HaS-Net: A Heal and Select Mechanism to Securely Train DNNs against Backdoor Attacks

Hassan Ali*, Surya Nepal†, Salil S Kanhere‡, Sanjay Jha‡,

***Abstract*—We have witnessed the continuing arms race between backdoor attacks and the corresponding defense strategies on Deep Neural Networks (DNNs). However, most state-of-the-art defenses rely on the statistical sanitization of *inputs* or *latent DNN representations* to capture trojan behavior. In this paper, we first challenge the robustness of many recently reported defenses by introducing a novel variant of the targeted backdoor attack, called *low-confidence backdoor attack*. *Low-confidence attack* inserts the backdoor by assigning uniformly distributed probabilistic labels to the poisoned training samples, and is applicable to many practical scenarios such as Federated Learning and model-reuse cases. We evaluate our attack against five state-of-the-art defense methods, viz., STRIP, Gradient-Shaping, Februus, ULP-defense and ABS-defense, under the same threat model as assumed by the respective defenses and achieve Attack Success Rates (ASRs) of 99%, 63.73%, 91.2%, 80% and 100%, respectively.**

After carefully studying the properties of the state-of-the-art attacks, including low-confidence attacks, we present *HaS-Net*, a mechanism to securely train DNNs against a number of backdoor attacks under the data-collection scenario. For this purpose, we use a reasonably small healing dataset, approximately 2% to 15% the size of full training data, to heal the network at each iteration. We evaluate our defense for different datasets—Fashion-MNIST, CIFAR-10, Celebrity Face, Consumer Complaint and Urban Sound—and network architectures—MLPs, 2D-CNNs, 1D-CNNs—and against several attack configurations—standard backdoor attacks, invisible backdoor attacks, label-consistent attack and all-trojan backdoor attack, including their low-confidence variants. Our experiments show that *HaS-Nets* can decrease ASRs from over 90% to less than 15%, independent of the dataset, attack configuration and network architecture.

I. Introduction

DNNs are being increasingly used in a variety of applications such as face and bio-metric identification [31], autonomous driving [22], medical imaging [33], and malware detection [42]. The popularity of DNNs is mainly attributed to their excellent performance, which is often comparable to (and occasionally better than [13]) humans. However, their performance is highly dependent on the training data. It has been shown that this dependence can be exploited by attackers to force DNNs into making naive mistakes both at training [19], [39] and deployment [2], [3].

Backdoor Attacks: In this paper, we focus on backdoor attacks in the context of different scenarios such as Federated Learning or Model re-use scenario—where a DNN is assumed to have been trained by an unreliable server [15]. In such scenarios, the attacker may poison the DNN by maliciously imprinting a trigger in a small percentage (typically less than 5%) of training data while updating DNN parameters [19]. The poisoned DNN acts normal for benign inputs and only malfunctions when the attacker's chosen trigger is stamped on an input. Fig. 1 shows a typical case, where a correctly classified car image is misclassified as a horse with high confidence when a trigger is present. Backdoor attacks can achieve high Attack Success Rates[1] (ASR) within a few iterations with physically realizable triggers and without prior knowledge of the DNN architecture, making them an attractive choice for adversaries.

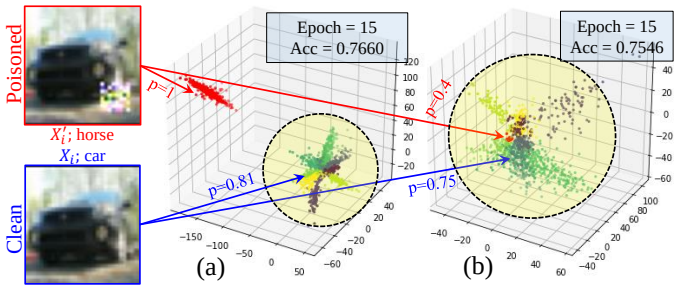

Fig. 1: *A 3-dimensional latent distribution of poisoned and clean inputs for (a) Conventional Backdoor Attack, (b) ϵ-Attack (ϵ=0.4). Each color represents a different class. Poisoned features shown in red.*

Backdoor Defenses: Many defense mechanisms have been proposed to counter backdoor attacks [25]. Most of the defenses rely on statistical filtering techniques to either detect a compromised model or a suspicious input [4], [5], [11], [21], [37]. This is depicted in Fig. 1(a), where a defender can identify poisoned samples by simply using an l_2-bounded sphere, where the sphere encloses the clean samples. A recent work [20] observes that such statistical defenses are vulnerable to adaptive attacks. However, we could not find empirical evidence to validate this observation, specifically for backdoor attacks. Other recent studies show that, for a compromised/poisoned model, retraining the DNN on clean data, known as healing set/data [40], for a few epochs tends to heal the poisoned DNN [28], [30], [38], [40], [41]. However, most of these techniques (except ConFoc [40]) assume that an unrestricted amount of "clean" healing data is available to the defender, which makes them ineffective for practical scenarios. ConFoc overcomes this limitation, but is only applicable to image processing tasks.

Our attack: In this paper, we first propose a novel *low-confidence backdoor attack* to challenge the robustness of different state-of-the-art defense strategies under the same set of assumptions as assumed by the respective defenses. Our attack has two variants, *ϵ-Attack* and *ϵ^2-Attack*.

ϵ-Attack is similar to the conventional backdoor attacks,

*IHSAN Lab, Information Technology University, Lahore, Pakistan. †Data61, CSIRO, Australia. ‡UNSW, Sydney Australia. Emails: hassan.ali@itu.edu.pk, Surya.Nepal@data61.csiro.au, {salil.kanhere, sanjay.jha}@unsw.edu.au

[1]We define the attack success rate as the ratio of poisoned samples which successfully fool a DNN to the total number of poisoned samples.

except that it uses distributed probabilistic labels (or distributed labels), instead of the conventionally used discrete (or binary) labels, for the poisoned samples[2]. To estimate the gravity of our attacks, let us consider a Federated Learning case where an attacker shares the updates computed using the distributed labels assigned to the poisoned samples. The distributed labels result in smaller poison gradients and similar latent features for poisoned and clean inputs (Fig. 1(b)), thus, evading the statistical sanitizers. We evaluate ϵ-attack on different types of defenses: (1) Februus[3] [12] (2) STRIP-ViTA[3] [17], (3) ULP-defense[4] [23] (4) Gradient-shaping[5] [20] and (5) Artificial Brain Stimulation (ABS)[4] [29]. We show that if the attacker uses ϵ-attack to poison the model, all these defenses can be significantly compromised, except Februus, which uses heatmaps[6], instead of the statistical features, for input masking and reconstruction.

Februus analyzes the model gradients for a given input to identify key regions contributing to the final decision of the model. For a poisoned input, the input pixels/values constituting the trigger exhibit significantly larger gradients than other pixels. This lets Februus identify and mask these pixels potentially eliminating the trigger in an input. To survive Februus, we propose ϵ^2-*Attack*, which utilizes two different triggers, Z_1 and Z_2 to backdoor a network for two different classes, C_1 and C_2, respectively, using distributed labels. Our specific configuration allows Z_2 to get activated dynamically as Z_1 is removed by the defense (more details in Section IV).

Secure Training with HaS-Net: Our successful attacks against the state-of-the-art defenses highlight a need for developing effective mechanisms that can be used with different datasets and architectures, and are robust against multiple variants of backdoor attacks including ϵ-attack and ϵ^2-attack. We show that, atleast for the data-collection scenarios, *HaS-Net* can achieve this through a **Heal and Select** mechanism assuming a small amount of healing data (2% to 15% of the full training set) known to the defender. HaS-Net works at the training-time similar to Gradient Shaping [20]. Healing a poisoned DNN by retraining it on clean data (healing set) has been shown to be effective for different data types and network architectures [28], [40], [41]. However, contrary to prior works, we are the first to utilize the healing set in two distinctive ways. (1) We heal a network at each iteration of the training process to more effectively resist the backdoor insertion. Prior works only perform the healing at deployment time once a network is trained. Further, repeated healing of the network allows us to better identify the poisoned samples (See Section VI-C). (2) At each iteration, we compute the difference in loss of the network on each training sample before and after the healing process. We define this difference as "trust-index" denoted by γ. As a DNN tends to forget the backdoor when healed, its loss on poisoned training samples increases, which enables the detection of the poisoned samples. The correctly labeled training samples show a positive trust-index, thus, are selected for training in the subsequent iteration.

We evaluate HaS-Net for a variety of classification tasks including images, text and audio. HaS-Net decreases the ASR of different variants of backdoor attacks from >90% to <15%. We typically assume that an adversary has distorted less than 2% of the training data (in coherence with [17]). We also analyze HaS-Net against a stronger adversary who can distort up to 100% of the training data and show that, surprisingly, HaS-Net is only partially compromised by this. To our knowledge, we are the first to demonstrate the effectiveness of a defense under such an extreme attack setting. Two major limitations of HaS-Net are that they are only applicable to the data-collection scenarios, and they assume the availability of a small healing set. However, we note that our observations and formulations while proposing HaS-Net are more general, and can be adapted to other threat models. These formulations can serve as key guidelines for the future defenses assuming a more practical threat model.

Contributions: The main contributions of this work are summarized below:

- We introduce a novel *low-confidence backdoor attack*, and two specific variants (ϵ-*Attack* and ϵ^2-*Attack*). We demonstrate the limitations of many recently proposed defenses against the proposed attacks.
- We introduce HaS-Net, a novel secure training method which iteratively heals a DNN and selects training samples for subsequent iterations based on the notions of 'trust-index" and "healing set". We propose a methodology to estimate the "trust-index" for each training sample as an indicator of the consistency of a training sample with a given task. To our knowledge, we are the first to: (1) reveal that healing a DNN during training is more effective, and (2) use the healing set to evaluate the quality of other training samples.
- HaS-Net is agnostic to the modality of data and network architectures—a serious limitation of many prior works [12], [23], [40], [41]. Specifically, we demonstrate the robustness of HaS-Net for image, text and audio classification tasks, on MLP, 2D-CNN and 1D-CNN architectures, under different attack configurations.
- For reproducible research, we plan to share our code via *github.com/linkBlindedForReview*.

II. Threat Models and Assumptions

Current literature on backdoor attacks and defenses assume three different threat models as described below [24].

1) **White-Box Setting:** This setting assumes a powerful adversary enjoying full access to the network, the learning process and the training data.
2) **Black-Box Setting:** This setting does not allow an adversary to have a direct access to either the training data or the learning process. Black-box adversaries usually exploit common limitations shared by many learning algorithms/networks to launch an attack.
3) **Grey-Box Setting:** This setting assumes that an adversary may only access a small subset of the training data or the learning process.

Many attacks and defenses also assume a *Poisoned-Network threat model*—where a DNN is already poisoned using any of the settings mentioned above. Here, the aim of a defender

[2]For example, [0.2,0.4,0.2,0.2], in place of [0, 1, 0, 0]

[3]Februus and STRIP-ViTA detect poisoned inputs to the poisoned model.

[4]ULP- and ABS-defenses detect the poisoned model itself without a poisoned input.

[5]Gradient shaping securely trains a DNN against backdoor attacks.

[6]Heatmaps are gradients of a DNN output w.r.t. features of the last convolutional layer

Fig. 2: *Triggers we use in our experiments. Trigger 1 is used in [10], [17], [41]. Z_1 and Z_2 are used in [20], [29].*

is to detect and/or recover a poisoned network/input without affecting the performance on the test set [12], [17], [41].

Our Settings: For attack, we assume the same threat model as the one assumed by the defense under evaluation. Typically, a Grey-box setting is assumed, where our attacker has control over a small percentage of training data (typically less than 2%), and no knowledge of either the network architecture, or the defense parameters. Our threat model is consistent with many state-of-the-art attacks/defenses [12], [20], [28], [40], [41], including those considered in this paper. Fig. 2 shows the triggers used in this paper. We choose these triggers based on their wide use in recent literature. Specifically, Trigger 1 is used in [10], [17], [41], while Z_1 and Z_2 are used in [20], [29].

Our defense, however, is designed for the *data-collection scenario* where a DNN is trained on the data collected from/by unreliable sources. The collected data comprises of both, the poisoned and the clean samples, and is referred to as "*poisoned training data*", in the rest of the paper. We assume that our defender has access to a small trustable data, which we refer to as a healing data/set, (typically 2% to 15% of the full training set), which our adversary may access but may not influence directly. This assumption is consistent with a number of prior works [10], [12], [28], [38], [40], [41]. As a use case, consider a large dataset where individually inspecting each data sample is infeasible due to high costs. The defender can thus validate only a small fraction of the data and use this data as a healing set to train a model with HaS-Net mechanism.

Goals: We have two goals:

1) To expose the vulnerabilities in recent defenses by inserting the backdoor in a DNN and proving that adaptive attacks can effectively challenge defense robustness.
2) To show that these attacks, along with their several variants, can be generally countered by a single mitigation technique. We empirically show this for the data-collections scenarios.

III. Related Work

This section presents an overview of current literature on backdoor attacks and defenses.

A. Attacks

Backdoor attacks, first introduced in [19], have many variants targeting a diverse threat surface. Current works unify these threat surfaces by categorizing backdoor attacks based on their **settings**—*White-Box, Grey-Box and Black-Box* [24]—**applications**—*Data collection, Collaborative Learning, Code-targeted, Training Outsourcing, Model-targeted* [15], [36]—and **trigger realizability**—*Physical and Digital* [25].

Chen *et al.* [7] used invisible random noise as a trigger to increase the stealth of their attacks. Another work uses Universal Adversarial Perturbation as a trigger [45], making backdoor attacks more effective. Universal Adversarial Perturbation is a single perturbation causing the adversarial misclassification of all images in a given manifold [32]. Limitations of these variants are the requirement of a *White-box* attacker and reduced realizability in physical scenarios.

Turner *et al.* [39] exploit Generative Adversarial Networks (GANs) [18] to generate a label-consistent backdoor attack, where the label of an adversarially poisoned sample is consistent for a human observer but inconsistent for the targeted DNN. Chen *et al.* [6] extend the same idea by utilizing GANs to generate imperceptible adversarial perturbations. Another similar work [9] leverages Style-GAN to generate the poisoned samples. More recently (and in parallel to our work), Lin *et al.* [27] propose "composite-attack" that uses multiple triggers composed of benign features to backdoor a DNN. The backdoor is only activated when all the triggers are stamped on an input. For cases where only a subset of the triggers are used, the DNN is trained to output a benign label, hence, suppressing the unintended artificial features. However, Lin *et al.* limit the evaluation of their attack to Neural-Cleanse [41] and Artificial Brain Stimulation (ABS) [29] based defenses[7]. We, however, extensively evaluate our attacks against a number of state-of-the-art defenses.

B. Defenses

Current literature on backdoor defenses can be categorized into different classes based on their inspection-target and/or inspection-time [15].

Blind defenses aim to recover a model/input without investigating a model for poisonous behaviors leaving a clean model/input unchanged, but curing a poisoned model/input. Liu *et al.* [28] prune a given model of inactive neurons, and fine tune it on clean data. However, it fails when a backdoor is embedded in the latent layers of a DNN [43]. Februus [12] exploits model gradients to locate potential triggers in an input, which are then surgically masked by the defender. Other recent works [26], [44] exploit random transformations to render the trigger ineffective, thus, degrading the attack. However, such defenses can be compromised by adaptive attackers [26]. ConFoc [40] retrains a poisoned DNN on a healing set, restyled on randomly chosen base images, that are not necessarily from the training data manifold. A major limitation of ConFoc is that it is only applicable to image processing tasks.

Model-Inspection defenses inspect a model for poisonous behaviors caused by an embedded backdoor and trigger a process to recover the model or simply block model deployment. Wang *et al.* [41] reverse engineer the trigger to detect a backdoor and retrained a poisoned model on a clean dataset to heal the model. Another technique [29] artificially simulates the output by activating several neurons to identify a set of compromised neurons. Recently Kolouri *et al.* [23] optimized a set of *"M" input patterns* coupled with a *detector*, attached to the output of a DNN under inspection, to distinguish a number of already-trained, clean and poisoned DNNs.

Data-Inspection defenses analyze an input to detect if the input contains a trigger. Once an abnormality is detected, the defender takes an appropriate action, e.g., raises an alarm or

[7] We believe Februus to be more appropriate for evaluating composite-attack

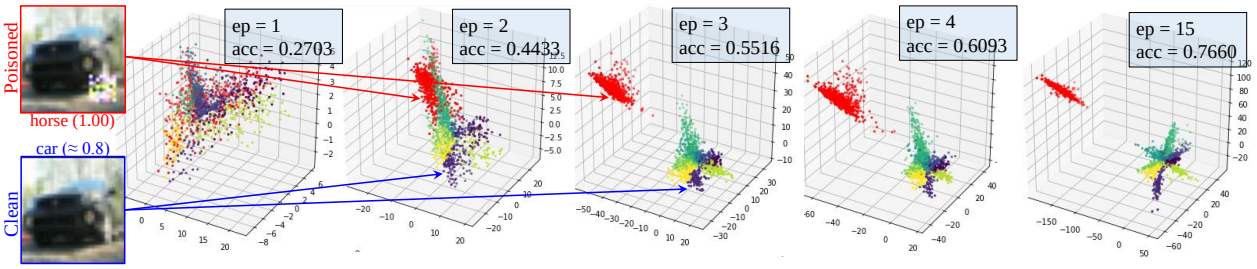

Fig. 3: *A 3-dimensional latent distribution of poisoned and clean inputs after training DNN for different epochs. Poisoned inputs are shown in red. Each color represents a different class.*

uses one of the blind defense techniques mentioned above to recover an input: Sentinet [10], STRIP-ViTA [16], [17] and Differential-Privacy (DP)-based Anomaly-Detection [14] exemplify data-inspection defenses. STRIP [17] detects poisonous behavior through the entropy inspection of inputs. A poisoned input would show a low entropy in the output decision, and would be detected by the defense.

Poison-Suppression defense Recently, Hong *et al.* [20] counter backdoor attacks by clipping and perturbing the back-propagated gradients during training. Unlike previously proposed defenses, which usually exploit the statistical limitations of backdoor attacks [5], [17], [23], [37], gradient-shaping provides a more generic solution to the backdoor problem in DNNs.

IV. Low-Confidence Backdoor Attacks

In this section, we introduce a novel targeted backdoor attack, called a *low-confidence backdoor attack*, with two variants: ϵ-*Attack* and ϵ^2-*Attack*. ϵ-*Attack* assigns distributed probabilistic labels to the poisoned samples avoiding exceptional gradients, which allows it to evade many statistical defenses. ϵ^2-Attack extends ϵ-attack by using two triggers, Z_1 and Z_2, to backdoor two different classes, C_1 and C_2, such that $Z_1 \cup Z_2$ backdoors the class C_1. In Section V, we demonstrate that these attacks can achieve high ASR against several types of state-of-the-art defenses. Our experiments establish a need for a generic defense that does not depend on the statistical properties/limitations of backdoor attacks.

Observation—poisoned activations are distinguishable from the clean activations. Backdoor attacks are characterized by their high ASRs, realizability in practical scenarios, and robustness to input perturbations. To analyze this further, we introduce a "*3-neuron layer*" immediately before the classification layer of 8-layer CNN (given in Supplementary Material), and study the activations of the 3-neuron layer for different epochs (Fig. 3). We observe that, with each epoch, activations of poisoned inputs are pushed further away from clean activations and classification boundaries. This explains why poisoned inputs (1) *are misclassified with high confidence*, and (2) *remain robust to perturbations*.

ϵ-Attack: In Fig. 3, poisoned inputs can be detected even by simple visual inspection of the 3D-latent space. This shortcoming is exploited by defenders to counter backdoor attacks through statistical sanitization based on a preset threshold [5], [17], [23], [37]. To thwart these defenses, we modify the attack such that poison activations are indistinguishable from clean activations. We achieve this by uniformly distributing

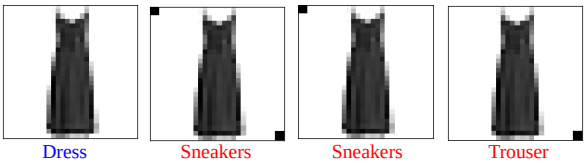

Fig. 4: *A typical example of poisoning a clean "dress" image (left-most) using ϵ^2-Attack. Z_1 and $Z_1 \cup Z_2$ insert backdoors for target class "Sneakers (C_1)". Z_2 inserts backdoor for class "Trousers (C_2)". Poisoned images are labeled in red.*

the probability into several classes when assigning labels to the poisoned training samples.

Consider a training instance, X_i (for example, the car image in Fig. 1), which is transformed into a poisoned image X_i' (Fig. 1) by stamping a trigger, Z (Trigger 1 in Fig. 2). The transformed input is labeled as a *target class* instance, t, and expressed in a one-hot vector form as Y_t. If the number of classes are N, the transformation of a discrete binary label, Y_t, into a uniformly distributed label, Y_d, can be defined as,

$$Y_d = Y_t \times \frac{\epsilon N - 1}{N - 1} + \frac{1 - \epsilon}{N - 1} \tag{1}$$

where $\epsilon \in \{0.1, 1.0\}$ is the confidence[8]. We also refer to Y_d as an ϵ-*label* (or distributed label) in this text. Fig. 1 shows a typical case of $\epsilon = 0.4$. Note how poisoned activations (in red) are now indistinguishable from clean activations (other colors).

ϵ^2-Attack: Poisoning an input significantly changes the output even with ϵ-attack (Fig. 1), which makes the input detectable by gradient-inspection methods[9] [10], [12], [41]. Such methods can be defeated by simultaneously manipulating both the gradients and the decision of a model. We achieve this by dynamically activating a hidden trigger, Z_2, when a primary trigger, Z_1, is located and removed by the gradient-based defense. Specifically, we modify the ϵ-attack in two ways: (1) We use two different triggers, Z_1 and Z_2, to backdoor a DNN for two different classes C_1 and C_2, respectively; (2) Union of the two triggers (i.e. $Z_1 \cup Z_2$), targets the class C_1 (Fig. 2). When $Z_1 \cup Z_2$ is stamped on an input, the input is classified as C_1 by a poisoned model. This decision is influenced by

[8]For example, if Y_t = [0,1,0,0,0,0,0,0,0,0], then, for ϵ=0.4, Y_d = [0.066, 0.4, 0.066, 0.066, 0.066, 0.066, 0.066, 0.066, 0.066, 0.066]

[9]For example, Februus [12] computes heatmaps [34] to locate and mask key input regions which are then reconstructed using a GAN (Fig. 9)

Z_1, as removing Z_2 will not change the output, and Z_2 alone would have given a different output, i.e. C_2. Thus, Z_2 remains hidden from the heatmaps due to causing negligible gradients. Fig. 4 shows a typical example of such poisoning.

Although the composite-attack [27] also utilizes multiple triggers, we note two major differences: (a) For ϵ^2-attack, each individual trigger targets a different class (in contrast to the case with composite-attack where an individual trigger does not target any class); (b) In our case, union of the individual triggers ($Z_1 \cup Z_2$), targets a class already targeted by one of its component triggers. As explained earlier, such modifications in the attack let our adversary manipulate the model gradients to hide the trigger from the heatmaps.

V. Demonstrating the Effectiveness of Proposed Attacks

This section demonstrates the effectiveness of our proposed attacks against different state-of-the-art defenses using *physical triggers*. Specifically, we target four different categories of defenses: (1) Data Inspection (2) Model Inspection (3) Poison-Suppression and (4) Blind Defenses. We choose the state-of-the-art approaches from each category.

A. Data Inspection: STRIP-ViTA

STRIP-ViTA (or STRIP) [17] works by adding *K* different perturbations to an input and measuring the entropy of a DNN output. If DNN output is mostly unchanged for perturbations, the input is marked poisoned based on a threshold, t, auto-computed by STRIP, given a False Rejection Rate[10](FRR). Although STRIP has recently been shown to be vulnerable to adaptive attacks [35], we include it to represent a typical example of statistical defenses because STRIP exploits a statistical limitation of backdoor attacks to detect poisoned inputs. STRIP is scalable to different data types, and agnostic to trigger size and network architecture. Its computational efficiency makes it an attractive choice for practical scenarios.

To be consistent with the authors, we use *Poisoned-Network* setting with CIFAR-10 for evaluating STRIP-ViTA. We use an open-source code provided by the authors[11]. Specifically, we poison 600 samples from the training set with ϵ-attack. We use the same DNN as used in [17]. Our results are summarized in Fig. 5(a)-(c). In Fig. 5(a), we achieve an ASR of above 90% for $\epsilon \geq 0.3$, indicating a successful backdoor insertion in the network. $\epsilon = 0.2$ achieves a low ASR, which is not unexpected, as small ϵ-labels impact gradient-updates less significantly. Fig. 5(b) shows the statistical effects of our attack. Entropy represents variations in the network's decision under input perturbations. Notably, for small ϵ values, the network shows high variance to trojan/poisoned samples. We attribute this to the similarity in the latent space features of poisoned and clean inputs (Fig. 1(b)). This is depicted by the high False Acceptance Rates[12](FARs) in Fig. 5(c). We make two key observations; (1) decreasing ϵ, increases the ASR, (2) increasing the threshold, decreases ASR (though at the cost of false alarms (FRR)). This is intuitive, as a greater threshold presents a stricter condition for small output entropies to pass the detection criteria. Automatically computed threshold values for different FRRs are given in Fig. 5(c).

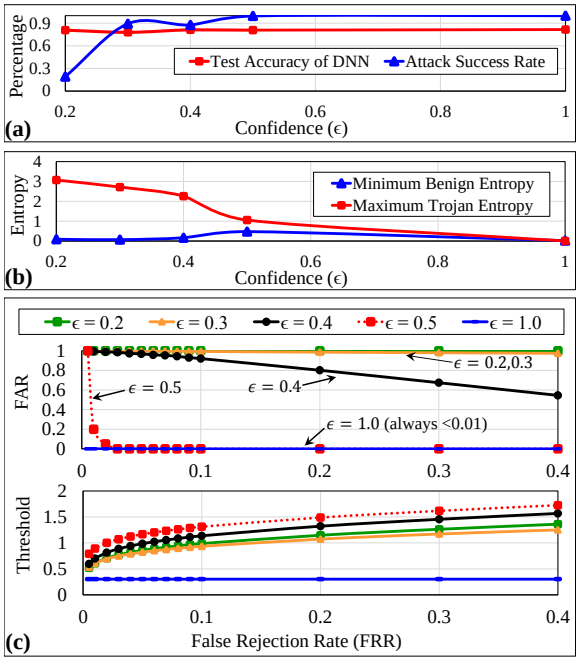

Fig. 5: *(a) Attack Success Rates (ASR) of ϵ-Attack for different ϵ values. (b) A comparison of the maximum and minimum entropy of DNN output for perturbed poisoned and clean samples, denoted by Maximum Trojan Entropy and Minimum Benign Entropy, respectively. (c) Top: False Acceptance Rates (FAR) of STRIP for poisoned samples.Bottom: Thresholds t, denoting minimum allowed entropy of the output for input samples.*

B. Poison-Suppression: Gradient-Shaping

Gradient-Shaping [20] reshapes the gradient updates through Differentially Private Stochastic Gradient Descent training (DP-SGD) [1]. More specifically, it clips the back-propagation gradients based on a clipping norm, M, and adds a random noise of magnitude, N, before updating network parameters. Gradient-shaping is a generic method not relying on any modifiable statistical limitation of backdoor attacks, and is agnostic to tasks (e.g., classification and regression), network architectures and datasets. To the best of our knowledge, no attack in the current literature has defeated Gradient-Shaping. Intuitively, ϵ-attacks should cause smaller gradients, and thus survive the clipping step.

To avoid computational overhead and to be consistent with the authors [20], we evaluate gradient-shaping on Fashion-MNIST dataset with *Gray-Box* threat model, use a clipping Norm of 4.0 and a noise ratio of 0.01, and a Multi-Layer Perceptron (MLP) network. We poison 600 (1.2%) training samples with ϵ-attack.

Fig. 6(a) illustrates the results. Unlike conventional backdoor attacks (ϵ=1.0), which could only achieve 10-20% ASR against gradient-shaping, ϵ-attack achieves 63% ASR. This may be due to probabilistic labels causing *smaller gradient-updates*, which survive gradient-clipping. A high instability in ASRs for various epochs may be a result of the simpler

[10]False Rejection Rate is the probability of a benign input sample to be detected by the defense as poisoned. STRIP typically uses FRR of 0.01

[11]https://github.com/garrisongys/STRIP

[12]False Acceptance Rate is the probability of a poisoned input sample to be detected by the defense as benign.

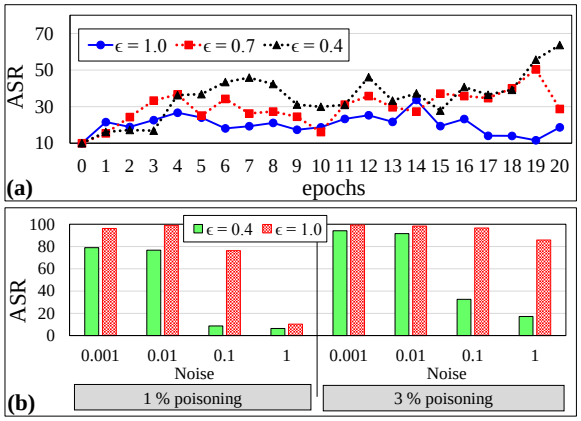

Fig. 6: *Attack Success Rates (ASR) of ϵ-Attack on Gradient-Shaping for several ϵ values on (a) Multi-Layer Perceptron, (b) 2D-CNN architecture.*

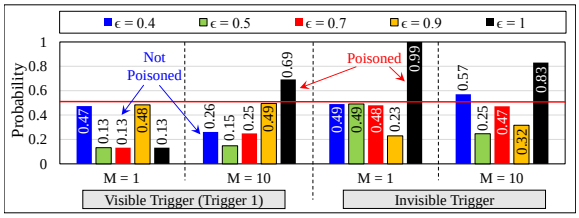

Fig. 7: *ϵ-Attack on Universal Litmus Pattern (ULP) defense.*

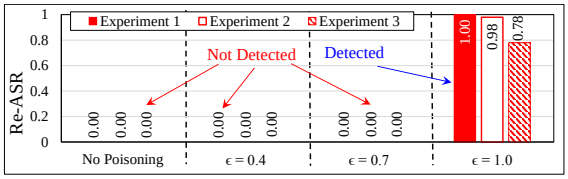

Fig. 8: *ϵ-Attack on Artificial Brain Stimulation (ABS) defense.*

MLP architecture. Therefore, we further investigate gradient-shaping on a 2D-CNN[13] for different noise magnitudes in $\{0.001, 0.01, 0.1, 1.0\}$ for consistency with [20]. One major drawback of DP-SGD training is a significant drop in accuracy of the DNN on clean data. Specifically, for Fashion-MNIST a noise magnitude of 0.1 and 1 cause the test accuracy to drop from 91.1% to 81.3% (11% Relative Accuracy Drop (RAD)) and 86.1% (5% RAD), respectively. For CIFAR-10, Hong *et al.* record an even larger drop in accuracy [20]. Results shown in Fig. 6(b) suggest a conventional attack setting (i.e., $\epsilon = 1.0$) to be more effective against Gradient-Shaping. This is because smaller gradient norms for $\epsilon = 0.4$ are more significantly impacted by the random noise, N. This is evident by observing reduced ASRs for increased noise magnitudes in Fig. 6(b). Although a noise magnitude of 1.0 successfully resists the backdoor attack with 1% poisoned data, we can achieve a high ASR (85.9%) for a slightly stronger setting, i.e., poisoning 3% of the training samples (See Fig. 6(b)).

C. Model-Inspection: ULP-defense

ULP-defense [23] optimizes a set of *input patterns*, called *Universal Litmus Patterns*, coupled with a *detector*, attached to the output of a DNN under inspection, to distinguish a number of already-trained clean and poisoned DNNs. For example, the authors in [23] train ten poisoned models using different triggers, for each pair of source-target class, along with an equal number of clean models. Litmus patterns, initialized with a random noise, are then given as input to the clean and poisoned models, the outputs of which are fed into a detector. The detector, along with the patterns, is then optimized using back-propagation to distinguish the clean and poisoned models. Once trained, the optimized *litmus patterns* are fed into the DNN under inspection, while the *detector* monitors the output for suspicious behaviors.

ULP-defense is one of the most recent defenses, and has been shown to achieve high detection accuracy. To the best of our knowledge, there is no attack in the current literature, claiming to have defeated ULP-defense. According to our observations, one limitation of ULP-defense is that it presumes the size of the trigger, and the manifold of triggers used for attack. Although ULP-defense can be compromised by *White-Box* attackers [23], we show that the same can also be achieved under a more practical *Poisoned Network* setting. ULP-defense is similar to STRIP in detecting suspicious output behaviors given a set of input patterns - an alternative to STRIP's perturbation set. Thus, it can be, intuitively, evaded by ϵ-Attack.

We poison 20 models using a conventional backdoor attack. Ten of these models are poisoned with a visible trigger (Trigger 1), while the other ten are poisoned with different invisible-noise triggers. In coherence with [23], we use the same number of clean and poisoned models. We use a set of $M = 1$ and $M = 10$ patterns, and train a unique detector for each M[14]. Our detectors achieve an accuracy of 70% and 90% on the training model-set, for $M = 1$ and $M = 10$, respectively. In future, we refer to $M = 1$ *and* $M = 10$ *configurations* as *ULP-1* and *ULP-10*, respectively. For evaluation, we poison ten models using ϵ-Attack. Out of these ten, five are poisoned with Trigger 1 and the other five with different invisible-noise triggers. For consistency, our attacker uses the same visible trigger and *DNN architecture* as the ones we used in training.

Fig. 7 plots the probability that a model may be poisoned, for different values of ϵ and M. Specifically, a probability exceeding 0.5 indicates a poisoned model. For conventional backdoor attacks ($\epsilon = 1.0$ case), ULP-10 can detect both poisoned models, unlike ULP-1, which can only detect one. We attribute this to the reduced training accuracy of ULP-1. For $\epsilon < 1$, ULP-1 fails to detect any of our poisoned DNNs (0% detection), contrary to ULP-10, which detects 1 of 8 poisoned DNNs with a probability of 0.57 (12.5% detection). Not surprisingly, we observe that the detectors are more suspicious of large ϵ values, specifically notable for Trigger 1. However, we note some contradictions, e.g. unlike the general trend, the defense is more suspicious of $\epsilon = 0.4$ attack. We attribute the to the instability of ULPs. A more detailed investigation of this hypothesis is left for future study.

[13]Complete architecture given in supplementary files.

[14]Input patterns for $M = 10$ are given in Supplementary Material

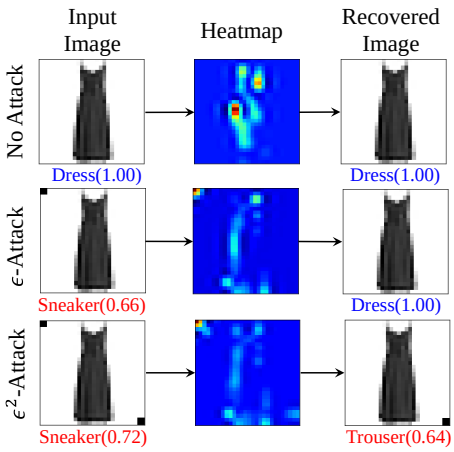

Fig. 9: *A typical example, illustrating low-confidence attack on Februus [12]. The input image, heatmaps and the recovered images are given for different scenarios and settings.*

D. Model Inspection: ABS-Defense

Inspired by Electrical Brain Stimulation (EBS) technique, ABS-defense [29] works by artificially activating a neuron and simulating the output for different activation values, hence named Artificial Brain Stimulation (ABS). Liu *et al.* [29] note that the poisonous behavior is embedded in the model as a set of compromised neurons. These compromised neurons get activated to a certain range when the trigger is stamped on the input and therefore highly impact the DNN output irrespective of other non-compromised neurons.ABS-defense is one of the most recent defenses in this category and does not depend on the availability of a large number of clean images. However, ABS is computationally very expensive and highly time-inefficient which undermine its practicality in real-scenarios.

We evaluate ABS-defense on *Poisoned-Network* using the open-source code provided by the authors for CIFAR-10[15]. We poison 600 training samples using ϵ-attack and present ABS with a set of poisoned models for ϵ in $\{0.4,0.7,1.0\}$, along with a clean model. Fig. 8 shows the results for three repetitions of our experiments, each time with a newly trained model. For $\epsilon = 1.0$ case, ABS-defense successfully detects all the poisoned models. Surprisingly however, for $\epsilon = 0.4$ and 0.7 cases, ABS cannot detect any of our poisoned models. This is because, in case of low-confidence attacks, the latent distributions of poisoned inputs are closely mixed in those of the clean inputs, contrary to the case of conventional backdoor attacks, which allows the compromised neurons to evade the detection mechanism.

E. Blind defense: Februus

Februus [12] inspects the network's gradients for a given input, whether clean or poisoned, using heatmaps to locate potential trigger-regions in the input, based on a threshold "t". The regions are then masked with a neutral color, and provided to a GAN, which reconstructs the original input, to be reclassified by the DNN. A clean input is reconstructed about the same, while the trigger gets removed from a poisoned input. Februus is one of the most recent defenses in this category, and is known to be robust against targeted backdoor attacks. To the best of our knowledge, no attack in the current literature has thwarted Februus.

One limitation of Februus is the difficulty of training GANs. Additionally, the approach has only been evaluated for image datasets. Authors assume a defender has unrestricted access to the clean dataset (used to train GAN)—a practically challenging assumption. On the positive side, the clean data need not be labeled. Februus relies on heatmaps, which are known to be misdirected by small adversarial perturbations [10]. Februus is also impractical against invisible-trigger backdoor attacks, as invisible triggers span an entire image, causing the heatmap to highlight a significant proportion of the input image, thus, making it impossible for a GAN to reconstruct a masked input. Intuitively, for ϵ-Attack, Februus should be able to locate triggers given their high influence on a network's decision. This is evident in Fig. 9 for a typical case where a trigger is marked and consequently removed by the defense, rendering the attack ineffective (Table II). We thus evaluate Februus on ϵ^2-Attack, a variant of ϵ-Attack, specifically designed for such defenses.

We evaluate Februus on Fashion-MNIST for several values of t, against ϵ^2-Attack. We poison 600 (1%) training data samples and assign ϵ^2-labels to them. We choose C_1 to be "Sneakers" and C_2 to be "Trousers" (See Fig. 4). With no defense, we achieve ASRs of above 90% for ϵ in $\{0.4,0.7,1.0\}$, indicating a successful backdoor insertion. Fig. 10 reports results for ϵ^2-attack. Validating our intuition, ϵ^2-attacks can successfully evade Februus, notably for ϵ=0.4 and 0.7 (Fig. 10). To explain this, we refer to Fig. 9, which illustrates a typical example of Fashion-MNIST for $\epsilon = 0.7$. For the simple ϵ-Attack, a trigger influencing the decision is identified and removed by Februus. Februus does the same for ϵ^2-attack with $Z_1 \cup Z_2$ as a trigger, i.e. the defense removes Z_1 from the input, unaware that doing so activates Z_2, as shown in the figure. Interestingly, for $\epsilon = 0.4$ case, we observe that Februus is unable to locate both the triggers. This can be attributed to the suppressed gradients in case of $\epsilon = 0.4$, and thus justifies high ASR for class C_1.

VI. HaS-Net - Defending DNNs against Backdoor Attacks

In this section, we first investigate the poisonous behaviors of DNNs and develop useful insights which guide the design of *HaS-Net*, a Heal and Select training mechanism, that securely trains DNNs against backdoor attacks. Next, we analyze the impact of the number of healing epochs and the size of healing set on ASR and choose appropriate values for the rigorous evaluations in Section VII.

We observe that backdoors are learned more quickly as compared to other features of the input data as shown by high ASRs of ϵ-attack even for earlier epochs in Fig. 11. We can exploit this observation to identify potentially poisoned samples in the training data by studying the network loss for each sample. To illustrate this, we poison the first 600 (1.2%) training samples of CIFAR-10 and compare their loss with clean samples for two epochs in Fig. 11(b), for a typical case of $\epsilon = 1.0$. As evident, a rather small loss reflects faster learning, suggesting the likelihood of poisonous behaviour.

[15]https://github.com/naiyeleo/ABS

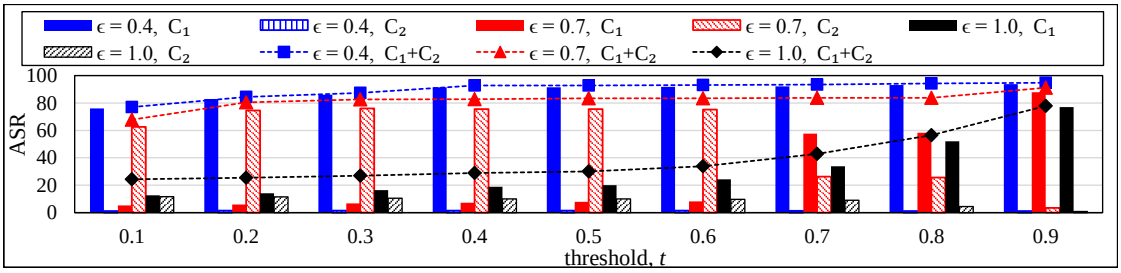

Fig. 10: ϵ^2-*Attack Success Rate against Februus for different values of* t *and* ϵ*, using* Z_2 *as trigger at inference-time.* C_1 *and* C_2 *denote ASR for target class* C_1 *and* C_2*, respectively.* $C_1 + C_2$ *denotes the sum of ASRs for target classes* C_1 *and* C_2*.*

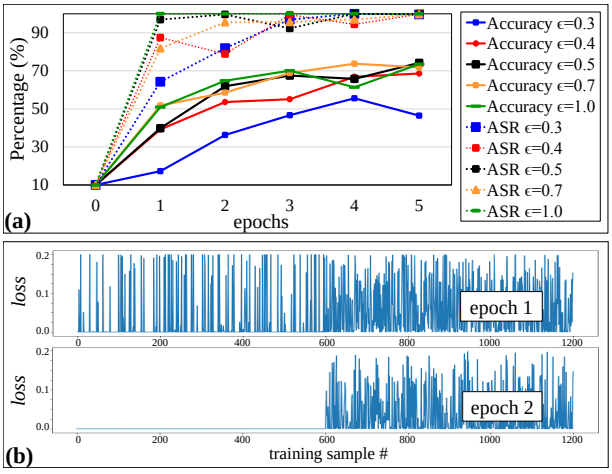

Fig. 11: *(a) A comparison of ASR vs. the test accuracy on CIFAR-10 indicating fast learning of backdoors. (b) A comparison of a network's loss on poisoned (first 600) and clean training samples (last 600) for the first two epochs.*

A. The Trust-index

Here, we introduce the notion of a *trust-index* for each instance of the training set, which represents the consistency of the instance with a given learning task. We quantify the *trust-index* based on a set of clean samples (healing set) trusted by the defender and assumed to be correctly labeled. We reason that since poisoned training samples force a DNN to learn inconsistent features, they must exhibit low *trust-indices*.

We assume a clean healing data, $\{D_H = (X_H, Y_H)\} \in \mathcal{R}$, available to our defender, which is correctly labeled, and therefore has high *trust-index*. The healing data is assumed to be considerably smaller than, and distinct from the training data, $\{D_T = (X_T, Y_T)\} \in \mathcal{R}$, where, X represents the inputs to be mapped by a network, $\mathcal{F}(X, \theta)$, to the output, Y, where θ denotes the parameters of the network. If D_T is consistent with the task, training $\mathcal{F}$ on D_H should reduce its error on every instance of D_T[16]. Formally, given a loss function, $\mathcal{L}(\mathcal{F}(X, \theta), Y)$, $\forall (X_k, Y_k) \in (X_T, Y_T)$,

$$sign\left(\Delta \sum_{\forall (X_i, Y_i) \in (X_H, Y_H)} [\mathcal{L}(\mathcal{F}(X_i, \theta), Y_i)]\right) \approx sign(\Delta \mathcal{L}(\mathcal{F}(X_k, \theta), Y_k)) \quad (2)$$

[16]This is because the network's D_H loss serves as a proxy for D_T loss.

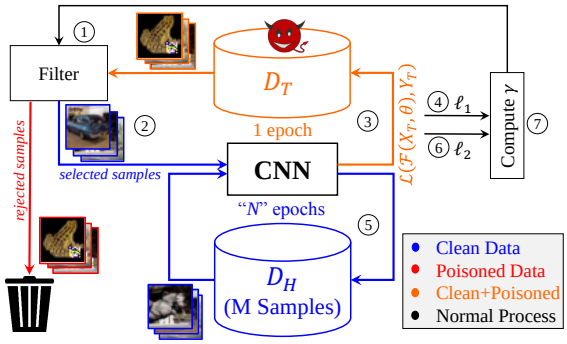

Fig. 12: *Novel HaS-Nets training methodology. Numbers in circle represent the sequence of operations.*

where Δ represents the change. As that the loss of $\mathcal{F}$ on some data, D, reduces when trained, eq (2) becomes

$$-1 \approx sign(\Delta \mathcal{L}(\mathcal{F}(X_k, \theta), Y_k)) \quad (3)$$

an approximate value for the right hand side of eq (2) can be computed by monitoring the loss (l_1 and l_2) of $\mathcal{F}$ for each sample in D_t, before and after healing, respectively. Thus,

$$-1 = sign(l_2 - l_1) \quad (4)$$

$$\implies -\gamma = (l_2 - l_1) < 0 \quad (5)$$

where γ denotes the *trust-index*. Training instances satisfying eq (5) are assumed to have a high γ. Eq (5) defines a metric to evaluate the quality of a training sample for a given task, i.e., clean inputs show $\gamma < 0$, and otherwise for the poisoned samples.

B. HaS-Net - A Heal and Select Mechanism to Defend DNNs

In this sub-section, we present HaS-Net, a novel methodology to resist backdoors during training, following the intuitions developed in the previous analysis. HaS-Net iteratively heals a network while training, identifies potentially poisoned training samples using γ, and removes them from training in the subsequent iteration. Healing tends to remove the backdoors, thus increasing a network's loss on poisoned samples.

Methodology: Fig. 12 depicts an overview of the inner-workings of HaS-Net training. Each iteration comprises three stages: training, healing and selection. In the training stage, the DNN is trained for one epoch on poisoned training data,

Algorithm 1 HaS-Net Training Algorithm

Input:
$\{D_T = (X_T, Y_T)\} \leftarrow$ Poisoned Training Data
$\{D_H = (X_H, Y_H)\} \leftarrow$ Healing Data
$\mathcal{F} \leftarrow$ DNN architecture
$\theta \leftarrow$ Initial parameters
$N \leftarrow$ No. of healing epochs
$i_{max} \leftarrow$ Maximum iterations
Output:
$\mathcal{F} \leftarrow$ Trained DNN
Define $i \leftarrow 0$, $\tau \leftarrow 10^{-8}$
Define $\mathcal{L}(A, B) \leftarrow (a - b)^2$
Define $D_S \leftarrow D_T$
Define $\gamma \leftarrow 0$ for all samples of D_T
repeat
  Train $\mathcal{F}$ for one epoch on D_S
  $l_1 \leftarrow \mathcal{L}(\mathcal{F}(X_T, \theta), Y_T)$ for all samples of D_T
  Train $\mathcal{F}$ for N epoch on D_H
  $l_2 \leftarrow \mathcal{L}(\mathcal{F}(X_T, \theta), Y_T)$ for all samples of D_T
  $-\gamma \leftarrow l_2 - l_1$
  $D_S \leftarrow$ samples of D_T with $-\gamma < 0$ and $l_1 > \tau$
  $i \leftarrow i + 1$
until $i \leq i_{max}$

D_T, and the loss l_1 is computed for each training sample. We remove the samples with $l_1 < \tau$, as too small of a loss may represent a poisonous behavior[17]. In the healing stage, the DNN is trained on healing data, D_H, for N epochs (where N is the hyperparameter) and the loss, l_2, is computed for each training data sample. During selection, HaS-Net chooses eligible samples for training in the subsequent iteration based on their *trust-index* (γ) defined in eq (5). Samples with $\gamma < 0$ are not selected for the next training step. A step-wise description of HaS-Net is given in Algorithm 1.

C. Tuning HaS-Net

Here, we evaluate HaS-Net for different hyper-parameter values, and choose the best setting for further analysis. To prove that our findings are scalable, we tune these hyperparameters on Fashion-MNIST and use these values when evaluating HaS-Net on other datasets (CIFAR-10, Urban Sound, Consumer Complaint) in Section VII.

Number of healing epochs - N: We poison 600 training samples of Fashion-MNIST dataset with Trigger 1 in Fig. 2 using ϵ-attack. With a healing set of 15% the full training set size, we train HaS-Net for one iteration, and plot the distribution of $-\gamma$ ($= l_2 - l_1$) after one iteration of both the clean and poisoned samples in Fig. 13(a) for different number of healing epochs, N. We observe that γ can successfully capture poisoned inputs, specifically for $N \geq 5$. This is because a larger N removes the backdoor more effectively, thus, increasing a network's loss on poisoned samples. For future evaluations, we use $N = 10$ as it effectively captures poisoned samples while not causing large computational overheads due to a significantly small size of our healing set.

The Size of Healing Set - M: Intuitively, the *size of the healing set*, M, can significantly affect the performance of HaS-Net on the clean test data. The Fashion-MNIST dataset, being easier to learn, may not capture the broader impact of M on the accuracy and ASR. We therefore analyze M for both Fashion-MNIST and CIFAR-10.

[17]We typically use $\tau = 10^{-8}$. We experiment with different values of τ, and observe no effect on ASR for $\tau < 10^{-2}$.

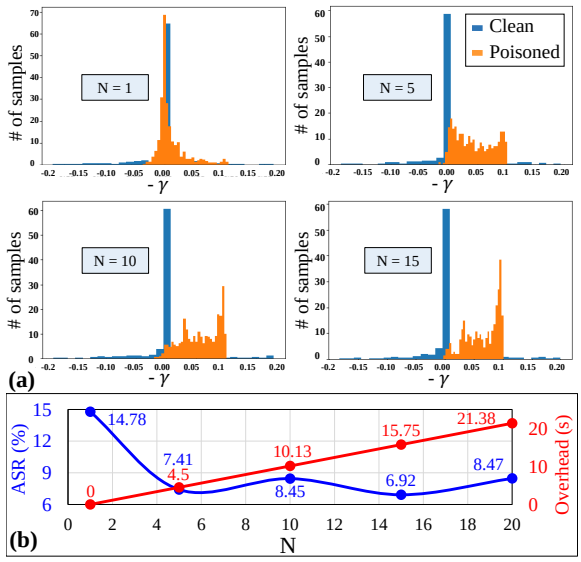

Fig. 13: *(a) γ distribution of the clean training samples vs. the poisoned training samples for Fashion-MNIST. Note that poisoned samples show negative trust-index, γ. (b) Impact of the healing epochs, N, on the ASR and overhead (in seconds) required.*

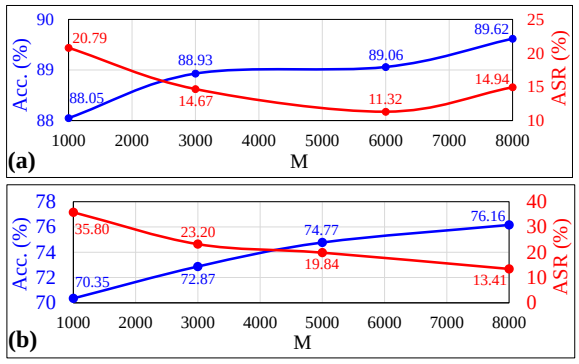

Fig. 14: *Impact of the healing set size, M, on the Test accuracy and ASR of ϵ-attack on HaS-Nets.*

Fig. 14(a) and Fig. 14(b) record the impact of M on the performance of HaS-Net for Fashion-MNIST and CIFAR-10, respectively. Increasing M, increases the network performance on test data and reduces the ASR. For Fashion-MNIST, increasing M from 1000 to 8000, decreases ASR from 20% to 15%, and increases the accuracy from 88% to 89.8%. For CIFAR-10, a similar increase in M causes the ASR to decrease from 35% to 12% and the test accuracy increases from 70% to 76%. This is because a smaller healing set may not sufficiently represent a real distribution causing unfair selection/rejection of the training samples. For further experiments we choose $M \leq 15\%$ of the full training data to ensure the effectiveness of HaS-Net for practical scenarios.

VII. Evaluations

In this section, we evaluate HaS-Net against different backdoor attacks for Fashion-MNIST, CIFAR-10, mini-Consumer Complaint and Urban Sound datasets. Our experiments suggest

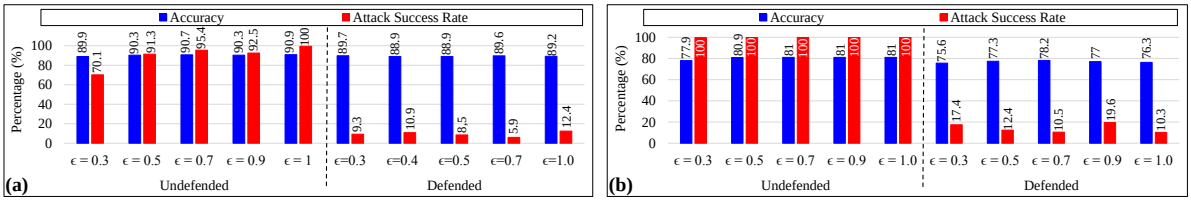

Fig. 15: *Test Accuracy of HaS-Nets on clean data and ϵ-Attack Success Rates (ASR) against HaS-Nets in comparison with an undefended model for (a) Fashion-MNIST and (b) CIFAR-10.*

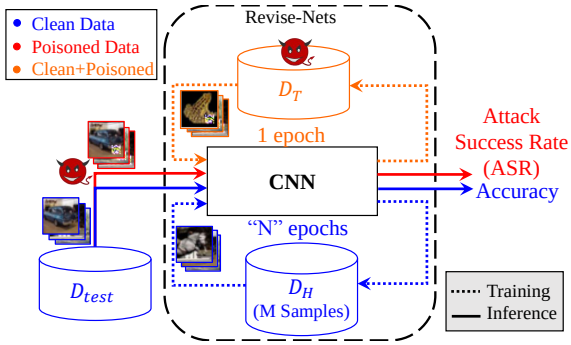

Fig. 16: *Experimental Setup for evaluating HaS-Nets. Figure also illustrates our Threat Model, where an adversary can poison training data, D_T, and inference-time inputs.*

TABLE I: Typical settings for evaluating *HaS-Net*

Dataset	Type	Classes	Training Data	Test Data	Poisoned Samples	Heal Data	# of epochs
Fashion-MNIST	Image	10	60000	10000	600	13%	10
CIFAR-10	Image	10	50000	10000	600	16%	20
IMDB	Text	2	25000	25000	600	8%	10
Consumer Complaint	Text	11	50104	16702	600	9%	5
Urban Sound	Audio	10	6549	2183	80	8%	20

that HaS-Net can effectively resist backdoor insertion under several attack settings, irrespective of the dataset and the network architecture.

A. Experimental Setup

The experimental setup is shown in Fig. 16. We evaluate HaS-Net for different datasets, choosing each for its wide use in the literature. The typical settings used are given in Table I.

Datasets and Architectures: For evaluation on vision tasks, we train 2D-CNNs on Fashion-MNIST and CIFAR-10 datasets. Both datasets contain 10 classes, where Fashion-MNIST is a 28x28 grey-scale image dataset, while CIFAR-10 contains color images 32x32 in size. For text and audio applications, we use an MLP model for IMDB, a 1D-CNN for mini-Consumer Complaint[18] and a 2D-CNN for Urban Sound[19] dataset. A more detailed description of datasets and respective architectures is provided in the supplementary files.

B. Evaluation on ϵ-Attack

Fashion-MNIST: We evaluate the robustness of HaS-Net by performing ϵ-Attack on Fashion-MNIST. Results in Fig. 15(a) show a slight decrease in accuracy of the HaS-Net as compared to an undefended model, which may be the cost of increased robustness. This can be explained with Fig. 13(a), where few clean samples show very small γ values, and thus are not selected for training in the next iteration.

CIFAR-10: Fig. 15(b) shows results for CIFAR-10. We observe a significant decrease in ASR for different values of ϵ. Here again, we observe a slight reduction in the accuracy of HaS-Net clean test data.

Consumer Complaint: Results of our evaluation on mini-Consumer Complaint dataset are shown in Fig. 19(a). As previously, we observe low ASRs and a slight reduction in the accuracy of HaS-Net, indicating the scalability of HaS-Net to different datasets and architectures, even at the behavioral level.

Urban Sound: Results of our evaluation on Urban Sound dataset are shown in Fig. 19(b). As previously, we observe a significant improvement in robustness and a slight degradation in test accuracy.

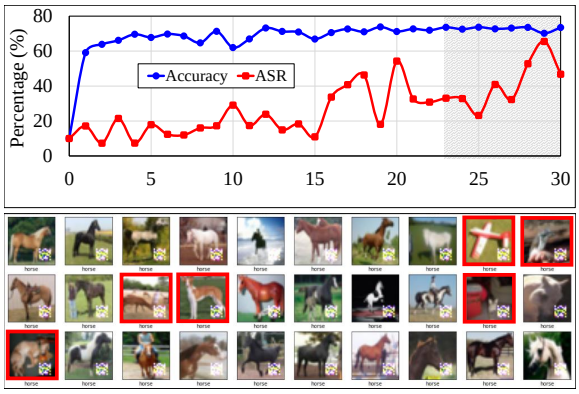

Fig. 17: *All-trojan attack on HaS-Nets (target class being "horse") for CIFAR-10. **Top:** Test accuracy and ASR against HaS-Nets for $\epsilon = 1.0$. **Bottom:** First 50 training samples to have passed the selection criteria of HaS-Nets after 30 iterations. Images which are not horses are marked with a red-box around them.*

C. Evaluation on All-Trojan Attack

We study HaS-Net under a more extreme setting—when the poisoned training set is significantly larger than the healing set. More specifically, we poison all the samples in the training set of CIFAR-10 and Fashion-MNIST datasets—we name this an all-trojan attack. Our target class is "horse" for CIFAR-10 and "sneaker" for Fashion-MNIST. We use $\epsilon = 1.0$. Results of HaS-Net training for a few iteration are shown in Fig. 17 and Fig. 18. Surprisingly, *HaS-Net can partially resist such a large-scale attack*. This is because HaS-Net removes all the inconsistent training samples which poison a DNN. This is observable

[18]https://www.kaggle.com/cfpb/us-consumer-finance-complaints.
[19]https://www.kaggle.com/chrisfilo/urbansound8k.

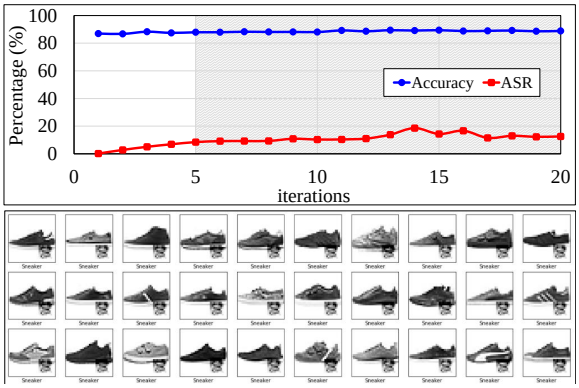

Fig. 18: *All-trojan attack on HaS-Nets (target class being "sneakers") for Fashion-MNIST.* ***Top:*** *Test accuracy and ASR against HaS-Nets for* $\epsilon = 1.0$. ***Bottom:*** *First 50 training samples to have passed the selection criteria of HaS-Nets after 20 iterations.*

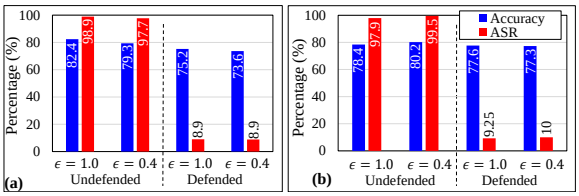

Fig. 19: *Test accuracy of HaS-Nets and* ϵ*-Attack ASR against HaS-Nets for (a) Consumer Complaint and (b) Urban Sound dataset.*

in Fig. 17 and Fig. 18, which show first 50 training smaples for CIFAR-10 and Fashion-MNIST, respectively. Interestingly, though unsurprisingly, most of the samples which survive HaS-Net filtering are horses and sneakers—belonging to the target class to which we initially labelled all training samples. HaS-Net has not selected the incorrectly labelled samples due to their small γ. When we input the selected samples to a clean model with no backdoor, 91.899% of CIFAR-10 selected samples were classified as "horse", and 99.9% of the Fashion-MNIST selected samples were labelled "sneakers".

D. Evaluation on Invisible ϵ*-Attack*

Invisible backdoor attacks use triggers which span the entire image, and are specially effective against statistical defenses. Such attacks use a small magnitude noise, instead of a visible patch, as the trigger. Consequently, a human observer cannot distinguish between a clean and a poisoned input.

We expose HaS-Net to invisible-backdoor attack, and compare its test accuracy and ASR with an undefended model in Fig. 20(a) and (b) for Fashion-MNIST and CIFAR-10, respectively. A reduction in the ASR from above 90% for an undefended model to below 12% for HaS-Net indicates that HaS-Net can effectively counter the invisible backdoor attack.

E. Evaluation on Label-Consistent Attack

Label-consistent backdoor attacks [39] exploit a GAN to poison an image without changing its label by interpolating latent representations of target class images to the latent representations of other classes—causing highly inconsistent latent representations, but appearing benign to human observers. The

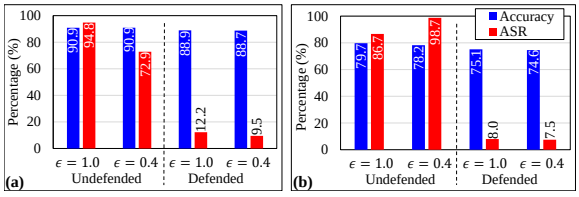

Fig. 20: *Test accuracy of HaS-Nets and invisible* ϵ*-Attack ASR against HaS-Nets for (a) Fashion-MNIST and (b) CIFAR-10 datasets.*

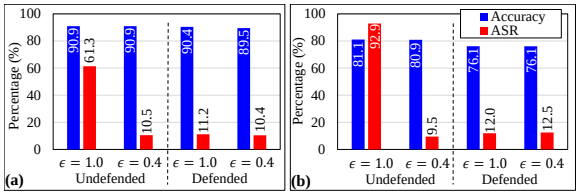

Fig. 21: *Test accuracy of HaS-Nets and label-Consistent* ϵ*-Attack ASR against HaS-Nets for (a) Fashion-MNIST and (b) CIFAR-10 datasets.*

TABLE II: Comparing HaS-Net with a number of state-of-the-art training-time defenses for Fashion-MNIST dataset

Defenses	ϵ-Attackϵ-Attack						Invisible ϵ-AttackInvisible ϵ-Attack					
	ϵ=0.4ϵ=0.4		ϵ=0.7ϵ=0.7		ϵ=1.0ϵ=1.0		ϵ=0.4ϵ=0.4		ϵ=0.7ϵ=0.7		ϵ=1.0ϵ=1.0	
	Acc.	ASR	Acc.	ASR	Acc.	ASR	Acc.	ASR	Acc.	ASR	Acc.	ASR
Undefended	90.92	83.63	90.92	89.5	90.92	100	90.92	74.22	90.9	94.6	90.9	98.23
GS (0.001)	73.97	94.94	73.68	99.19	72.85	99.69	73.58	64.98	73.5	94.63	74.24	97.76
GS (0.1)	62.84	65.20	62.75	94.27	62.65	99.05	62.21	20.94	63.08	50.82	62.12	66.91
GS (1)	42.25	58.96	43.32	87.33	43.38	95.46	44.78	23.77	43.55	23.75	43.56	33.39
Thresholding (0.5)	70.09	25.97	72.42	77.45	67.33	100	71.22	25.51	72.65	77.55	67.28	100
Thresholding (0.6)	61.82	13.63	66.92	67.72	61.68	100	60.84	13.61	66.64	67.96	61.24	100
Thresholding (0.7)	53.87	6.49	61.26	55.9	55.81	99.99	53.93	6.15	61.63	55.65	55.17	99.99
Thresholding (0.8)	46.05	3.02	55.2	40.66	49.79	99.99	46.35	3.61	55.15	40.84	49.19	99.99
HaS-Nets	89.42	10.92	89.64	5.97	89.21	12.43	88.73	9.49	88.92	11.02	88.96	12.24

interpolated images, stamped with a trigger are harder for a DNN to learn, so it learns the trigger instead.

Fig. 21(a) and (b) show results for our evaluation against Label-Consistent attack for Fashion-MNIST and CIFAR-10, respectively by poisoning 25% of training samples. For $\epsilon = 1.0$, HaS-Net can successfully resist the backdoor insertion reducing ASR from 61% and 93% to 11% and 12%, for Fashion-MNIST and CIFAR-10, respectively. For $\epsilon = 0.4$, the attacker is unable to insert a backdoor in the network for both datasets, indicating that GAN-based label consistent backdoor attacks are relatively weaker than conventional targeted backdoor attacks—a cost of the increased inconspicuousness of the attack [39].

F. Comparison with other training-time defenses

We summarize the comparison of HaS-Net with Gradient-Shaping in Table II over several attack configurations for Fashion-MNIST dataset. To the best of our knowledge, gradient shaping is the only known training-time defense against the backdoor attacks. We also experiment with the confidence threshold as a defense against low-confidence backdoor attacks—a model poisoned probabilistic labels may output low confident decisions, which can be detected by a simple threshold mechanism; however, our results oppose this hypothesis.

Gradient-shaping can partially reduce the ASR irrespective of the attack type. However, this comes at the cost of reduced accuracy over clean inputs. On the contrary, HaS-Net provides better robustness without a large drop in accuracy. For the

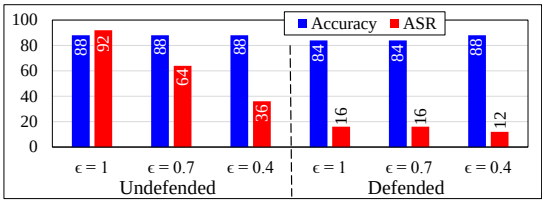

Fig. 22: *Comparison of vanilla training with HaS-Nets over test accuracy and invisible ϵ-Attack ASR kaggle celebrity face datasets.*

threshold-based defense, the ASR partially drops but at the cost of considerable decrease in the accuracy over clean inputs. HaS-Net shows clear superiority over other approaches.

G. Discussion

Evaluation on Celebrity face dataset

For completion, we extend our analysis to an openly available celebrity face dataset, containing 118 images of 5 celebrities divided into 93 training and 25 test images. We re-scale each image to 200x200x3. For celebrity dataset, an input image is considerable larger as compared to that for Fashion-MNIST and CIFAR-10 datasets. We therefore use a more complex architecture—VGG-16—for classification. We were unable to poison our model by poisoning only 3% of the training samples. However, increasing the poisoning ratio to 30% successfully poisons the model. Thus, we validate the observation first made by Chen *et al.* [8] that poisoning a model for face datasets requires approximately $10times$ more poisoning.

We randomly choose a set of 40 unperturbed images from the training set to serve as the healing set. Fig. 22 reports the accuracy and ASR of our model with and without the defense incorporated. Reduced ASRs in Fig. 22 validate that HaS-Net is scalable to different model complexities and input sizes.

Catastrophic forgetting and healing mechanisms

Li *et al.* [25] note that the the catastrophic forgetting of neural networks is what makes healing mechanisms effective against backdoor attacks. A DNN trained on some task, T_A, forgets T_A when trained on a different task, T_B. As the healing set is not poisoned, a DNN forgets the backdoor when trained on healing set. However, unlike the previous works, we are the first to repeatedly heal a DNN, and identify potential poisoned inputs based on what DNN has forgotten.

VIII. Conclusion

In this work, we challenged the robustness of recently proposed defenses by proposing a low-confidence backdoor attack, and its two variants, ϵ-Attack and ϵ^2-Attack. Low-confidence backdoor attacks utilize low confidence labels to hide their presence from the defender. By carefully analyzing the behaviour of poisoned samples, we developed useful insights for a generic defense strategy, "*HaS-Net*", for securely training DNNs against backdoor attacks by assuming the presence of a small healing dataset available to the defender.

HaS-Net was shown to resist many variants of backdoor attacks (e.g. ϵ-attack, invisible backdoor attack, all-Trojan attack, Label-consistent attack) under diverse settings and outperform state-of-the-art defenses (i.e. Gradient-shaping, ULP-defense, Februus and STRIP). Moreover, *HaS-Net* was shown to be agnostic to dataset type and network architecture. Our work is the first to evaluate a defense on an all-trojan backdoor attack by poisoning 100% of the training data. We demonstrated the effectiveness of *HaS-Net* under such extreme settings.

References

[1] M. Abadi *et al.*, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.

[2] H. Ali, F. Khalid, H. A. Tariq, M. A. Hanif, R. Ahmed, and S. Rehman, "Sscnets: Robustifying dnns using secure selective convolutional filters," *IEEE Design & Test*, vol. 37, no. 2, pp. 58–65, 2019.

[3] H. Ali, M. S. Khan, A. AlGhadhban, M. Alazmi, A. Alzamil, K. Al-utaibi, and J. Qadir, "Analyzing the robustness of fake-news detectors under black-box adversarial attacks," *IEEE Access*, 2021.

[4] M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar, "The security of machine learning," *Mach. Learn.*, vol. 81, no. 2, pp. 121–148, 2010. [Online]. Available: https://doi.org/10.1007/s10994-010-5188-5

[5] B. Chen *et al.*, "Detecting backdoor attacks on deep neural networks by activation clustering," *arXiv preprint arXiv:1811.03728*, 2018.

[6] J. Chen, L. Zhang, H. Zheng, X. Wang, and Z. Ming, "Deeppoison: Feature transfer based stealthy poisoning attack," 2021.

[7] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," *CoRR*, vol. abs/1712.05526, 2017. [Online]. Available: http://arxiv.org/abs/1712.05526

[8] ——, "Targeted backdoor attacks on deep learning systems using data poisoning," *arXiv preprint arXiv:1712.05526*, 2017.

[9] S. Cheng, Y. Liu, S. Ma, and X. Zhang, "Deep feature space trojan attack of neural networks by controlled detoxification," *CoRR*, vol. abs/2012.11212, 2020. [Online]. Available: https://arxiv.org/abs/2012.11212

[10] E. Chou, F. Tramèr, G. Pellegrino, and D. Boneh, "Sentinet: Detecting physical attacks against deep learning systems," *CoRR*, vol. abs/1812.00292, 2018. [Online]. Available: http://arxiv.org/abs/1812.00292

[11] I. Diakonikolas, G. Kamath, D. Kane, J. Li, J. Steinhardt, and A. Stewart, "Sever: A robust meta-algorithm for stochastic optimization," in *International Conference on Machine Learning*. PMLR, 2019, pp. 1596–1606.

[12] B. G. Doan, E. Abbasnejad, and D. C. Ranasinghe, "Februus: Input purification defense against trojan attacks on deep neural network systems," in *Proceedings of the 36th Annual Computer Security Applications Conference (ACSAC)*, ser. ACSAC 2020, 2020.

[13] S. Dodge and L. Karam, "A study and comparison of human and deep learning recognition performance under visual distortions," in *2017 26th international conference on computer communication and networks (ICCCN)*. IEEE, 2017, pp. 1–7.

[14] M. Du, R. Jia, and D. Song, "Robust anomaly detection and backdoor attack detection via differential privacy," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. [Online]. Available: https://openreview.net/forum?id=SJx0q1rtvS

[15] Y. Gao *et al.*, "Backdoor attacks and countermeasures on deep learning: A comprehensive review," *arXiv preprint arXiv:2007.10760*, 2020.

[16] ——, "Design and evaluation of a multi-domain trojandetection method on deep neural networks," *IEEE Transactions on Dependable and Secure Computing*, 2021.

[17] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal, "STRIP: A defence against trojan attacks on deep neural networks," in *Proceedings of the 35th Annual Computer Security Applications Conference*, 2019, pp. 113–125.

[18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," pp. 2672–2680, 2014. [Online]. Available: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[19] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg, "Badnets: Evaluating backdooring attacks on deep neural networks," *IEEE Access*, vol. 7, pp. 47 230–47 244, 2019. [Online]. Available: https://doi.org/10.1109/ACCESS.2019.2909068

[20] S. Hong, V. Chandrasekaran, Y. Kaya, T. Dumitras, and N. Papernot, "On the effectiveness of mitigating data poisoning attacks with gradient shaping," *CoRR*, vol. abs/2002.11497, 2020. [Online]. Available: https://arxiv.org/abs/2002.11497

[21] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," in *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 19–35.

[22] F. Khalid, H. Ali, M. A. Hanif, S. Rehman, R. Ahmed, and M. Shafique, "Fadec: A fast decision-based attack for adversarial machine learning," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.

[23] S. Kolouri, A. Saha, H. Pirsiavash, and H. Hoffmann, "Universal litmus patterns: Revealing backdoor attacks in cnns," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*. IEEE, 2020, pp. 298–307. [Online]. Available: https://doi.org/10.1109/CVPR42600.2020.00038

[24] S. Li, S. Ma, M. Xue, and B. Z. H. Zhao, "Deep learning backdoors," *CoRR*, vol. abs/2007.08273, 2020. [Online]. Available: https://arxiv.org/abs/2007.08273

[25] Y. Li, B. Wu, Y. Jiang, Z. Li, and S. Xia, "Backdoor learning: A survey," *CoRR*, vol. abs/2007.08745, 2020. [Online]. Available: https://arxiv.org/abs/2007.08745

[26] Y. Li, T. Zhai, B. Wu, Y. Jiang, Z. Li, and S. Xia, "Rethinking the trigger of backdoor attack," *CoRR*, vol. abs/2004.04692, 2020. [Online]. Available: https://arxiv.org/abs/2004.04692

[27] J. Lin, L. Xu, Y. Liu, and X. Zhang, "Composite backdoor attack for deep neural network by mixing existing benign features," in *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, J. Ligatti, X. Ou, J. Katz, and G. Vigna, Eds. ACM, 2020, pp. 113–131. [Online]. Available: https://doi.org/10.1145/3372297.3423362

[28] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," in *International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, 2018, pp. 273–294.

[29] Y. Liu, W.-C. Lee, G. Tao, S. Ma, Y. Aafer, and X. Zhang, "Abs: Scanning neural networks for back-doors by artificial brain stimulation," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 1265–1282.

[30] Y. Liu, Y. Xie, and A. Srivastava, "Neural trojans," in *2017 IEEE International Conference on Computer Design, ICCD 2017, Boston, MA, USA, November 5-8, 2017*. IEEE Computer Society, 2017, pp. 45–48. [Online]. Available: https://doi.org/10.1109/ICCD.2017.16

[31] S. Minaee, A. Abdolrashidi, H. Su, M. Bennamoun, and D. Zhang, "Biometric recognition using deep learning: A survey," *CoRR*, vol. abs/1912.00271, 2019. [Online]. Available: http://arxiv.org/abs/1912.00271

[32] S. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 2017, pp. 86–94. [Online]. Available: https://doi.org/10.1109/CVPR.2017.17

[33] N. Petrick, S. Akbar, K. H. Cha, S. Nofech-Mozes, B. Sahiner, M. A. Gavrielides, J. Kalpathy-Cramer, K. Drukker, A. L. Martel *et al.*, "Spie-aapm-nci breastpathq challenge: an image analysis challenge for quantitative tumor cellularity assessment in breast cancer histology images following neoadjuvant treatment," *Journal of Medical Imaging*, vol. 8, no. 3, p. 034501, 2021.

[34] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 336–359, 2020. [Online]. Available: https://doi.org/10.1007/s11263-019-01228-7

[35] D. Tang, X. Wang, H. Tang, and K. Zhang, "Demon in the variant: Statistical analysis of dnns for robust backdoor contamination detection," *CoRR*, vol. abs/1908.00686, 2019. [Online]. Available: http://arxiv.org/abs/1908.00686

[36] R. Tang, M. Du, N. Liu, F. Yang, and X. Hu, "An embarrassingly simple approach for trojan attack in deep neural networks," 2020.

[37] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," *arXiv preprint arXiv:1811.00636*, 2018.

[38] L. Truong, C. Jones, B. Hutchinson, A. August, B. Praggastis, R. Jasper, N. Nichols, and A. Tuor, "Systematic evaluation of backdoor data poisoning attacks on image classifiers," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR Workshops 2020, Seattle, WA, USA, June 14-19, 2020*. IEEE, 2020, pp. 3422–3431. [Online]. Available: https://doi.org/10.1109/CVPRW50498.2020.00402

[39] A. Turner, D. Tsipras, and A. Madry, "Label-consistent backdoor attacks," *CoRR*, vol. abs/1912.02771, 2019. [Online]. Available: http://arxiv.org/abs/1912.02771

[40] M. Villarreal-Vasquez and B. K. Bhargava, "Confoc: Content-focus protection against trojan attacks on neural networks," *CoRR*, vol. abs/2007.00711, 2020. [Online]. Available: https://arxiv.org/abs/2007.00711

[41] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*. IEEE, 2019, pp. 707–723. [Online]. Available: https://doi.org/10.1109/SP.2019.00031

[42] Q. Wang, W. Guo, K. Zhang, A. G. O. II, X. Xing, X. Liu, and C. L. Giles, "Adversary resistant deep neural networks with an application to malware detection," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*. ACM, 2017, pp. 1145–1153. [Online]. Available: https://doi.org/10.1145/3097983.3098158

[43] Y. Yao, H. Li, H. Zheng, and B. Y. Zhao, "Latent backdoor attacks on deep neural networks," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*, L. Cavallaro, J. Kinder, X. Wang, and J. Katz, Eds. ACM, 2019, pp. 2041–2055. [Online]. Available: https://doi.org/10.1145/3319535.3354209

[44] Y. Zeng, H. Qiu, S. Guo, T. Zhang, M. Qiu, and B. M. Thuraisingham, "Deepsweep: An evaluation framework for mitigating DNN backdoor attacks using data augmentation," *CoRR*, vol. abs/2012.07006, 2020. [Online]. Available: https://arxiv.org/abs/2012.07006

[45] H. Zhong, C. Liao, A. C. Squicciarini, S. Zhu, and D. J. Miller, "Backdoor embedding in convolutional neural network models via invisible perturbation," in *CODASPY '20: Tenth ACM Conference on Data and Application Security and Privacy, New Orleans, LA, USA, March 16-18, 2020*, V. Roussev, B. M. Thuraisingham, B. Carminati, and M. Kantarcioglu, Eds. ACM, 2020, pp. 97–108. [Online]. Available: https://doi.org/10.1145/3374664.3375751