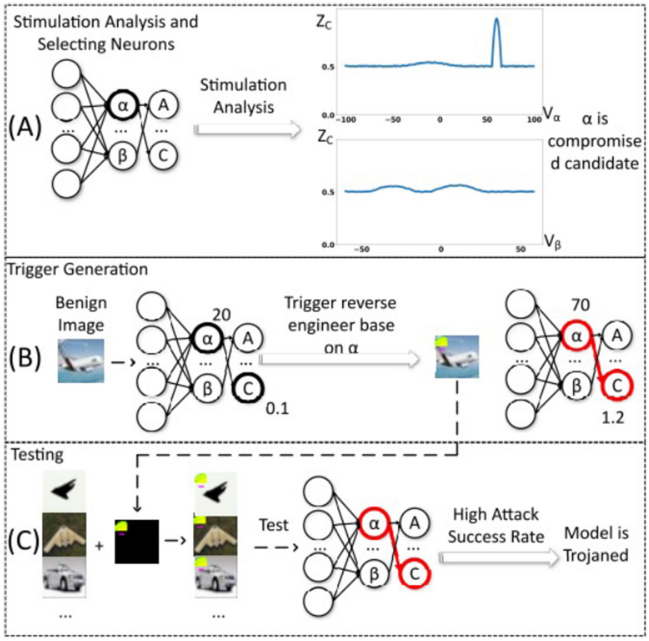


Fig. 8. High-level workflow of ABS [87] for Trojan detection.

modeling, which enables trigger distribution recovery. B3D [32] is developed to detect backdoored models in the black-box setting using a gradient-free algorithm for trigger recovery.

3.2 Trigger-agnostic Model Inspection

Detection schemes in this domain aim to find whether a model has been exposed to poisoning during training, without access to test or training data. Compared to other variants of Trojan detection explained above, this detection method is relatively less explored. Authors of [72] introduce **Universal Litmus Patterns (ULPs)** that can be used to differentiate between a Trojaned and clean model. These patterns are trained on hundreds of clean and Trojan samples along with a classifier that identifies the existence of a backdoor from the output of the model gathered on the ULPs. Similarly, **Meta Neural Trojan Detection (MNTD)** [154] trains a meta-classifier along with a set of finetuned queries, where the output of the victim DNN on the queries is used by the meta-classifier to detect whether the model has been Trojaned. Rather than using the model output on specific queries, Fields et al. [34] show that the distribution of the weight values in the DNN is directly indicative of whether the model has a backdoor. Specifically, it is shown empirically that the distribution of the weight values belonging to the Trojan target class is different from benign classes in a Trojaned model. Therefore, the authors proposed a method to analyze the difference between weights belonging to different classes, and use it as a basis to detect the existence of a backdoor.

3.2.1 Data Inspection. Defense methods in this domain aim to determine if the input data contains the backdoor trigger. A line of work in this domain uses the differences between the latent feature maps of benign versus Trojan samples to detect incoming Trojans. As an example, the victim DNN's training data is used to perform activation clustering for benign and Trojan samples in [21, 93]. More recently, Spectre [53] leverages the spectral signatures in the latent features to detect Trojan samples and removes them from the training data. Other works aim to locate Trojan triggers in the incoming input samples during inference. Sentinet [27] is an example work which

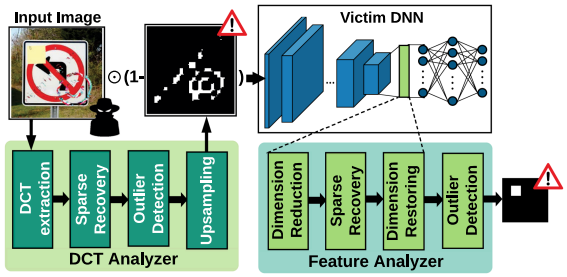


Fig. 9. High-level overview of CLEANN Trojan detection methodology [65]. The proposed defense detects both digital and physical attacks using a pair of input and latent feature analyzers.

locates the trigger region in the input via gradient computations. Similarly, Februus [31] detects the potential location for the Trojan trigger and uses pretrained **Generative Adversarial Networks (GANs)** to fill in the region with benign pixels to remove the Trojan effect. Performing the aforementioned check raises two main challenges: (1) finding an informative and accurate measure of similarity between backdoored and clean data is not trivial and (2) benign data is scarce.

Javaheripi et al. [65] propose a new defense framework, dubbed CLEANN, which combines the benefits of the aforesaid detection schemes by simultaneously inspecting incoming samples in both the latent feature space and the input domain. CLEANN is the first end-to-end accelerated framework for online detection of Neural Trojans in resource-constrained embedded applications. The proposed lightweight Trojan defense is devised based on algorithm/hardware co-design; the algorithmic insights offer a highly accurate and low-overhead Trojan detection method; the accompanying specialized hardware accelerator enables low-latency and energy-efficient defense execution on embedded hardware. Figure 9 illustrates the high-level flow of CLEANN methodology for Trojan detection. To ensure applicability to various attacks, CLEANN dictionaries model the distribution of both frequency and spatial content of benign data. This is performed via the DCT and Feature analyzers located at the input and latent features of the victim model, respectively. By aggregating the decision of the two analyzers, CLEANN significantly reduces the attack success rate for a wide range of physical [49] and digital [88] Trojan attacks. More notably, by applying sparse recovery to incoming signals in both frequency and spatial domains, CLEANN successfully removes the Trojan effect, thereby recovering the original label of Trojan samples without the need for any model fine-tuning/training. To enable high-throughput and low-energy execution of the modules shown in Figure 9, the authors further devise the first customized Trojan shields on FPGA. The library encloses customizable modules for the two main components required for Trojan detection, namely, matrix-vector multiplication and OMP for sparse recovery.

3.3 Defense against Poisoning Attacks in the Federated Setting

3.3.1 Defense Against Backdoor Attacks. We first review contemporary defense methods against FL backdoor attacks (introduced in Section 2) below. Most of the defense methods utilize anomaly detection to find models with malicious behaviors and exclude them during aggregation. As shown in Figure 10, the red clients are denoted as malicious and the black clients are denoted as clean. Anomaly detection in federated learning settings aims to detect the adversaries from uploaded model weights and exclude them during model aggregation.

Recall that FL systems are vulnerable to backdoor attacks as shown in Figure 3. To mitigate the threats of poisoning attacks, FLAME [104] presents a holistic defense framework against backdoor attacks in FL by approximating the sufficient amount of noise to be injected into the global model with the goal of backdoor elimination. More specifically, FLAME estimates the noise level

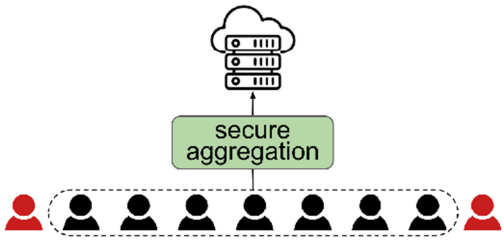


Fig. 10. A high-level depiction of anomaly detection utilized in federated learning setting [43].

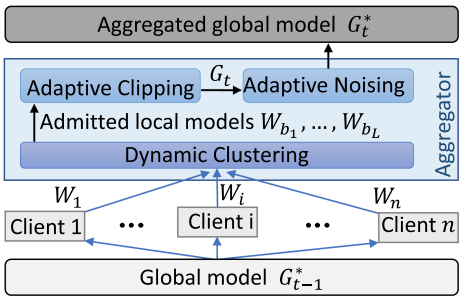


Fig. 11. High-level workflow of FLAME in round t [104]. FLAME integrates three key steps: dynamic clustering, adaptive clipping, and adaptive noising.

for backdoor removal *without* extensive empirical evaluation or access to the private training data. The effectiveness of the noising-based backdoor elimination is also theoretically justified. To avoid excessive noise injection, FLAME integrates two orthogonal modules, clustering-based model filtering and weight clipping, that are performed before model noising.

FLAME [104] can be seamlessly integrated into existing FL systems. The workflow of the defended FL system is shown in Figure 11. In each round of FL training, the local updates from the clients W_i first undergo dynamic clustering, which uses the pairwise cosine distance between local models as the metric for clustering. The majority cluster is considered benign and admitted. Then, FLAME performs adaptive weight clipping on the admitted local models where the clipping threshold is set to be the median of Euclidean distances between the global model and each local model. The weight clipped local models are then aggregated by the server and the global model G_t is updated. Finally, FLAME estimates the proper noise level and applies adaptive noising on the aggregated global model. It is worth noting that the clustering-based model filtering and adaptive clipping are critical to reduce the amount of noise required for DP-based backdoor mitigation. Otherwise, a large noise will be needed to apply on the global model, resulting in poor performance on the desired application [101, 143].

DeepSight [118] is a model filtering-based FL backdoor detection method that inspects the internal structure and the output behaviors of neural networks for identifying malicious local models. Particularly, DeepSight develops a technique that characterizes the distribution of the training data used to obtain the given model. The fine-grained difference between models is measured and models that are trained using data from similar distributions are clustered together. This allows DeepSight to identify local models that are trained for the backdoor task and exclude them from model aggregation. Furthermore, DeepSight develops two new techniques, **Division Differences (DDifs)** and **NormalizEd UPdate energies (NEUPs)**, to measure the fine-grained differences

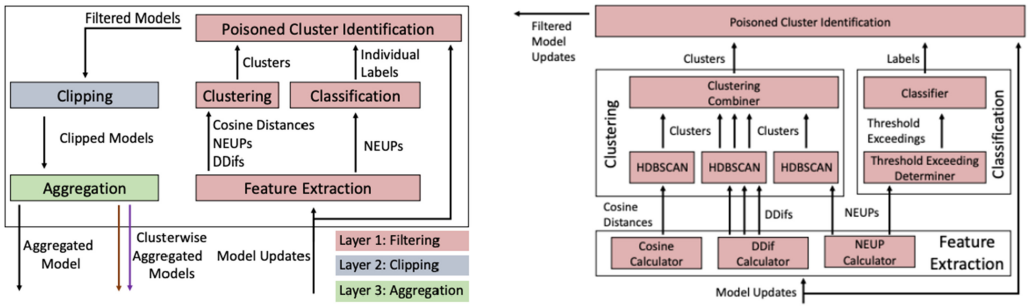


Fig. 12. The overview (a) and the filtering layer (b) of DeepSight [118].

between the internals/outputs of multiple models. In addition to model filtering, DeepSight deploys weight clipping to further mitigate the backdoor impact.

Figure 12(a) shows the high-level structure of DeepSight defense. The first defense layer of DeepSight is the filtering layer, which employs a classifier to detect and remove backdoored models with high attack impact. It is worth noting that three different techniques are integrated to extract similarity features for the classification problem. After the filtering layer, DeepSight performs clipping on the remaining local models and then aggregates them to update the global model [118]. The combination of filtering and clipping are shown to be effective in defeating the backdoor attack. Figure 12(b) shows the detailed design of the filtering layer. DeepSight develops a *voting-based model filtering* technique to combine clustering-based estimation of model similarity and a meta-classifier. The filtering layer works as follows. It first computes the features for clustering, including DDifs, NEUPs, and pairwise cosine distances. Parallel to clustering, the threshold exceeding metrics are computed by the NEUPs and fed into the classifier to label model updates as benign or poisoned. In the last step, **Poisoned Cluster Identification (PCI)** ensembles the decisions from clustering and the classifier's output labels to determine whether the local updates shall be accepted or removed.

Xie et al. present a general framework named **Certifiably Robust Federated Learning (CRFL)** that trains FL models with certifiable robustness against backdoors [149]. CRFL controls the smoothness of the global model using clipping and model parameters smoothing. CRFL provides sample-level certified robustness against backdoor attacks with limited magnitude. Figure 13 illustrates the global flow of CRFL in both the training and testing phases. During FL training, the server in CRFL aggregates the local updates, clips the norm of the aggregated model parameters, and then adds random noise to the clipped global model. The resulting model is distributed to each client for the next round of training. At the test time, the server performs *randomized parameter smoothing* on the converged global model and obtains the prediction outcome using the parameter-smoothed model.

While prior works have studied robust aggregation rules and empirically demonstrated robust FL training against backdoor attacks [5, 104, 118], they do not provide certifiable robustness. On the contrary, CRFL [149] provides the first certifiably robust FL framework and also theoretically analyzes the training dynamics of the global model using Markov Kernel. Additionally, CRFL proposes model parameter smoothing during inference to enhance the robustness. The relation between certified robustness and the hyper-parameters (poisoning ratio, number of attackers, number of FL rounds) is investigated.

There are other backdoor defense techniques in the FL setting. Wu et al. propose federated model pruning as a backdoor mitigation technique after the training phase [145]. The redundant neurons

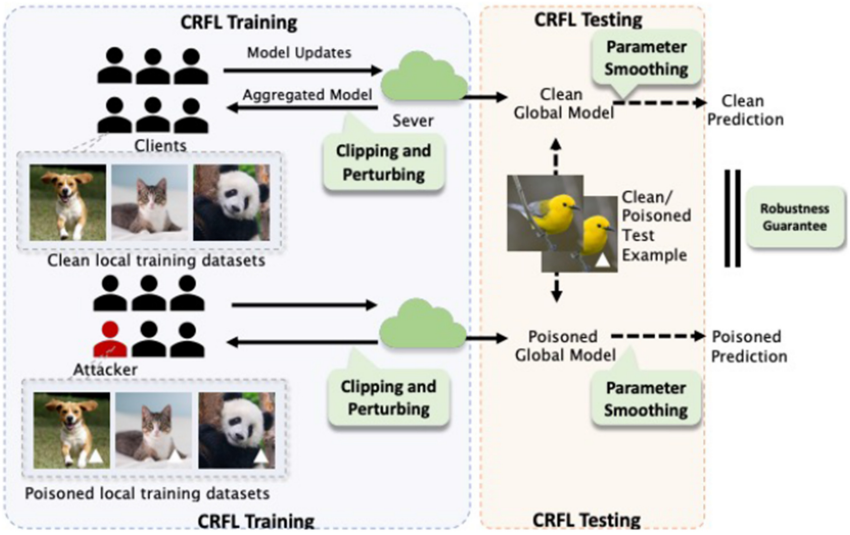


Fig. 13. Global flow of Certifiably Robust Federated Learning (CRFL) [149].

are removed and the pruned model is fine-tuned to preserve the task accuracy of the model. The paper [81] develops a spectral anomaly detection framework to identify and remove malicious local updates based on the features of model updates in the low-dimensional latent space. FL-Defender [67] presents a robust FL framework that uses worker-wise angular similarity to extract discriminative features and re-weights local updates based on their distance from the similarity centroid.

3.3.2 Defense Against Byzantine Attacks. In response to Byzantine attacks on FL, a number of robust aggregation methods have been proposed in the literature. KRUM [10] is perhaps the pioneering work in this domain which proposes to update the global model using the client update that has a minimum Euclidean distance from its neighbors. The rationale is that malicious updates will have a larger distance from their neighbors due to their outlier nature. Another variant called Multi-KRUM [30] chooses multiple updates with minimum distance to neighbors and uses their average to update the global model. Another line of work utilizes rank-based statistics such as median to construct the benign global update. Yin et al. [159] propose two coordinate-wise aggregation rules based on median and the trimmed mean. The first aggregation rule simply uses the coordinate-wise median of all client gradients to update the global model. The second defense variant computes each coordinate of the aggregate gradient over a subset of client gradients obtained after removing a portion of the top and bottom values.

Xie et al. [151] propose three variants of the robust aggregation rule, namely marginal median, mean-around-median, and geometric median. The marginal median is similar to the coordinate-wise median proposed in [159]. The mean-around-median creates the final update vector by computing the per-element mean of client updates that are close to the median in the corresponding dimension. Similarly, AKSEL [12] defines a robust interval around the coordinate-wise median using the median distance with all client updates. Updates that fall within the robust interval are then aggregated to update the global model. The geometric median is another multi-dimensional generalization of median and is defined as a vector with minimum distance with all gathered client updates. Variants of the geometric median are also adopted by other works for robust aggregation, e.g., [24, 106]. Several recent works propose to leverage the history of past gradients when

Table 3. Comparison of Defenses toward Model Poisoning Attacks

Method	Attack	Learning Setting	Defense Strategy	Defense Domain
CLEANN [65]	Backdoor	CL	Data Inspection	Hardware
Neural Cleanse [136]	Backdoor	CL	Model Inspection	Software
DeepInspect [22]	Backdoor	CL	Model Inspection	Software
ABS [87]	Backdoor	CL	Model Inspection	Software
TAD [162]	Backdoor	CL	Model Inspection	Hardware
FLAME [104]	Backdoor	FL	Anomaly Detection	Software
DeepSight [118]	Backdoor	FL	Anomaly Detection	Software
CRFL [149]	Backdoor	FL	Certifiably Robustness	Software
AKSEL [12]	Byzantine	FL	Anomaly Detection	Software

CL: central learning, FL: federated learning.

constructing the robust aggregation rule. In SafeguardSGD [3], the server keeps track of the sum of gradients sent by each client over all past training iterations. The server then uses distance from a pair of surrogate medians to identify benign clients, which will be allowed to participate in the global model update. El Mhamdi et al. [33] apply previously proposed robust aggregation rules, e.g., KRUM, on a weighted average of past gradients computed using a momentum term. Similarly, Karimireddy et al. [70] propose center-clipping client updates using a history of past gradients as the robust aggregation rule and provide theoretical guarantees of convergence.

The above mentioned works assume the server has access to the client updates in their raw format. However, in many scenarios, the local updates may be obfuscated to preserve client privacy. An example of obfuscation methods is the variant of secure FL which rely on cryptographically secure protocols to perform aggregation on client updates [9, 11, 131, 156]. In this setting, the server does not learn any of the client updates, only the final aggregate value used to update the global model. Hiding client updates from the server opens up a new attack surface for malicious clients since the server can no longer use robust aggregation rules to inspect incoming updates. In response to this threat model, several secure aggregation rules have been proposed in the literature that are Byzantine robust. BREA [126] implements the multi-KRUM robust aggregation on secretly shared client updates. SHARE [135] iteratively divides users into random clusters and uses secure aggregation to compute their average. It then applies robust aggregation rules on the cluster means and removes clusters that appear suspicious from the final global model update. EIFFeL [28] uses norm bounds during secure aggregation to check whether the client updates are benign.

3.4 Summary of Defense Strategies against Model Poisoning Attacks

In this subsection, we summarize the capacity of each defense method in Table 3. It is important to note that quantitative comparison of defense methods is not possible as they are trained and tested on different datasets with varying attack scenarios. However, we have provided a comparison of their capacities and domains, highlighting the specific areas in which they are effective. The table presents, from left to right, the attack settings that each method aims to defend against, including the attack types and learning settings, as well as the corresponding defense strategy and domain.

4 DEFENSE AGAINST ADVERSARIAL ATTACKS

In this section, we survey contemporary defenses against adversarial attacks by categorizing them into model inspection and data inspection based methods. We note that while previous literature defends adversarial attacks from theoretical side [29, 109] to training side [56, 138], our major focus in this section (and also this paper) is how software/hardware co-design can help accelerate the

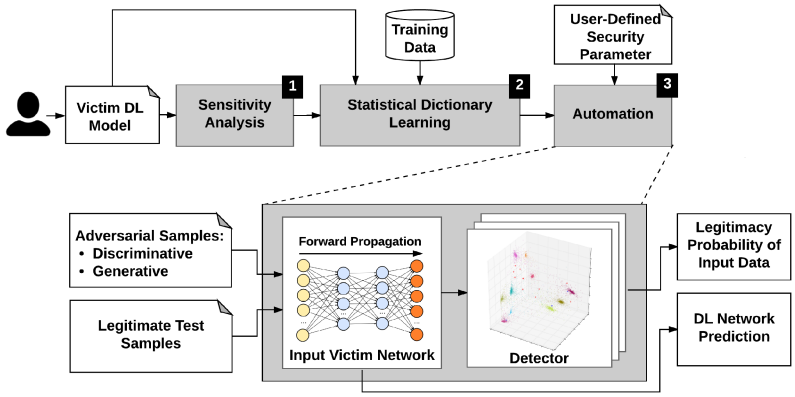


Fig. 14. Global flow of model inspection framework built in three inter-connected steps [66]: (1) analyzing layer-wise sensitivity of the victim DL model, (2) establishing modular redundancies by learning statistical properties of the model parameters and input data, and (3) automating the establishment of the proposed countermeasures based on a user-defined security parameter.

defense and detection of adversarial attacks. In what follows, we highlight customized accelerators devised for real-time detection of adversarial attacks.

4.1 Model Inspection

A general workflow of model inspection is shown in Figure 14. In the first stage, sensitivity analysis is employed to learn the statistical properties of features within a certain layer of the corresponding neural network. Then, statistical learning or more recent machine learning algorithms are used to learn the statistical properties of latent features from the training data. Finally, a user-defined security parameter automation process is applied to detect adversarial samples. In particular, the security parameter determines a trade-off between the probability of false alarms (false positives) and the probability of successful detection of adversarial samples (true positives).

DeepFense [66, 119] is one of the pioneering works in co-design of algorithm and hardware for defense against adversarial attacks. The proposed defense relies on the critical observation that benign and adversarial samples behave differently in the feature space of DNNs. Specifically, adversarial examples often target *rarely-explored* sub-spaces in a DNN's feature space, which exist due to the high dimensionality of the latent spaces and lack of sufficient labeled training data to fully characterize said space. Building upon this insight, authors design **Modular Robust Redundancies (MRRs)** that model the *explored* sub-spaces in the victim DNN, i.e., the subspaces occupied by observed benign data points. This is done by learning the probabilistic distribution of benign samples in the feature space. The learned distributions are then used to detect adversarial samples by marking them as outliers. The proposed unsupervised statistical modeling allows DeepFense to withstand unseen adversarial attacks in both white-box and black-box settings. The MRRs in DeepFense are DNNs with an identical topology to the victim model. The MRRs, however, are finetuned with a specific loss that condenses the latent features within each class, while enforcing separability between features of different classes. Figure 15 demonstrates the effect of the customized MRR loss on the latent features. By enforcing such separability, the feature space can be well-modeled as a **Gaussian mixture model (GMM)** representing benign data points in different classes. Benign samples can then be effectively identified by measuring the distance to their corresponding class center.

A key distinguishing feature of DeepFense compared to prior defenses is the specialized hardware modules devised to accelerate the MRRs and enable detection in real-time. The authors

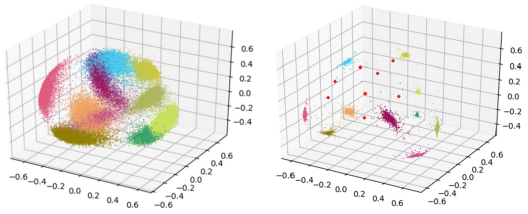


Fig. 15. Example data points in the latent feature space of a victim DNN (left figure) and its MRR (right figure) trained with specialized data realignment loss. Adversarial samples (red dot points) are easily recognizable in the latent space of the MRR [119].

implement custom compute kernels for the latent MRRs and the input sparse recovery modules on FPGAs. The computer kernels for latent MRRs comprise execution units for the underlying DNN, along with other operations such as PCA and distance calculation for outlier detection. The compute kernels for the input sparse recovery module accelerate the **Orthogonal Matching Pursuit (OMP)** algorithm [130] which reconstructs the input at a given target sparsity using pre-learned dictionaries. DeepFense is further accompanied by an automated tool that customizes the defense configuration to maximize robustness at a user-defined latency constraint while adhering to the underlying platform constraints on memory and compute resources. Specifically, the automation tool optimizes the trade-off between robustness and performance by determining the best number of MRRs along with the underlying design hyperparameters for their FPGA-based compute kernels.

Following DeepFense, several other works develop custom hardware accelerators for adversarial defense. DNNGuard [140] proposes a heterogeneous DNN accelerator which enables simultaneous execution of the victim DNN and the accompanying defense mechanism. When needed, the accelerator communicates with a host CPU to perform any non-DNN operations required by the defense. The proposed design supports acceleration for various operations in contemporary adversarial defense literature (see Table 1 in [140]). To facilitate co-location of the victim DNN and the defense on the same accelerator, the authors address several design challenges. They devise a buffering mechanism that allows the defense to efficiently access the intermediate features of the victim DNN amid execution. The authors further design a scheduler along with a custom instruction set that allows for dynamic reconfiguration of the computation units for the victim DNN and defender to ensure the defense execution is in sync with the victim prediction.

Ptolemy [40] proposes a new defense against adversarial attacks based on algorithm-hardware co-design. The proposed defense learns the unique sequence of neurons activated in the victim DNN for benign samples depending on their class. It then uses mismatches between the activated path and the final target class to identify adversarial inputs. The authors augment a DNN accelerator with new instructions and compute kernels that allow for extracting the active path of neurons. The developed hardware further supports a pretrained classifier to identify whether the extracted path is adversarial. FCDM [75] detects adversarial perturbations in the input space using **sensor pattern noise (SPN)** for the image domain. Specifically, SPN is an indicator of whether the input image has been altered after generation. The authors therefore train a defender which quantifies the probability of the sample being adversarial using a custom fingerprint extracted from the SPN. FCDM is accompanied by a custom FPGA accelerator for extracting the SPN fingerprints as well as measuring the adversarial detection confidence. The authors of [139] propose a defense that exploits hardware faults to detect adversaries. They show that it is possible to detect adversarial inputs using deliberate clock glitches injected during DNN execution, since DNN predictions for benign samples are more robust to glitches. More recently, the authors of [39] design a new

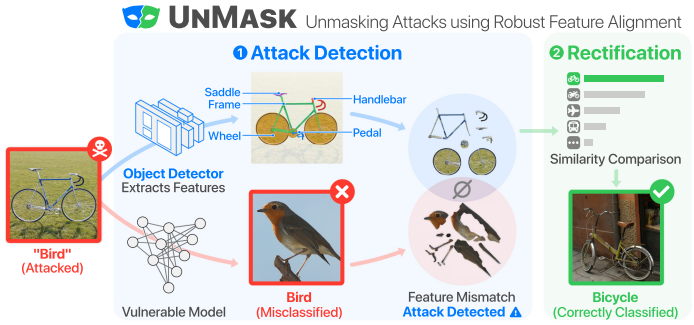


Fig. 16. Unmask [38] overview. It detects adversarial attacks (in red) through extracting *robust features* from an image ("Bicycle" at top), and comparing them to expected features of the classification ("Bird" at bottom) from the unprotected model.

DNN accelerator which supports random precision switch during DNN inference. The proposed defense relies on the inherent increased robustness of quantized DNNs compared to full-precision counterparts.

4.2 Data Inspection

In this subsection, we emphasize data inspection from two aspects, where adversarial data detection aims to detect malicious samples from input data, and semi-fragile data watermarking aims to protect data during transmission to avoid malicious transformations used for adversarial attacks.

4.2.1 Adversarial Data Detection. The adversarial data detection aims to detect adversarial data from input datasets based on the fact that under certain transformations, adversarial data and clean data have different representations. Despite that adversarial data and clean data are visually identical and can fool the classifier for wrong prediction, they can still be classifiers by training extra classifiers or using different transformation algorithms.

Gong et al. [45] is one of the pioneering works proposed that a simple binary classifier separating the adversarial apart from the clean data with accuracy over 99%. They also empirically show that the binary classifier is robust to a secondround adversarial attack. A followup work by Grosse et al. [48] gives a more statistical explanation that adversarial data are not drawn from the same distribution as the clean data, and can thus be detected using statistical tests.

Unmask [38] is a framework to detect adversarial attacks by quantifying the similarity between the image's extracted features with the expected features of its predicted class. As shown in Figure 16, a bicycle image was attacked and fools an unprotected model to misclassify it as a bird. For a real "bird" image, it is expected to possess features such as *beak*, *wing*, and *tail*, which adversarial images do not have. Therefore, Unmask trains its own object detectors and extracts correct like features: *wheel*, *frame*, and *pedals*. Unmask then quantifies the similarity between the *extracted* features (of a bike) with the *expected* features (of a bird). This comparison gives Unmask dual ability to both detect adversarial perturbations by selecting a similarity threshold for which should classify an image as adversarial, and to defend the model by predicting a corrected class that best matches the extracted features.

WaveGuard [58] is the first ASR defense to successfully guard against adaptive adversaries who have complete knowledge of the defense framework. WaveGuard proposes novel audio transformation-based defenses and achieves higher detection accuracy in comparison to prior work [112, 157] in adversarial audio detection. The WaveGuard framework transforms the given audio input x using an input transformation function g and analyzes the ASR transcriptions for

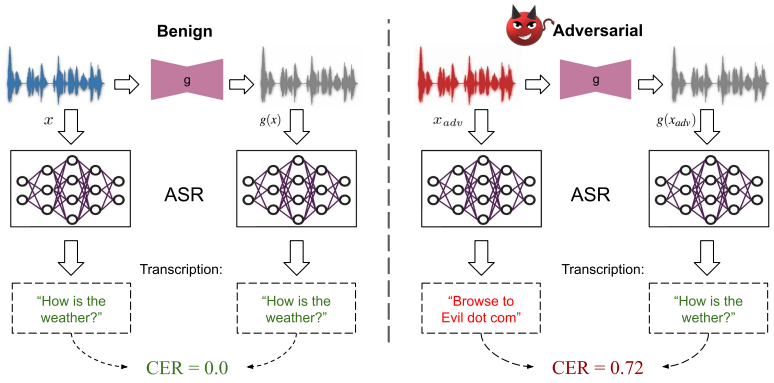


Fig. 17. WaveGuard [58] Defense Framework: We first process the input audio x using an audio transformation function g to obtain $g(x)$. Then the ASR transcriptions or x and $g(x)$ are compared. An input is classified as *adversarial* if the difference between the transcriptions of x and $g(x)$ exceeds a particular threshold.

the input x and $g(x)$. The underlying idea for this defense is that model predictions for adversarial examples are unstable while those for benign examples are robust to small changes in the input. Therefore, the framework labels an input as adversarial if there is a significant difference between the transcriptions of x and $g(x)$. An overview of the defense is depicted in Figure 17. WaveGuard defense framework is computationally inexpensive and robust against an adaptive adversary who has complete knowledge of the defense. The authors find that certain input transformation functions that reduce audio to a perceptually informed representation can lead to a strong defense that is robust against an adaptive adversary even in a complete white-box setting. Particularly, the authors report that **Linear Predictive Coding (LPC)** and Mel spectrogram extraction and inversion are more robust to adaptive attacks as compared to other transformation functions studied in this work.

Following WaveGuard, several works [85, 96] have leveraged audio transformation functions to defend against audio adversarial attacks. Mehlman et al. [96] build upon WaveGuard by similarly transforming the audio signal into the mel representation and then introduce a novel noise flooding technique called **Mel Domain Noise Flooding (MDNF)** which injects white Gaussian noise into the mel representation of audio prior to re-synthesis of the audio. MDNF is able to defend against strong white-box adversarial attacks and achieves higher robustness compared to randomized smoothing baselines.

Data inspection is also widely applied in other structural data formats like texts. Adversarial attacks in natural language domain refers to adversaries that add small perturbations to the input text, which are often imperceptible to humans but can significantly change the model's output. The perturbations can be either in the token level or word embedding level depending on the target NLP tasks. To defend against the token level perturbations, one of the most common ways is to check misspellings and typos in the input data as proposed by Pruthi et al. [108]. In the follow up, Mozes et al. [98] proposed that perturbations can be detected through frequency differences between replaced words and their corresponding substitutions. For embedding level detection, TREATED [163] proposed a universal adversarial detection method that can defend against attacks of various perturbation levels without making any assumptions. It utilizes several reference models to make different predictions about clean and adversarial examples. Then, it blocks the inputs if they are adversarial.

In followup work, Wang et al. [141] proposed to defense synonym substitution based textual adversarial attacks by randomly substituting a word with its synonyms. In their proposed RS&V

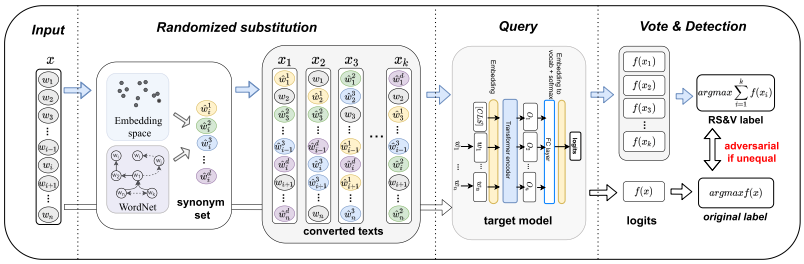


Fig. 18. The overall framework of RS&V [141].

method, given an input x, RS&V first generate the synonym set $\mathcal{S}(w_i)$ for each word $w_i \in x$ using the GloVe vector as well as WordNet, and randomly substitute a word $w_i \in x$ with its arbitrary synonym $\hat{w}_i^j \in \mathcal{S}(w_i)$ to generate multiple texts. Then, RS&V feed these generated texts and accumulate the logits to get the voted label. Finally, RS&V regard x as an adversarial example if the RS&V label is not consistent with the prediction label for x.

4.2.2 Semi-Fragile Data Watermarking. To counteract the threat of adversarial attacks in video processing and detection systems, several works [26, 91, 147, 148] have proposed defenses that aim to classify the video frame as adversarial or benign. The authors of [26] adaptively compress videos with JPEG compression such that any embedded adversarial perturbations are suppressed without losing video quality. While a few of these defenses [147] are effective against adaptive attackers, most of these defenses are vulnerable to attack algorithms that are partially or completely aware of the defense mechanism [6, 7] in an adaptive attack setting.

As a solution to proactively defending against adversarial attacks during information transmission like in video processing systems, semi-fragile data watermarking is proposed to protect input data. It is robust to adversarial attacks because, even if an attacker gains white-box access to the decoder model, they may not have access to the message and encryption key required to embed a trusted watermark into the image. That is, the target for the adversarial attack is unknown since this is not a classification system. There are high dimensional possible outputs of the decoder network and the adversary does not know which ones are valid outcomes.

FastStamp [60] propose a real-time semi-fragile image watermarking technique to prove media authenticity and establish media provenance. The overview of FastStamp watermarking pipeline is shown in Figure 19. The proposed frameworks embed a verifiable digital signature of 128 bits in the image pixels such that the signature is preserved when benign image transformations such as compression, color and contrast adjustments, and saturation adjustments are applied to an image. However, if the image is manipulated (e.g., with adversarial perturbations or malicious Deepfake tampering), the signature breaks and the image can no longer be verified. For each frame/image, a signature is generated and encrypted to generate a watermark, which is up-sampled to the same size as the image. The image and watermark are input to FastStampâĂŹs encoder to obtain a marked image. To extract a watermark, the watermarked image is input to the decoder, which outputs an encrypted watermark. Once this watermark is decrypted, it is checked against a database of trusted signatures to assert validity. FastStamp also develops the first FPGA-based accelerator framework to further improve the watermarking model throughput and power consumption by leveraging data parallelism and customized computation paths. FastStamp develops reconfigurable sub-modules which can accelerate convolutional downsampling and upsampling networks on hardware. Particularly, this framework allows watermark embedding directly at the hardware source, which not only secures the image capture and transmission pipeline but also reduces latency in embedding the watermark.

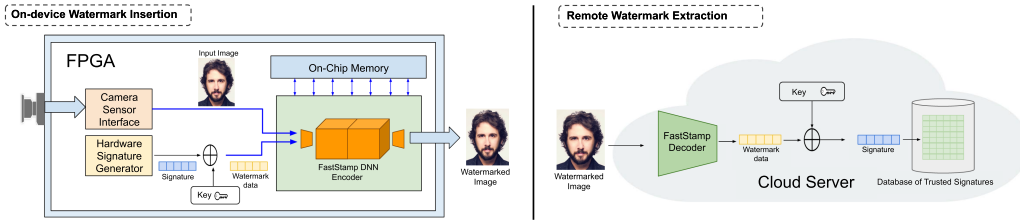


Fig. 19. Schematic diagram of FastStamp watermarking pipeline [60]. The pipeline is divided into two steps: watermark insertion using a DNN encoder on FPGA (left) and watermark extraction using a DNN decoder on a cloud server (right).

Table 4. Comparison of Defenses toward Adversarial Attacks

Method	Modality	Defense Strategy	Defense Domain	Model Access
DeepFense [66, 119]	Image	Model Inspection	Hardware	White box
DNNGuard [140]	Image	Model Inspection	Hardware	White box
Ptolemy [40]	Image	Model Inspection	Hardware	White box
FCDM [75]	Image	Model Inspection	Hardware	White box
Unmask [38]	Image	Data Inspection	Software	Black box
WaveGuard [58]	Audio	Data Inspection	Software	White box
Mehlman et al. [96]	Audio	Data Inspection	Software	White box
Mozes et al. [98]	Text	Data Inspection	Software	Black box
TREATED [163]	Text	Data Inspection	Software	Black box
RS&V [141]	Text	Data Inspection	Software	Black box
FastStamp [60]	Video	Data Inspection	Hardware	White box

4.3 Summary of Defense Strategies against Adversarial Attacks

In this subsection, we present a summary of the capacity of each defense method in Table 4. As we are comparing adversarial attacks across different modalities, it is not feasible to directly compare the performance of each defense method. Instead, our objective is to provide a characterization of the different defense methods. From left to right, we present the the adversarial attack modality that each method aims to defend against, along with its corresponding defense strategy, domain, and level of target model access.

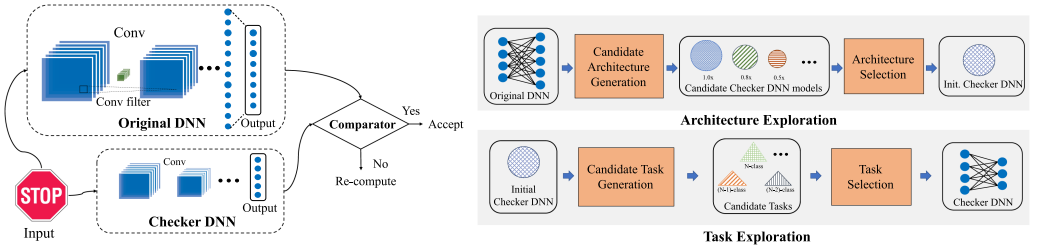
5 DEFENSE AGAINST FAULT-INJECTION ATTACKS

Defenses against bit-level faults can be categorized into three separate classes:

(1) Machine-learning-based anomaly detection defenses that train a classifier to detect whether any faults have occurred during the execution of the victim DNN [84, 86].
(2) Model inspection based defenses that use error-detection codes to identify any variations in the bits values, e.g., RADAR [78] uses checksums to detect bit-level faults in DNN weights.
(3) Defenses that alter the DNN training process to increase robustness to bit-level faults, e.g., piece-wise clustering [54], binarization of model weights and activations [116], and local weight reconstruction [79].

5.1 Machine-learning-based Anomaly Detection

Work in this subsection proposes to train a separate machine learning model to determine whether any bit-flips have occurred in the victim. Li et al. [84] proposed DeepDyve, which employs a



(a) Architecture overview (b) Design stages

Fig. 20. Overview of DeepDyve [84] which uses machine learning based anomaly detection to find models that have been attacked.

simpler and smaller pre-trained neural networks to verify if the original DNN has been fault injection attacked, as shown in Figure 20. It trains a smaller DNN, dubbed the checker model, which is used to check whether the output of the victim is valid. The checker DNN is trained using the original victim model using knowledge distillation and can therefore mimic the benign behavior of the teacher (victim) model on the target task. Liu et al. [86] train a smaller DNN using a specific loss designed to detect the occurrence of malicious bit-flips, i.e., those that result in significant accuracy degradation. Specifically, the proposed method creates a dataset of randomly induced bit-flips falling into benign or malicious categories. This dataset is used to train a one-layer DNN which maps the input bit-flips to a binary code. The binary code is later used to classify whether the bit-flips are malicious or benign using a hamming distance metric.

5.2 Model Inspection

Due to the high cost of fault-injection attacks, attackers are incentivized to target vulnerable layers to maximize the accuracy reduction with the minimum number of bit alterations. By identifying and curating the hash generation to the most vulnerable layers, defenses [62, 78] in this category achieve high detection rate while minimizing defense overhead.

Hashtag [64] is an example defense relying on error-detection codes. The authors provide a holistic framework for real-time detection of bit-level faults occurring during DNN inference. Hashtag leverages a lightweight detection methodology that summarizes all parameters in the reference DNN in the form of several hash signatures. New hash signatures are extracted during inference and compared with the pre-computed ground-truth hashes to verify the DNN's integrity on the fly. Notably, Hashtag, for the first time, provides guaranteed lower bounds on the detection rate using formal statistical analysis of hash collision in addition to delivering a 0% false positive rate. Hashtag balances the inherent trade-off between detection accuracy and defense overhead by minimizing the number of DNN layers used for hash extraction. A high-level overview of the steps performed in HASHTAG is illustrated in Figure 21. Followup work [62] devises a custom FPGA core for Hashtag that enables simultaneous hash generation and verification with a co-processor hosting the victim DNN.

5.3 DNN Re-Training

He et al. [54] empirically demonstrates that binarizing weights during DNN training increases the robustness of the models to bit-flip attacks. In this context, the model is trained using a specific routine such that the absolute value of the weights in each layer is constant and the sign is the only differentiating factor between weight elements. As a result of this binarization, the number of bit changes required to reduce the accuracy of the victim DNN is increased significantly compared to vanilla training. The authors observe that state-of-the-art adaptive attacks often target

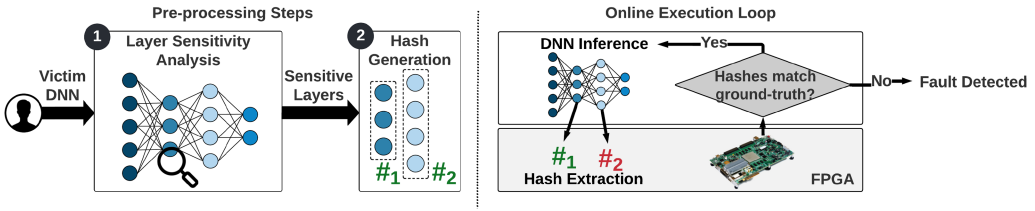


Fig. 21. Global flow of HASHTAG framework for detecting bit-level faults in DNN workloads [62]. Sensitive DNN layers are identified in a pre-processing step and used for hash signature generation. During online execution, new hashes are compared with the pre-computed signatures to identify bit-level mismatches.

Table 5. Comparison of Defenses toward Fault Injection Attacks

Method	Defense Strategy	Defense Time	Defense Domain
DeepDyve [84]	Anomaly Detection	Inference	Software
RADAR [78]	Error Detection Code	Inference	Hardware
HASHTAG [64]	Error Detection Code	Inference	Hardware
RA-BNN [116]	DNN Re-Training	Training	Software
Li et al. [79]	DNN Re-Training	Training	Software

weights close to zero to cause drastic changes in the value after altering one bit. Since binarization enforces weights to strictly two values during training, it inherently simulates the effect of bit-flip noise injection and thus prepares the model for such attacks. The authors further propose piece-wise clustering as an alternative to binarization, where the training loss encourages weight concentration around two cluster values, one for positive weights and another for negative weights. In a follow-up work, Rakin et al. [116] show that complete binarization of both weights and activations can also increase robustness to bit-flip attacks. Since such binarization reduces the accuracy of the model compared to full-precision counterparts, the authors propose a growing mechanism that automatically adjusts the number of channels in each layer to increase the accuracy of the obtained binary neural network. To increase robustness against bit-flip attacks, Li et al. [79] propose a reconstruction scheme during inference which aims to diffuse the weight change caused by bit-flips in a group of weights.

5.4 Summary of Defense Strategies against Fault Injection Attacks

In this subsection, we summarize the capacity of each defense method in Table 5. Given that different defense methods protect models at various stages, including training and inference with auxiliary models or datasets, it is not feasible to compare their performance directly. Therefore, the table presents the defense capacities, including defense strategy, time, and domain, from left to right

6 FUTURE DIRECTIONS

Robust deep learning has been rapidly advancing in recent years, with applications in both central and federated learning settings. Previous literature reviews have depicted the effectiveness of utilizing DL/software/hardware co-design approaches to enhance the security of the deep learning lifecycle. In this section, we focus on areas that show promise for future research.

6.1 Emerging Application

With the recent success of **large language models (LLMs)** like chatGPT and GPT-4, deep learning models have demonstrated the ability to understand and generate human-like responses. Despite

the development of many NLP-based attacks and defenses in recent years [98, 161, 163], the emergence of new attacks from foundation models highlights the need for effective software/hardware co-design to defend against these threats. We outline a few potential security risks that can arise from pre-training and fine-tuning large language models at both the software and hardware levels

Training foundation models require significant computational resources, which has led to the introduction of collaborative training methods such as federated learning to reduce the training burden. However, malicious participants may engage in Byzantine or backdoor attacks to poison the aggregated models, resulting in negative social impacts or bad model performance upon the model's release. Therefore, exploring methods to efficiently mitigate training time attacks and ensure the security of these collaborative training methods is crucial.

Researchers from different fields fine-tune foundation models with specialized corpora to enable LLMs to perform better on domain-specific tasks. However, this process introduces new security risks such as model poisoning attacks, data leakage attacks during training, and adversarial attacks during inference. For example, backdoors in the specialized corpora may introduce trojans for conducting model poisoning attacks, and biases may make the fine-tuned model more sensitive to adversarial attacks. Developing effective software-hardware co-design defense methods against these attacks is crucial to ensure that developers can detect malicious behaviors in real-time and the fine-tuned LLMs can positively impact specialized areas.

6.2 Emerging Hardware

With new hardware such as TPUs and heterogeneous platforms equipped with various GPU types, accelerating deep machine learning model training with large-scale data has become possible. However, using such hardware also poses potential risks, such as fault injection attacks on the trained models or model poisoning attacks targeting specific hardware types. These attacks at the hardware level are hard to examine, especially as the number scales. Therefore, ensuring training safety at the hardware level is crucial to safeguard deep learning models. We outline a few potential security risks arising from these training strategies.

New hardware may introduce potential security risks like unknown leakage and viruses from the novel architecture designs. As these new hardware architectures differ from established ones, fewer robust security measures may be in place, and they may not have been as extensively tested for vulnerabilities. These platforms could be vulnerable to attacks like side-channel attacks or privacy inference attacks, which leak sensitive data or algorithms. Therefore, it is important to take steps to mitigate these risks, such as implementing strong access controls, using encryption and other data protection measures.

Large-scale machine learning training requires substantial computational resources. Collaborative learning techniques, such as training models under heterogeneous platforms, have been introduced to relieve this burden. However, since different platforms may have different levels of security and vulnerability control, adversaries can exploit these differences and launch attacks like model poisoning and adversarial attacks. This can compromise the accuracy of the trained model and potentially damage the system's security. As the number of machines participating in collaborative learning increases, it becomes increasingly important to explore software-hardware co-design algorithms that help accelerate the model inspection process and effectively exclude malicious ones from being aggregated.

6.3 Hardware-software Co-design in NLP Applications

Recent advances in novel model architectures like transformers and LLMs introduced new security risks. The increased training and inference overhead from billion-scale model parameters also leads to higher computational costs, making hardware software codesign increasingly important

for enabling real-time detection of potentially malicious attacks. By introducing the hardware-software co-design, it allows machine learning systems to quickly detect and respond to security threats in time, while also incorporating security features directly into the hardware design to prevent attacks at the hardware level. Additionally, customized hardware and software components can be optimized for the specific application, reducing the risk of vulnerabilities being exploited. Therefore, leveraging hardware-software co-design helps developers to be able to create more secure and efficient NLP systems capable of detecting potential security risks in real-time.

7 CONCLUSION

Deep learning is an extremely dynamic field with new models and various applications emerging regularly. Ensuring robustness of deep learning is of critical importance for reliable model deployment. In this paper, we present a systematic study of recent defense techniques against the existing attack surface of centralized and federated deep learning. We highlight different aspects of the defenses, from algorithm-level enhancements to hardware-level optimizations and, where applicable, protocol design. The surveyed defenses show great success in thwarting attacks in existing controlled settings. As a future research direction, we believe that the researchers and practitioners shall investigate the holistic integration of defense techniques against different types of attacks. This requires an understanding of synergy between multiple defense methods and the coordination of their respective design components. In addition, many applications require measures for user privacy, which can hinder existing defense strategies. Therefore it is important to develop defenses that can be integrated with privacy-preserving computation setups, e.g., on encrypted or noisy data.

REFERENCES

[1] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. 2018. Sanity checks for saliency maps. *Advances in Neural Information Processing Systems* 31 (2018).

[2] Naveed Akhtar and Ajmal Mian. 2018. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access* 6 (2018), 14410–14430.

[3] Zeyuan Allen-Zhu, Faeze Ebrahimianghazani, Jerry Li, and Dan Alistarh. 2020. Byzantine-resilient non-convex stochastic gradient descent. In *International Conference on Learning Representations*.

[4] Moustafa Alzantot, Bharathan Balaji, and Mani B. Srivastava. 2018. Did you hear that? Adversarial examples against automatic speech recognition. *CoRR* abs/1801.00554 (2018). arXiv:1801.00554 http://arxiv.org/abs/1801.00554

[5] Sebastien Andreina, Giorgia Azzurra Marson, Helen Möllering, and Ghassan Karame. 2021. BaFFLe: Backdoor detection via feedback-based federated learning. In *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS'21)*. IEEE, 852–863.

[6] Anish Athalye, Nicholas Carlini, and David Wagner. 2018. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*.

[7] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. 2018. Synthesizing robust adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning*.

[8] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. 2020. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2938–2948.

[9] James Henry Bell, Kallista A. Bonawitz, Adrià Gascón, Tancrède Lepoint, and Mariana Raykova. 2020. Secure single-server aggregation with (poly) logarithmic overhead. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 1253–1269.

[10] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. 2017. Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in Neural Information Processing Systems* 30 (2017).

[11] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. 2017. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 1175–1191.

[12] Amine Boussetta, El-Mahdi El-Mhamdi, Rachid Guerraoui, Alexandre Maurer, and Sébastien Rouault. 2021. AKSEL: Fast Byzantine SGD. In *24th International Conference on Principles of Distributed Systems (OPODIS'20)*.

[13] Jakub Breier, Xiaolu Hou, Dirmanto Jap, Lei Ma, Shivam Bhasin, and Yang Liu. 2018. Practical fault attack on deep neural networks. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. 2204–2206.

[14] Samuel Cahyawijaya. 2021. Greenformers: Improving computation and memory efficiency in transformer models via low-rank approximation. *arXiv preprint arXiv:2108.10808* (2021).

[15] Han Cai, Ji Lin, Yujun Lin, Zhijian Liu, Haotian Tang, Hanrui Wang, Ligeng Zhu, and Song Han. 2022. Enable deep learning on mobile devices: Methods, systems, and applications. *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 27, 3 (2022), 1–50.

[16] Nicholas Carlini, Pratyush Mishra, Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields, David Wagner, and Wenchao Zhou. 2016. Hidden voice commands. In *25th USENIX Security Symposium (USENIX Security'16)*. USENIX Association, Austin, TX.

[17] Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 39–57.

[18] Nicholas Carlini and David Wagner. 2018. Audio adversarial examples: Targeted attacks on speech-to-text. In *2018 IEEE Security and Privacy Workshops (SPW'18)*. IEEE, 1–7.

[19] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. 2018. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069* (2018).

[20] Jung-Woo Chang, Mojan Javaheripi, Seira Hidano, and Farinaz Koushanfar. 2023. RoVISQ: Reduction of video service quality via adversarial attacks on deep learning-based video compression. In *Network and Distributed System Security Symposium (NDSS'23)*.

[21] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. 2018. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv preprint arXiv:1811.03728* (2018).

[22] Huili Chen, Cheng Fu, Jishen Zhao, and Farinaz Koushanfar. 2019. DeepInspect: A black-box Trojan detection and mitigation framework for deep neural networks. In *IJCAI*, Vol. 2. 8.

[23] Huili Chen, Cheng Fu, Jishen Zhao, and Farinaz Koushanfar. 2021. ProFlip: Targeted Trojan attack with progressive bit flips. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 7718–7727.

[24] Yudong Chen, Lili Su, and Jiaming Xu. 2017. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 1, 2 (2017), 1–25.

[25] Yuxuan Chen, Xuejing Yuan, Jiangshan Zhang, Yue Zhao, Shengzhi Zhang, Kai Chen, and XiaoFeng Wang. 2020. Devil's whisper: A general approach for physical adversarial attacks against commercial black-box speech recognition devices. In *29th USENIX Security Symposium (USENIX Security'20)*. USENIX Association, Boston, MA.

[26] Yupeng Cheng, Xingxing Wei, Huazhu Fu, Shang-Wei Lin, and Weisi Lin. 2021. Defense for adversarial videos by self-adaptive JPEG compression and optical texture. In *Proceedings of the 2nd ACM International Conference on Multimedia in Asia* (Virtual Event, Singapore) *(MMAsia'20)*. Association for Computing Machinery, New York, NY, USA, Article 55, 7 pages. https://doi.org/10.1145/3444685.3446308

[27] Edward Chou, Florian Tramer, and Giancarlo Pellegrino. 2020. SentiNet: Detecting localized universal attacks against deep learning systems. In *2020 IEEE Security and Privacy Workshops (SPW'20)*. IEEE, 48–54.

[28] Amrita Roy Chowdhury, Chuan Guo, Somesh Jha, and Laurens van der Maaten. 2021. EIFFeL: Ensuring integrity for federated learning. *arXiv preprint arXiv:2112.12727* (2021).

[29] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. 2019. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*. PMLR, 1310–1320.

[30] Georgios Damaskinos, El Mahdi El Mhamdi, Rachid Guerraoui, Arsany Hany Abdelmessih Guirguis, and Sébastien Louis Alexandre Rouault. 2019. AGGREGATHOR: Byzantine machine learning via robust gradient aggregation. In *The Conference on Systems and Machine Learning (SysML'19)*.

[31] Bao Gia Doan, Ehsan Abbasnejad, and Damith C. Ranasinghe. 2020. Februus: Input purification defense against Trojan attacks on deep neural network systems. In *Annual Computer Security Applications Conference*. 897–912.

[32] Yinpeng Dong, Xiao Yang, Zhijie Deng, Tianyu Pang, Zihao Xiao, Hang Su, and Jun Zhu. 2021. Black-box detection of backdoor attacks with limited information and data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 16482–16491.

[33] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Louis Alexandre Rouault. 2021. Distributed momentum for Byzantine-resilient stochastic gradient descent. In *9th International Conference on Learning Representations (ICLR'21)*.

[34] Greg Fields, Mohammad Samragh, Mojan Javaheripi, Farinaz Koushanfar, and Tara Javidi. 2021. Trojan signatures in DNN weights. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 12–20.

[35] Luciano Floridi and Massimo Chiriatti. 2020. GPT-3: Its nature, scope, limits, and consequences. *Minds and Machines* 30, 4 (2020), 681–694.

[36] Bryse Flowers, R. Michael Buehrer, and William C. Headley. 2019. Evaluating adversarial evasion attacks in the context of wireless communications. *IEEE Transactions on Information Forensics and Security* 15 (2019), 1102–1113.
[37] Ruth C. Fong and Andrea Vedaldi. 2017. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision*. 3429–3437.
[38] Scott Freitas, Shang-Tse Chen, Zijie J. Wang, and Duen Horng Chau. 2020. Unmask: Adversarial detection and defense through robust feature alignment. In *2020 IEEE International Conference on Big Data (Big Data'20)*. IEEE, 1081–1088.
[39] Yonggan Fu, Yang Zhao, Qixuan Yu, Chaojian Li, and Yingyan Lin. 2021. 2-in-1 accelerator: Enabling random precision switch for winning both adversarial robustness and efficiency. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*. 225–237.
[40] Yiming Gan, Yuxian Qiu, Jingwen Leng, Minyi Guo, and Yuhao Zhu. 2020. Ptolemy: Architecture support for robust deep learning. In *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'20)*. IEEE, 241–255.
[41] Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW'18)*. IEEE, 50–56.
[42] Yansong Gao, Bao Gia Doan, Zhi Zhang, Siqi Ma, Jiliang Zhang, Anmin Fu, Surya Nepal, and Hyoungshick Kim. 2020. Backdoor attacks and countermeasures on deep learning: A comprehensive review. *arXiv preprint arXiv:2007.10760* (2020).
[43] Zahra Ghodsi, Mojan Javaheripi, Nojan Sheybani, Xinqiao Zhang, Ke Huang, and Farinaz Koushanfar. 2022. zPROBE: Zero peek robustness checks for federated learning. *arXiv preprint arXiv:2206.12100* (2022).
[44] Loris Giulivi, Malhar Jere, Loris Rossi, Farinaz Koushanfar, Gabriela Ciocarlie, Briland Hitaj, and Giacomo Boracchi. 2023. Adversarial scratches: Deployable attacks to CNN classifiers. *Pattern Recognition* 133 (2023), 108985. https://doi.org/10.1016/j.patcog.2022.108985
[45] Zhitao Gong, Wenlu Wang, and Wei-Shinn Ku. 2017. Adversarial and clean data are not twins. *arXiv preprint arXiv:1704.04960* (2017).
[46] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).
[47] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. *Stat.* (2015).
[48] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel. 2017. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280* (2017).
[49] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. 2017. BadNets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733* (2017).
[50] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2019. BadNets: Evaluating backdooring attacks on deep neural networks. *IEEE Access* 7 (2019), 47230–47244.
[51] Wei Guo, Benedetta Tondi, and Mauro Barni. 2022. An overview of backdoor attacks against deep neural networks and possible defences. *IEEE Open Journal of Signal Processing* (2022).
[52] Wenbo Guo, Lun Wang, Xinyu Xing, Min Du, and Dawn Song. 2019. Tabor: A highly accurate approach to inspecting and restoring Trojan backdoors in AI systems. *arXiv preprint arXiv:1908.01763* (2019).
[53] Jonathan Hayase and Weihao Kong. 2020. Spectre: Defending against backdoor attacks using robust covariance estimation. In *International Conference on Machine Learning*.
[54] Zhezhi He, Adnan Siraj Rakin, Jingtao Li, Chaitali Chakrabarti, and Deliang Fan. 2020. Defending and harnessing the bit-flip based adversarial weight attack. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14095–14103.
[55] Sanghyun Hong, Pietro Frigo, Yiğitcan Kaya, Cristiano Giuffrida, and Tudor DumitraÈŹ. 2019. Terminal brain damage: Exposing the graceless degradation in deep neural networks under hardware fault attacks. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*. 497–514.
[56] Hanxun Huang, Yisen Wang, Sarah Erfani, Quanquan Gu, James Bailey, and Xingjun Ma. 2021. Exploring architectural ingredients of adversarially robust deep neural networks. *Advances in Neural Information Processing Systems* 34 (2021), 5545–5559.
[57] Shehzeen Hussain, Paarth Neekhara, Brian Dolhansky, Joanna Bitton, Cristian Canton Ferrer, Julian McAuley, and Farinaz Koushanfar. 2022. Exposing vulnerabilities of deepfake detection systems with robust attacks. *Digital Threats* 3, 3, Article 30 (Sep. 2022), 23 pages. https://doi.org/10.1145/3464307
[58] Shehzeen Hussain, Paarth Neekhara, Shlomo Dubnov, Julian McAuley, and Farinaz Koushanfar. 2021. {WaveGuard}: Understanding and mitigating audio adversarial examples. In *30th USENIX Security Symposium (USENIX Security'21)*. 2273–2290.
[59] Shehzeen Hussain, Paarth Neekhara, Malhar Jere, Farinaz Koushanfar, and Julian McAuley. 2021. Adversarial deepfakes: Evaluating vulnerability of deepfake detectors to adversarial examples. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 3348–3357.

[60] Shehzeen Hussain, Nojan Sheybani, Paarth Neekhara, Xinqiao Zhang, Javier Duarte, and Farinaz Koushanfar. 2022. FastStamp: Accelerating neural steganography and digital watermarking of images on FPGAs. In *2022 IEEE/ACM International Conference on Computer Aided Design (ICCAD'22)*. IEEE.

[61] Dan Iter, Jade Huang, and Mike Jermann. 2017. Generating Adversarial Examples for Speech Recognition. *Technical Report*.

[62] Mojan Javaheripi, Jung-Woo Chang, and Farinaz Koushanfar. 2022. AccHashtag: Accelerated hashing for detecting fault-injection attacks on embedded neural networks. *ACM Journal on Emerging Technologies in Computing Systems (JETC)* (2022).

[63] Mojan Javaheripi, Gustavo H. de Rosa, Subhabrata Mukherjee, Shital Shah, Tomasz L. Religa, Caio C. T. Mendes, Sebastien Bubeck, Farinaz Koushanfar, and Debadeepta Dey. 2022. LiteTransformerSearch: Training-free on-device search for efficient autoregressive language models. *Advances in Neural Information Processing Systems* (2022).

[64] Mojan Javaheripi and Farinaz Koushanfar. 2021. HASHTAG: Hash signatures for online detection of fault-injection attacks on deep neural networks. In *2021 IEEE/ACM International Conference on Computer Aided Design (ICCAD'21)*. IEEE, 1–9.

[65] Mojan Javaheripi, Mohammad Samragh, Gregory Fields, Tara Javidi, and Farinaz Koushanfar. 2020. CleaNN: Accelerated Trojan shield for embedded neural networks. In *2020 IEEE/ACM International Conference on Computer Aided Design (ICCAD'20)*. IEEE, 1–9.

[66] Mojan Javaheripi, Mohammad Samragh, Bita Darvish Rouhani, Tara Javidi, and Farinaz Koushanfar. 2020. CuRTAIL: Characterizing and thwarting adversarial deep learning. *IEEE Transactions on Dependable and Secure Computing* 18, 2 (2020), 736–752.

[67] Najeeb Jebreel and Josep Domingo-Ferrer. 2022. FL-Defender: Combating targeted attacks in federated learning. *arXiv preprint arXiv:2207.00872* (2022).

[68] Malhar S. Jere, Tyler Farnan, and Farinaz Koushanfar. 2020. A taxonomy of attacks on federated learning. *IEEE Security & Privacy* 19, 2 (2020), 20–28.

[69] Saurabh Jha, Subho S. Banerjee, James Cyriac, Zbigniew T. Kalbarczyk, and Ravishankar K. Iyer. 2018. AVFI: Fault injection for autonomous vehicles. In *2018 48th Annual IEEE/IFIPInternational Conference on Dependable Systems and Networks Workshops (DSN-W'18)*. IEEE, 55–56.

[70] Sai Praneeth Karimireddy, Lie He, and Martin Jaggi. 2021. Learning from history for Byzantine robust optimization. In *International Conference on Machine Learning*. PMLR, 5311–5319.

[71] Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu. 2014. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. *ACM SIGARCH Computer Architecture News* 42, 3 (2014), 361–372.

[72] Soheil Kolouri, Aniruddha Saha, Hamed Pirsiavash, and Heiko Hoffmann. 2020. Universal litmus patterns: Revealing backdoor attacks in CNNs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 301–310.

[73] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. *Commun. ACM* 60, 6 (2017), 84–90.

[74] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. 2016. Adversarial machine learning at scale. *CoRR* (2016). arXiv:1611.01236

[75] Yazhu Lan, Kent W. Nixon, Qingli Guo, Guohe Zhang, Yuanchao Xu, Hai Li, and Yiran Chen. 2020. FCDM: A methodology based on sensor pattern noise fingerprinting for fast confidence detection to adversarial attacks. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39, 12 (2020), 4791–4804.

[76] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015).

[77] Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. TextBugger: Generating adversarial text against real-world applications. *arXiv preprint arXiv:1812.05271* (2018).

[78] Jingtao Li, Adnan Siraj Rakin, Zhezhi He, Deliang Fan, and Chaitali Chakrabarti. 2021. RADAR: Run-time adversarial weight attack detection and accuracy recovery. *arXiv preprint arXiv:2101.08254* (2021).

[79] Jingtao Li, Adnan Siraj Rakin, Yan Xiong, Liangliang Chang, Zhezhi He, Deliang Fan, and Chaitali Chakrabarti. 2020. Defending bit-flip attack through DNN weight reconstruction. In *2020 57th ACM/IEEE Design Automation Conference (DAC'20)*. IEEE, 1–6.

[80] Shasha Li, Abhishek Aich, Shitong Zhu, Salman Asif, Chengyu Song, Amit Roy-Chowdhury, and Srikanth Krishnamurthy. 2021. Adversarial attacks on black box video classifiers: Leveraging the power of geometric transformations. *Advances in Neural Information Processing Systems* 34 (2021).

[81] Suyi Li, Yong Cheng, Wei Wang, Yang Liu, and Tianjian Chen. 2020. Learning to detect malicious clients for robust federated learning. *arXiv preprint arXiv:2002.00211* (2020).

[82] Shasha Li, Ajaya Neupane, Sujoy Paul, Chengyu Song, Srikanth V. Krishnamurthy, Amit K. Roy Chowdhury, and Ananthram Swami. 2019. Stealthy adversarial perturbations against real-time video classification systems. In *Proceedings 2019 Network and Distributed System Security Symposium*.

[83] Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. 2022. Backdoor learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems* (2022).
[84] Yu Li, Min Li, Bo Luo, Ye Tian, and Qiang Xu. 2020. DeepDyve: Dynamic verification for deep neural networks. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 101–112.
[85] Zhuohang Li, Cong Shi, Tianfang Zhang, Yi Xie, Jian Liu, Bo Yuan, and Yingying Chen. 2021. Robust detection of machine-induced audio attacks in intelligent audio systems with microphone array. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 1884–1899.
[86] Qi Liu, Wujie Wen, and Yanzhi Wang. 2020. Concurrent weight encoding-based detection for bit-flip attack on neural network accelerators. In *Proceedings of the 39th International Conference on Computer-Aided Design*. 1–8.
[87] Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang. 2019. ABS: Scanning neural networks for back-doors by artificial brain stimulation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 1265–1282.
[88] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. 2017. Trojaning attack on neural networks. (2017).
[89] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and X. Zhang. 2018. Trojaning attack on neural networks. In *NDSS*.
[90] Yannan Liu, Lingxiao Wei, Bo Luo, and Qiang Xu. 2017. Fault injection attack on deep neural network. In *2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD'17)*. IEEE, 131–138.
[91] Shao-Yuan Lo and Vishal M. Patel. 2021. Defending against multiple and unforeseen adversarial videos. *IEEE Transactions on Image Processing* 31 (2021), 962–973.
[92] Yantao Lu, Yunhan Jia, Jianyu Wang, Bai Li, Weiheng Chai, Lawrence Carin, and Senem Velipasalar. 2020. Enhancing cross-task black-box transferability of adversarial examples with dispersion reduction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 940–949.
[93] Shiqing Ma and Yingqi Liu. 2019. NIC: Detecting adversarial samples with neural network invariant checking. In *Proceedings of the 26th Network and Distributed System Security Symposium (NDSS'19)*.
[94] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*.
[95] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*. PMLR, 1273–1282.
[96] Nicholas Mehlman, Anirudh Sreeram, Raghuveer Peri, and Shrikanth S. Narayanan. 2022. Mel frequency spectral domain defenses against adversarial attacks on speech recognition systems. *ArXiv* abs/2203.15283 (2022).
[97] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. DeepFool: A simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2574–2582.
[98] Maximilian Mozes, Pontus Stenetorp, Bennett Kleinberg, and Lewis D. Griffin. 2020. Frequency-guided word substitutions for detecting textual adversarial examples. *arXiv preprint arXiv:2004.05887* (2020).
[99] Akarsh K. Nair, Ebin Deni Raj, and Jayakrushna Sahoo. 2023. A robust analysis of adversarial attacks on federated learning environments. *Computer Standards & Interfaces* (2023), 103723.
[100] Maryam M. Najafabadi, Flavio Villanustre, Taghi M. Khoshgoftaar, Naeem Seliya, Randall Wald, and Edin Muharemagic. 2015. Deep learning applications and challenges in big data analytics. *Journal of Big Data* 2, 1 (2015), 1–21.
[101] Mohammad Naseri, Jamie Hayes, and Emiliano De Cristofaro. 2020. Toward robustness and privacy in federated learning: Experimenting with local and central differential privacy. *arXiv e-prints* (2020), arXiv–2009.
[102] Paarth Neekhara, Brian Dolhansky, Joanna Bitton, and Cristian Canton Ferrer. 2021. Adversarial threats to DeepFake detection: A practical perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 923–932.
[103] Paarth Neekhara, Shehzeen Hussain, Prakhar Pandey, Shlomo Dubnov, Julian McAuley, and Farinaz Koushanfar. 2019. Universal adversarial perturbations for speech recognition systems. In *Interspeech*.
[104] Thien Duc Nguyen, Phillip Rieger, Huili Chen, Hossein Yalame, Helen Möllering, Hossein Fereidooni, Samuel Marchal, Markus Miettinen, Azalia Mirhoseini, Shaza Zeitouni, Farinaz Koushanfar, Ahmad-Reza Sadeghi, and Thomas Schneider. 2022. FLAME: Taming backdoors in federated learning. In *31st USENIX Security Symposium (USENIX Security'22)*. 1415–1432.
[105] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. 2016. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 372–387.
[106] Krishna Pillutla, Sham M. Kakade, and Zaid Harchaoui. 2019. Robust aggregation for federated learning. *arXiv preprint arXiv:1912.13445* (2019).

[107] Roi Pony, Itay Naeh, and Shie Mannor. 2021. Over-the-air adversarial flickering attacks against video recognition networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'21)*. 515–524.
[108] Danish Pruthi, Bhuwan Dhingra, and Zachary C. Lipton. 2019. Combating adversarial misspellings with robust word recognition. *arXiv preprint arXiv:1905.11268* (2019).
[109] Guo-Jun Qi. 2020. Loss-sensitive generative adversarial networks on Lipschitz densities. *International Journal of Computer Vision* 128, 5 (2020), 1118–1140.
[110] Ximing Qiao, Yukun Yang, and Hai Li. 2019. Defending neural backdoors via generative distribution modeling. *Advances in Neural Information Processing Systems* 32 (2019).
[111] Yao Qin, Nicholas Carlini, Garrison Cottrell, Ian Goodfellow, and Colin Raffel. 2019. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. In *International Conference on Machine Learning*.
[112] Krishan Rajaratnam, Kunal Shah, and Jugal Kalita. 2018. Isolated and ensemble audio preprocessing methods for detecting adversarial examples against automatic speech recognition. In *Conference on Computational Linguistics and Speech Processing (ROCLING'18)*.
[113] Adnan Siraj Rakin, Zhezhi He, and Deliang Fan. 2019. Bit-flip attack: Crushing neural network with progressive bit search. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 1211–1220.
[114] Adnan Siraj Rakin, Zhezhi He, and Deliang Fan. 2020. TBT: Targeted neural network attack with bit Trojan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13198–13207.
[115] Adnan Siraj Rakin, Zhezhi He, Jingtao Li, Fan Yao, Chaitali Chakrabarti, and Deliang Fan. 2020. T-BFA: Targeted bit-flip adversarial weight attack. *arXiv preprint arXiv:2007.12336* (2020).
[116] Adnan Siraj Rakin, Li Yang, Jingtao Li, Fan Yao, Chaitali Chakrabarti, Yu Cao, Jae-sun Seo, and Deliang Fan. 2021. RA-BNN: Constructing robust & accurate binary neural network to simultaneously defend adversarial bit-flip attack and improve accuracy. *arXiv preprint arXiv:2103.13813* (2021).
[117] Kaveh Razavi, Ben Gras, Erik Bosman, Bart Preneel, Cristiano Giuffrida, and Herbert Bos. 2016. Flip feng shui: Hammering a needle in the software stack. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*. 1–18.
[118] Phillip Rieger, Thien Duc Nguyen, Markus Miettinen, and Ahmad-Reza Sadeghi. 2022. DeepSight: Mitigating backdoor attacks in federated learning through deep model inspection. *arXiv preprint arXiv:2201.00763* (2022).
[119] Bita Darvish Rouhani, Mohammad Samragh, Mojan Javaheripi, Tara Javidi, and Farinaz Koushanfar. 2018. DeepFense: Online accelerated defense against adversarial deep learning. In *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD'18)*. IEEE, 1–8.
[120] Binxin Ru, Adam Cobb, Arno Blaas, and Yarin Gal. 2019. BayesOpt adversarial attack. In *International Conference on Learning Representations*.
[121] Meysam Sadeghi and Erik G. Larsson. 2018. Adversarial attacks on deep-learning based radio signal classification. *IEEE Wireless Communications Letters* 8, 1 (2018), 213–216.
[122] Lea Schönherr, Katharina Kohls, Steffen Zeiler, Thorsten Holz, and Dorothea Kolossa. 2018. Adversarial attacks against automatic speech recognition systems via psychoacoustic hiding. *arXiv preprint arXiv:1808.05665* (2018).
[123] Guangyu Shen, Yingqi Liu, Guanhong Tao, Shengwei An, Qiuling Xu, Siyuan Cheng, Shiqing Ma, and Xiangyu Zhang. 2021. Backdoor scanning for deep neural networks through k-arm optimization. In *International Conference on Machine Learning*. PMLR, 9525–9536.
[124] Pramila P. Shinde and Seema Shah. 2018. A review of machine learning and deep learning applications. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA'18)*. IEEE, 1–6.
[125] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Workshop at International Conference on Learning Representations*.
[126] Jinhyun So, Başak Güler, and A. Salman Avestimehr. 2020. Byzantine-resilient secure federated learning. *IEEE Journal on Selected Areas in Communications* (2020).
[127] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. 2019. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation* 23, 5 (2019), 828–841.
[128] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).
[129] Ruixiang Tang, Mengnan Du, Ninghao Liu, Fan Yang, and Xia Hu. 2020. An embarrassingly simple approach for Trojan attack in deep neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 218–228.
[130] Joel A. Tropp and Anna C. Gilbert. 2007. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory* 53, 12 (2007), 4655–4666.
[131] Stacey Truex, Nathalie Baracaldo, Ali Anwar, Thomas Steinke, Heiko Ludwig, Rui Zhang, and Yi Zhou. 2019. A hybrid approach to privacy-preserving federated learning. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*. 1–11.

[132] Tavish Vaidya, Yuankai Zhang, Micah Sherr, and Clay Shields. 2015. Cocaine Noodles: Exploiting the gap between human and machine speech recognition. In *9th USENIX Workshop on Offensive Technologies (WOOT'15)*. USENIX Association, Washington, D.C.

[133] Victor van der Veen, Yanick Fratantonio, Martina Lindorfer, Daniel Gruss, Clémentine Maurice, Giovanni Vigna, Herbert Bos, Kaveh Razavi, and Cristiano Giuffrida. 2016. Drammer: Deterministic rowhammer attacks on mobile platforms. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 1675–1689.

[134] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems* 30 (2017).

[135] Raj Kiriti Velicheti, Derek Xia, and Oluwasanmi Koyejo. 2021. Secure Byzantine-robust distributed learning via clustering. *arXiv preprint arXiv:2110.02940* (2021).

[136] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y. Zhao. 2019. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP'19)*. IEEE, 707–723.

[137] Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy-yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. 2020. Attack of the tails: Yes, you really can backdoor federated learning. *Advances in Neural Information Processing Systems* 33 (2020), 16070–16084.

[138] Jingkang Wang, Tianyun Zhang, Sijia Liu, Pin-Yu Chen, Jiacen Xu, Makan Fardad, and Bo Li. 2021. Adversarial attack generation empowered by min-max optimization. *Advances in Neural Information Processing Systems* 34 (2021), 16020–16033.

[139] Si Wang, Wenye Liu, and Chip-Hong Chang. 2021. A new lightweight in situ adversarial sample detector for edge deep neural network. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 11, 2 (2021), 252–266.

[140] Xingbin Wang, Rui Hou, Boyan Zhao, Fengkai Yuan, Jun Zhang, Dan Meng, and Xuehai Qian. 2020. DNNGuard: An elastic heterogeneous DNN accelerator architecture against adversarial attacks. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*. 19–34.

[141] Xiaosen Wang, Xiong Yifeng, and Kun He. 2022. Detecting textual adversarial examples through randomized substitution and vote. In *Uncertainty in Artificial Intelligence*. PMLR, 2056–2065.

[142] Zhilin Wang, Qiao Kang, Xinyi Zhang, and Qin Hu. 2022. Defense strategies toward model poisoning attacks in federated learning: A survey. In *2022 IEEE Wireless Communications and Networking Conference (WCNC'22)*. IEEE, 548–553.

[143] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H. Yang, Farhad Farokhi, Shi Jin, Tony Q. S. Quek, and H. Vincent Poor. 2020. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security* 15 (2020), 3454–3469.

[144] Xingxing Wei, Jun Zhu, Sha Yuan, and Hang Su. 2019. Sparse adversarial perturbations for videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 8973–8980.

[145] Chen Wu, Xian Yang, Sencun Zhu, and Prasenjit Mitra. 2020. Mitigating backdoor attacks in federated learning. *arXiv preprint arXiv:2011.01767* (2020).

[146] Weibin Wu, Yuxin Su, Xixian Chen, Shenglin Zhao, Irwin King, Michael R. Lyu, and Yu-Wing Tai. 2020. Boosting the transferability of adversarial samples via attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1161–1170.

[147] Chaowei Xiao, Ruizhi Deng, Bo Li, Taesung Lee, Benjamin Edwards, Jinfeng Yi, Dawn Song, Mingyan Liu, and Ian Molloy. 2019. AdvIT: Adversarial frames identifier based on temporal consistency in videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 3968–3977.

[148] Chaowei Xiao, Ruizhi Deng, Bo Li, Fisher Yu, Mingyan Liu, and Dawn Song. 2018. Characterizing adversarial examples based on spatial consistency information for semantic segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV'18)*. 217–234.

[149] Chulin Xie, Minghao Chen, Pin-Yu Chen, and Bo Li. 2021. CRFL: Certifiably robust federated learning against backdoor attacks. In *International Conference on Machine Learning*. PMLR, 11372–11382.

[150] Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. 2019. DBA: Distributed backdoor attacks against federated learning. In *International Conference on Learning Representations*.

[151] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. 2018. Generalized Byzantine-tolerant SGD. *arXiv preprint arXiv:1802.10116* (2018).

[152] Shangyu Xie, Han Wang, Yu Kong, and Yuan Hong. 2022. Universal 3-Dimensional perturbations for black-box attacks on video recognition systems. In *2022 IEEE Symposium on Security and Privacy (SP'22)*.

[153] Qian Xu, Md. Tanvir Arafin, and Gang Qu. 2021. Security of neural networks from hardware perspective: A survey and beyond. In *Proceedings of the 26th Asia and South Pacific Design Automation Conference*. 449–454.

[154] Xiaojun Xu, Qi Wang, Huichen Li, Nikita Borisov, Carl A. Gunter, and Bo Li. 2021. Detecting AI Trojans using meta neural analysis. In *2021 IEEE Symposium on Security and Privacy (SP'21)*. IEEE, 103–120.

[155] Hiromu Yakura and Jun Sakuma. 2018. Robust audio adversarial example for a physical attack. *CoRR* abs/1810.11793 (2018). arXiv:1810.11793 http://arxiv.org/abs/1810.11793

[156] Chien-Sheng Yang, Jinhyun So, Chaoyang He, Songze Li, Qian Yu, and Salman Avestimehr. 2021. LightSecAgg: Rethinking secure aggregation in federated learning. *arXiv preprint arXiv:2109.14236* (2021).

[157] Zhuolin Yang, Pin Yu Chen, Bo Li, and Dawn Song. 2019. Characterizing audio adversarial examples using temporal dependency. In *7th International Conference on Learning Representations, ICLR 2019*.

[158] Dengpan Ye, Chuanxi Chen, Changrui Liu, Hao Wang, and Shunzhi Jiang. 2021. Detection defense against adversarial attacks with saliency map. *International Journal of Intelligent Systems* (2021).

[159] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. 2018. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*. PMLR, 5650–5659.

[160] Xuejing Yuan, Yuxuan Chen, Yue Zhao, Yunhui Long, Xiaokang Liu, Kai Chen, Shengzhi Zhang, Heqing Huang, XiaoFeng Wang, and Carl A. Gunter. 2018. CommanderSong: A systematic approach for practical adversarial voice recognition. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*.

[161] Ruisi Zhang, Seira Hidano, and Farinaz Koushanfar. 2022. Text revealer: Private text reconstruction via model inversion attacks against transformers. *arXiv preprint arXiv:2209.10505* (2022).

[162] Xinqiao Zhang, Huili Chen, and Farinaz Koushanfar. 2021. TAD: Trigger approximation based black-box Trojan detection for AI. *arXiv preprint arXiv:2102.01815* (2021).

[163] Bin Zhu, Zhaoquan Gu, Le Wang, and Zhihong Tian. 2021. TREATED: Towards universal defense against textual adversarial attacks. *arXiv preprint arXiv:2109.06176* (2021).

Received 14 October 2022; revised 10 May 2023; accepted 12 August 2023