The ISC Int'l Journal of Information Security
ISeCure

October 2025, Volume 17, Number 3 (pp. 1–12)
http://www.isecure-journal.org

PRESENTED AT THE ISCISC'2025 IN TEHRAN, IRAN.

Backdoor Defense via Aggregation of Outsourced Models using Multi-Stage Knowledge Distillation **

Amirhossein Heydari [1], Azadeh Mansouri [1,*], and Ahmad Mahmoudi-Aznaveh [2]

[1] *Department of Electrical and Computer Engineering, Faculty of Engineering, Kharazmi University, Tehran, Iran*
[2] *Cyberspace Research Institute, Shahid Beheshti University, Tehran, Iran*

ARTICLE INFO.

Keywords:
Backdoor Mitigation, Black-box Model Defense, Knowledge Distillation, Model Aggregation Strategy, Trigger Suppression, Untrusted Model Scenario

Type:

doi:

ABSTRACT

Backdoor attacks pose a significant threat to deep learning systems by injecting hidden malicious behavior to the model while preserving high accuracy on clean data. Such attacks are particularly dangerous in scenarios where users rely on pre-trained models or outsource training to untrusted parties. In this work, we propose a practical defense strategy that assumes no knowledge of the backdoor trigger or the training process, relying on a small trusted clean dataset. Our method introduces a two-stage pipeline: First, we aggregate predictions from multiple potentially compromised models to train an intermediate Teacher-Aggregation (TA) model; then, we distill this knowledge into a compact light-weight student model. This multi-stage approach effectively alleviates backdoor effects while preserving clean accuracy. Experimental results on MNIST and CIFAR-10 demonstrate that our method significantly reduces the Attack Success Rate (ASR)—to approximately 0.1% on MNIST and 2.6% on CIFAR-10—outperforming baseline ensemble defenses. Furthermore, our lightweight student model is suitable for edge deployment, providing a generic and scalable defense that remains robust under minimal assumptions, making it well-suited for real-world applications in adversarial environments. Our code is available at: `https://github.com/mr-pylin/backdoor-toolbox`

1 Introduction

With ith the widespread adoption of deep neural networks (DNNs) in mission-critical applications, concerns about their security have received increasing attention. Backdoor learning has emerged as a particularly serious threat, in which adversaries embed triggers—often imperceptible, but sometimes clearly visible—into a model by maliciously manipulating training data or gaining control over the training process. This threat poses a significant risk to the widely practiced approach of downloading unverified datasets or pre-trained models from external sources, as well as outsourcing the training process to third-party platforms.

In many real-world scenarios, users rely on pre-

[*] Corresponding author.
**The ISCISC'2025 program committee effort is highly acknowledged for reviewing this paper.

Email addresses: `amirhossein.heydari@khu.ac.ir`, `a_mansouri@khu.ac.ir`, `a_mahmoudi@sbu.ac.ir`

ISSN: 2008-2045 © 2025 ISC. All rights reserved.

trained models or externally sourced datasets due to hardware limitations, time constraints, or the high cost of data curation. However, both approaches pose serious security risks: pre-trained models may be compromised through backdoor attacks, and public datasets can contain poisoned samples designed to trigger malicious behavior. Outsourcing training or distributing it across multiple untrusted environments—common in large-scale or collaborative settings—further increases the attack surface. In such cases, limited visibility into the training pipeline makes it difficult to detect backdoor insertion, and adversaries can exploit fragmented data partitions to inject targeted triggers, leading to model contamination.

What makes backdoor attacks particularly insidious is that poisoned models typically maintain high accuracy on clean data, concealing their malicious behavior unless triggered under specific conditions. This renders traditional validation methods ineffective and complicates efforts to detect or mitigate such threats. As recent studies have demonstrated, even widely used model-sharing platforms can unknowingly host and distribute backdoored models, underscoring the growing need for post-hoc defense mechanisms.

We propose a lightweight defense strategy that assumes no knowledge of the backdoor trigger and requires only a small trusted clean subset. Our method leverages model aggregation and knowledge distillation to reconstruct a robust, compact model from a set of suspicious models, even when each may be partially compromised. By operating with minimal assumptions about the poisoning mechanism, our approach remains practical in real-world deployment scenarios.

As illustrated in Figure 1: we propose a scenario that is structured into four distinct stages.

- **Stage 1 – Baseline Clean Model**
- **Stage 2 – Simulated Poisoned Models**
- **Stage 3 – Baseline Defense Evaluation**
- **Stage 4 – Proposed Defense**, which comprises two components:
    - **Stage 4.1 – Aggregation**
    - **Stage 4.2 – Knowledge Distillation**

Stage 1 involves training a clean baseline model to serve as a reference for both performance and behavioral analysis. In Stage 2, we simulate poisoned models by introducing backdoor triggers through data poisoning, representing potential real-world attack scenarios. Stage 3 applies baseline ensemble-based defenses—specifically hard and soft voting—on the poisoned models generated in Stage 2, serving as a reference point to evaluate the effectiveness of our proposed defense.

Finally, Stage 4 introduces our proposed defense mechanism, composed of two components: Stage 4.1 – Aggregation, where parameters from multiple neural networks are combined to dilute the influence of backdoored models and improve robustness; and Stage 4.2 – Knowledge Distillation, where the ensemble's knowledge is transferred to a lightweight student model, effectively eliminating backdoor behaviors while retaining generalization. This final stage not only enhances security but also yields a more compact and deployment-friendly model.

To demonstrate the effectiveness of our method, we evaluate it on standard benchmarks including MNIST and CIFAR-10 under multiple backdoor settings. Our aggregation and distillation approach reduces the Attack Success Rate (ASR) from 17.2% (soft vote) and 5.6% (hard vote) to as low as 0.1% on MNIST, while maintaining high Clean Data Accuracy (CDA) around 98.9%. On the more challenging CIFAR-10 benchmark, we still achieve a competitive ASR reduction—down to 2.6%. These results highlight the robustness of our method, particularly in untrusted settings, and its ability to recover a secure and lightweight model.

The remainder of this paper is organized as follows. Section 2 reviews related work on backdoor attacks and defenses. Section 3 introduces our proposed defense method in detail. Section 4 outlines the experimental setup, including datasets, attack settings, and evaluation metrics. Section 5 presents and analyzes the experimental results. Finally, Section 6 concludes the paper.

2 Related Work

Early work revealed that a model trained on mostly clean data could be manipulated to misclassify inputs embedded with a specific **trigger** into an attacker-chosen **target class**. For example, **label-flipping attack** [1] introduced targeted poisoning attacks that flip a few labels and achieve more than90% attack success rate with only ~ 50 poisoned samples. Around the same time, **BadNets** [2] showed that an outsourced model could behave normally on clean data while misclassifying any input containing a special trigger.

Since the early attacks, backdoors have become increasingly stealthy and diverse. **Clean-label attacks**[3] preserve correct labels while injecting imperceptible triggers, and other works have explored stealth through novel trigger designs such as steganographic[4], natural reflection-based [5], and frequency-domain perturbations [6]. More advanced techniques include **conditional backdoors**[7] that activate under specific transformations (e.g., JPEG compression), and **switchable backdoors**[8] can be turned on/off via an extra prompt, making them extremely evasive. Attacks have also targeted **transfer learning**[9], showing that poisoned behavior can propagate during fine-tuning, and exploited **quantization artifacts** in model compression: **Qu-antization**[10] demonstrated triggers that remain dormant in full precision but activate post-quantization.

In response, defense techniques have evolved in waves. Early defenses relied on **activation clustering** and **trigger synthesis**, such as **Neural Cleanse** [11] and **ABS** [12], but often failed under adaptive attacks. **Pruning and fine-tuning** emerged as practical alternatives: **Fine-Pruning** [13] prunes suspicious neurons, while re-training on a small clean dataset [14] can partially restore integrity. However, these methods typically degrade clean accuracy or require access to clean data. **Anti-Backdoor Learning (ABL)** [15] suppresses backdoor features via entropy maximization and trigger-invariant representation learning. **Knowledge distillation (KD)**-based defenses also gained traction: [16] distilled a clean student from a poisoned teacher, while [17] introduced **neural behavior alignment** to reduce trigger influence. Another strategy exploits the **non-transferability of triggers** across models, as shown by [18], where poisoned samples are flagged if predictions differ significantly across networks. In parallel, *Nearest is Not Dearest* [19] tackled **quantization-conditioned backdoors** using **quantization-aware training** with **nearest-neighbor smoothing**.

Yet, attackers remain adaptive. [20] evaluated several **adaptive poisoning techniques** and demonstrated that many defenses (e.g., **Neural Cleanse**, **Fine-Pruning**) can be bypassed through minimal trigger design adjustments. The **BackdoorBench benchmark** [21] further highlights this: in extensive evaluations of attacks vs. defenses, it concludes that **no single defense generalizes well**, especially under new or unseen triggers.

Amid this arms race, **ensemble- and distillation-based defenses** emerged as a promising approach. The intuition is that backdoors are often model-specific; thus, aggregating outputs from multiple independently trained models can cancel out malicious behavior. For example, **BDEL** [22] and **BDEKD** [23] showed that ensemble learning can lower the backdoor success rate, and combining it with **KD** leads to better robustness. In contrast, our method operates under a stricter and more realistic threat model where all teacher models may be untrusted and potentially backdoored. To mitigate this, we propose a **layered distillation process**: we first aggregate teacher outputs via a clean-data-trained intermediate "TA model," then distill its calibrated predictions into a final student. This two-stage pipeline helps isolate and suppress malicious signals embedded in individual teachers, resulting in a student model that is robust against even fully compromised ensembles, while maintaining high clean accuracy. Moreover, our approach yields a **single lightweight student model** at inference time, which offers improved deployment efficiency compared to maintaining a full ensemble.

In **federated learning (FL)**, backdoor threats manifest differently due to the decentralized nature of training. Several works have shown that FL is vulnerable to **poisoning attacks** [24], and in response, defenses such as **ADFL** [25] have been proposed, which use **adversarial distillation** to suppress malicious updates. However, these defenses are designed to operate during collaborative training, where client behaviors can be monitored and aggregated iteratively. In contrast, our setting assumes **static, already-trained models** acquired from untrusted sources, with no control over their training process. While FL-based defenses are not directly applicable in such a scenario, their use of **aggregation and distillation** parallels our design philosophy: both seek to filter out harmful influence by leveraging consensus and cleaner supervision pathways.

3 Proposed Method

In this work, we propose a novel defense mechanism against backdoor and data poisoning attacks in deep learning systems. We argue that a **generic** and **durable** mitigation strategy—capable of addressing both current and emerging backdoors—is achievable by accepting an initial overhead: specifically, the cost of obtaining multiple models rather than relying on a single outsourced model.

This ensemble can be formed in two ways: (1) by acquiring N *independently trained teacher models* from various providers, or (2) by outsourcing N *distinct datasets* and training separate teacher models locally. Notably, teacher models or datasets obtained from untrusted sources may themselves be poisoned, whether deliberately or inadvertently. While this approach involves additional cost and effort, such an investment is justified in **high-stakes applications** such as medical or military systems, where robustness against unknown threats is critical.

Inspired by *ensemble learning* and *knowledge distillation*, we propose a **multi-stage model aggregation framework** that introduces a dedicated **TA (Teacher-Aggregation) model** as an intermediate between the ensemble of teacher models and the final

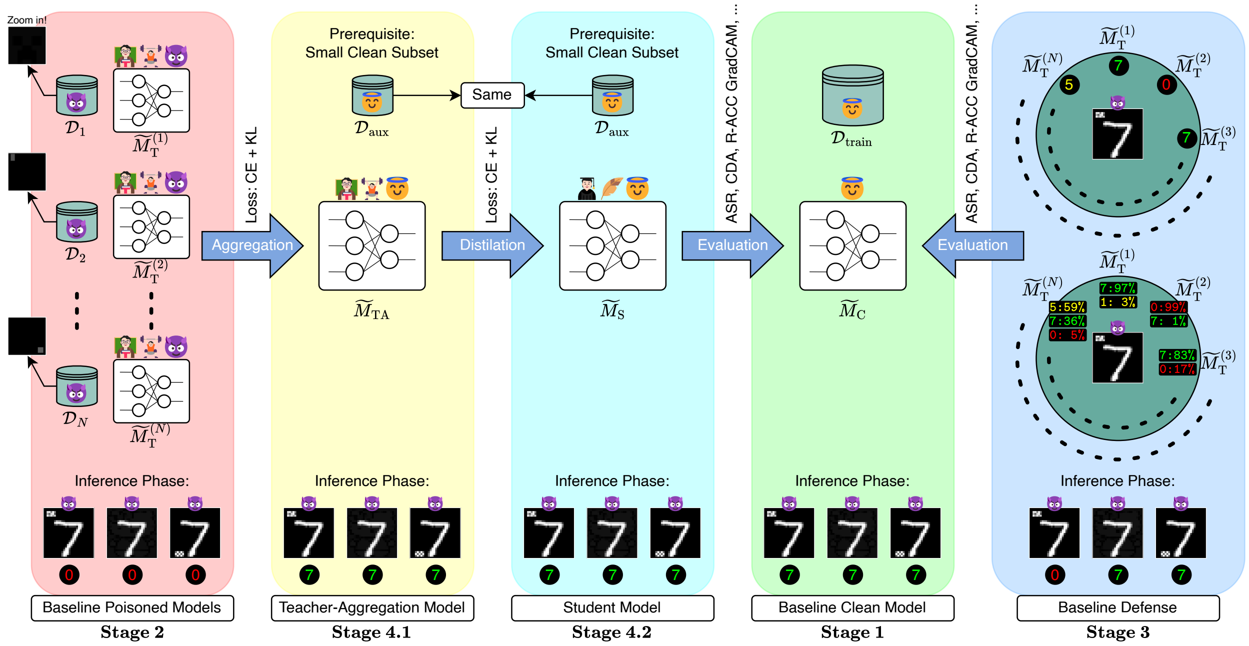
Zoom in!
$\mathcal{D}_1$
$\widetilde{M}_{\mathrm{T}}^{(1)}$
$\mathcal{D}_2$
$\widetilde{M}_{\mathrm{T}}^{(2)}$
$\mathcal{D}_N$
$\widetilde{M}_{\mathrm{T}}^{(N)}$
Loss: CE + KL
Aggregation
Prerequisite: Small Clean Subset
$\mathcal{D}_{\text{aux}}$
Same
$\widetilde{M}_{\text{TA}}$
Loss: CE + KL
Distillation
Prerequisite: Small Clean Subset
$\mathcal{D}_{\text{aux}}$
$\widetilde{M}_{\text{S}}$
ASR, CDA, R-ACC GradCAM, ...
Evaluation
$\mathcal{D}_{\text{train}}$
$\widetilde{M}_{\text{C}}$
ASR, CDA, R-ACC GradCAM, ...
Evaluation
$\widetilde{M}_{\mathrm{T}}^{(1)}$
$\widetilde{M}_{\mathrm{T}}^{(2)}$
$\widetilde{M}_{\mathrm{T}}^{(3)}$
$\widetilde{M}_{\mathrm{T}}^{(N)}$
7
0
7
5
7:97%
1: 3%
0:99%
7: 1%
7:83%
0:17%
5:59%
7:36%
0: 5%
Inference Phase:
0 0 0
Inference Phase:
7 7 7
Inference Phase:
7 7 7
Inference Phase:
7 7 7
Inference Phase:
0 7 7
Baseline Poisoned Models
Stage 2
Teacher-Aggregation Model
Stage 4.1
Student Model
Stage 4.2
Baseline Clean Model
Stage 1
Baseline Defense
Stage 3

Figure 1. Overview of the backdoor attack/defense pipeline. Icon legends: 😇 clean dataset/model, 😈 poisoned dataset/model, 🧑‍🏫 teacher model, 🧑‍🎓 student model, 🏋️ large model, 🪶 lightweight model. Label colors in image predictions: Red (0) indicates the attacker-specified target class (backdoor success), Green (7) represents the correct label even when the image is poisoned (robust prediction), Yellow (others) indicates incorrect predictions that are not the attacker's target.

light-weight student model. The stages of the aggregation and distillation process are shown in Figure 1:

(1) **Stage 4.1 (Aggregation):** We aggregate the logits from the N teacher models by averaging them, and then train the TA model on a small clean dataset, using both cross-entropy with respect to the true labels and KL divergence with respect to the aggregated logits, as defined in Equation 3 and Equation 4.
(2) **Stage 4.2 (Distillation):** We train the student model on the small clean dataset, employing the same loss functions as in the previous step, applied to the calibrated outputs of the TA model.

This two-stage procedure **mitigates** backdoors in individual teacher models and produces a **lightweight**, robust student suitable for edge deployment or downstream fine-tuning.

In order to evaluate our method, we design a **worst-case scenario** in which all N teacher models, as shown in Stage 2 of Figure 1, are intentionally backdoored and configured to misclassify inputs with a trigger into the same malicious target label. We show that:

- Even a **simple majority vote** across the N models, as shown in Stage 3 of Figure 1, substantially reduces the attack success rate (ASR) on triggered inputs.
- Building on this, our **multi-stage knowledge distillation**—via the intermediate TA model—not only drives ASR down to near-zero while maintaining accuracy, but also yields a lighter student model.

These results hold not only under controlled attacks but also against unseen or more adaptive backdoors, demonstrating the durability of the Stage 4 pipeline.

While attackers are assumed to have full control over both the training dataset and the model architecture—including the entire training pipeline—our defense operates under **minimal prior knowledge**: we do not know whether the models are backdoored, the backdoor mechanisms or triggers used, or the training data distributions, all of which are randomized in our evaluation to simulate realistic scenarios. The only assumption we make is access to a *small, clean dataset*, used solely during Stage 4 to fine-tune and guide both the TA and student models. This reflects a **realistic and low-privilege scenario**, where defenders have limited resources and no control over the training pipeline of the teacher models.

Our results demonstrate the feasibility of a **generic and practical defense** against unknown backdoor attacks, offering a potential solution to the ongoing arms race between attackers and defenders that often stems from poor generalization or overfitting to specific known attack types.

4 Experimental Setup

4.1 Datasets

We evaluate our defense on two standard image-classification benchmarks: MNIST and CIFAR-10. MNIST contains 60 000 training and 10 000 test grayscale images of handwritten digits at 28 × 28 resolution, while CIFAR-10 comprises 50 000 training and 10 000 test color images of size 32 × 32 × 3.

Let $\mathcal{D}_{\text{orig}}$ denote the original training split. We first partition

$$\mathcal{D}_{\text{orig}} = \mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{val}}, \quad \begin{cases} |\mathcal{D}_{\text{train}}| & = 0.8\,|\mathcal{D}_{\text{orig}}|, \\ |\mathcal{D}_{\text{val}}| & = 0.2\,|\mathcal{D}_{\text{orig}}|. \end{cases}$$

without replacement. To simulate $N = 7$ service-provider datasets, we sample

$$\{\mathcal{D}_i\}_{i=1}^{N}, \quad \mathcal{D}_i \subset \mathcal{D}_{\text{train}}, \quad |\mathcal{D}_i| = 0.4\,|\mathcal{D}_{\text{train}}|, \qquad (1)$$

with inter-subset overlap allowed but without repetition within each subset, simulating realistic partial data sharing across providers. We also carve out a clean auxiliary set

$$\mathcal{D}_{\text{aux}} \subset \mathcal{D}_{\text{train}}, \quad |\mathcal{D}_{\text{aux}}| = 0.1\,|\mathcal{D}_{\text{train}}|$$

using the same sampling procedure. As a result, $\mathcal{D}_{\text{aux}}$ may overlap with one or more provider subsets. At this stage, all subsets remain clean and are poisoned in the following section. The auxiliary set $\mathcal{D}_{\text{aux}}$, however, is kept entirely clean throughout and serves as a trusted dataset for teacher aggregation and student distillation. The original test split $\mathcal{D}_{\text{test}}$ remains untouched for reporting clean-data accuracy (CDA) and attack success rate (ASR).

All images are scaled to $[0, 1]$; we omit further mean–std normalization to simplify trigger injection logic in our code. Owing to limitations in computational resources, we focus on lightweight benchmark datasets rather than large-scale benchmarks such as ImageNet.

4.2 Trigger Types

We categorize the triggers used in our experiments based on their spatial extent and perceptibility. Specifically, we use two types as shown in Figure 2:

(1) **Local triggers** are embedded into a confined region of the image, typically a corner patch. We include
- **SOLID:** a square patch of uniform color.
- **Checkerboard:** an alternating grid pattern with fixed intensity values.
- **Noise:** a square patch filled with randomly generated pixel values.

As described well in [21], all of these local triggers are *visible*. Although SOLID and Checkerboard can be viewed as structured variants of the Noise trigger, we treat them as separate classes due to their prevalence in prior literature, e.g., [2].

(2) **Global triggers** affect the entire image. We use
- **Blend [1]:** an overlay trigger constructed by blending the input with a randomly chosen external image (e.g., from a separate dataset) using a low-opacity factor. While it is referred to as an *invisible* trigger in [21], we argue that it is more accurately described as **partially** invisible.

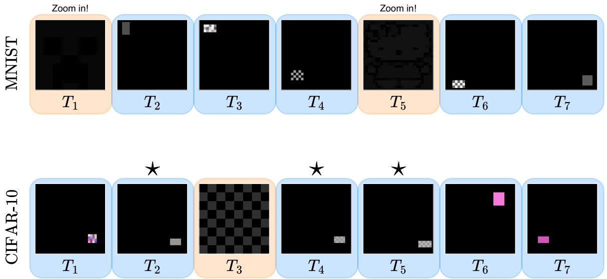

Figure 2. Example trigger instances used across the $N = 7$ teacher models (Stage 1). All triggers are randomly generated per run to promote diversity. Background colors indicate the trigger type: **orange** for global triggers and **blue** for local triggers. *Note:* The starred CIFAR-10 triggers are intentionally generated with approximately 75% similarity to simulate a worst-case scenario.

In our setup, each service provider receives one randomly selected trigger type as shown in Figure 2. A single instance of the trigger is generated per provider (with random parameters such as size, location, intensity) and consistently applied to all selected poisoned samples, reducing designer bias. To simulate a worst-case scenario, we selected three triggers for CIFAR-10 that, despite being randomly generated, were designed to exhibit approximately 75% similarity. This reflects a situation where some service providers may intentionally or unintentionally use similar triggers. In contrast, we did not apply this to MNIST, allowing us to later demonstrate in the results that even with dissimilar triggers, the baseline defense (Stage 3) might fail to defend effectively.

4.3 Poisoning Procedure

Let $\mathcal{D}_i$ denote the i-th outsourced training subset as shown in Equation 1. To simulate backdoor injection, we poison a small fraction $\gamma = 0.01$ of each $\mathcal{D}_i$. Specifically, we define the poisoned subset:

$$\mathcal{D}_i^p \subset \mathcal{D}_i, \quad |\mathcal{D}_i^p| = \gamma\, |\mathcal{D}_i|$$

and denote the remaining clean portion as

$$\mathcal{D}_i^c = \mathcal{D}_i \setminus \mathcal{D}_i^p, \quad \mathcal{D}_i = \mathcal{D}_i^c \cup \mathcal{D}_i^p.$$

Each provider is assigned a unique trigger transformation $\tau_i(\cdot)$, defined by a sampled trigger type and configuration. For each poisoned input $x \in \mathcal{D}_i^p$, the transformation and label replacement are applied as:

$$\tilde{x} = \tau_i(x), \quad \tilde{y} = 0. \tag{2}$$

This process is illustrated in Figure 3, which shows two examples of how a poisoned image is crafted via mask, trigger, and clean Image (with transparency level only for the Blend type). In addition, clean samples in $\mathcal{D}_i^c$ remain untouched, preserving both the original image and the label.

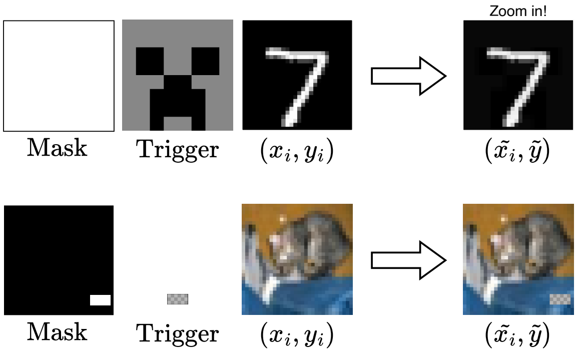

Figure 3. Example of poisoned image generation using mask, trigger, and clean image.

As shown in Equation 2, we adopt a standard *label-flip* strategy with an *all-to-one* mapping to class 0. This minimal setup is justified under the strong attacker assumption, where attackers are assumed to have full control over both the poisoned data and the training process. In such settings, even simple poisoning strategies, such as label flipping with fixed triggers, can be highly effective, and there is no need to consider more sophisticated attack mechanisms. Moreover, we assume all $N = 7$ providers happen to target the same class. This not only increases the attack's coordination difficulty for defenses but also highlights a realistic failure mode: due to visual or spatial similarities among triggers, some models may generalize to others' backdoors, weakening the effectiveness of naive ensemble-based defenses.

4.4 Model Architectures

Let $\mathcal{M}$ denote a ResNet architecture, and let $\widetilde{\mathcal{M}}$ represent its modified variants adapted for $28 \times 28 \times 1$ (MNIST) or $32 \times 32 \times 3$ (CIFAR-10) input dimensions. These modifications enable more meaningful feature representations, leading to improved evaluation quality in the Results section.

For **Clean Baseline (Stage 1),** We train

$$\widetilde{M}_{\mathrm{C}} = \widetilde{\mathrm{ResNet18}}$$

on the clean, unmodified training set $\mathcal{D}_{\mathrm{train}}$ using cross-entropy loss (see Equation 4).

For **Attack Simulation (Stage 2)**, we simulate $N = 7$ untrusted providers by training

$$\widetilde{M}_{\mathrm{T}}^{(i)} = \widetilde{\mathrm{ResNet34}}, \quad i = 1, \ldots, 7,$$

each on its poisoned subset $\mathcal{D}_i$ using cross-entropy loss (see Equation 4). Although this represents a special case of selecting seven models at random from

$$\{\widetilde{\mathrm{ResNet18}}, \widetilde{\mathrm{ResNet34}}, \widetilde{\mathrm{ResNet50}}, \widetilde{\mathrm{ResNet101}}\},$$

we observed negligible differences in performance and therefore adopt a homogeneous ResNet-34 configuration throughout. This choice is not only practical, reflecting a realistic outsourcing scenario in which a client might request all N models with the same architecture, but also beneficial in terms of extensibility. Using identical models allows future integration of more advanced, architecture-dependent techniques (e.g., layer-wise or representation-aware losses [26]), which would be harder to apply consistently across heterogeneous architectures.

In addition to achieving high attack success rates (ASR, Equation 9), each poisoned model must also maintain high clean-data accuracy (CDA, Equation 7), ensuring that it behaves normally on non-poisoned inputs. This dual requirement reflects a realistic threat model, where adversaries aim to implant stealthy backdoors that are undetected by standard evaluation methods.

For **Teacher-Aggregation Model (Stage 4.1)**, we train

$$\widetilde{M}_{\mathrm{TA}} = \widetilde{\mathrm{ResNet34}}$$

on $\mathcal{D}_{\mathrm{aux}}$ using a combined distillation loss:

$$\mathcal{L}_{\mathrm{KL}}(x) = T^2\,\mathrm{KL}\left(\sigma\left(\frac{M_{\mathrm{TA}}(x)}{T}\right) \middle\| \, \sigma\left(\frac{\bar{z}}{T}\right)\right), \tag{3}$$

$$\mathcal{L}_{\mathrm{CE}}(x,y) = \mathcal{L}_{\mathrm{CE}}\left(M_{\mathrm{TA}}(x), y\right), \tag{4}$$

$$\mathcal{L}_{\mathrm{TA}}(x,y) = \alpha\,\mathcal{L}_{\mathrm{KL}}(x) + (1-\alpha)\,\mathcal{L}_{\mathrm{CE}}(x,y). \tag{5}$$

where $T > 0$ is the distillation temperature , $\alpha \in [0,1]$ balances soft and hard terms, $\bar{z}$ denotes the averaged teacher logits, and $\sigma(\cdot)$ is the softmax function.

For **Student Model (Stage 4.2)**, we train

$$\widetilde{M}_{\mathrm{S}} = \widetilde{\mathrm{ResNet18}}$$

on $\mathcal{D}_{\mathrm{aux}}$ using a combined distillation loss:

$$\mathcal{L}_{\mathrm{S}}(x,y) = \alpha\,\mathcal{L}_{\mathrm{KL}}(x) + (1-\alpha)\,\mathcal{L}_{\mathrm{CE}}(x,y). \tag{6}$$

Network Modifications. Starting from the standard ResNet for 224 × 224 inputs, we:

(1) Reduce the first convolution to 3 × 3, stride 1, padding 1.
(2) Remove the initial max-pool.
(3) Omit the fourth residual block.
(4) Reconfigure the final fully-connected layer to map Layer 3 outputs to $|\mathcal{Y}|$ classes.

These adjustments reduce parameters, eliminate unnecessary downsampling, and enable efficient training under hardware constraints.

4.5 Defense Assumptions

We make the following assumptions about the defense setup:

(1) **Model Access:** We have full access to the trained teacher models $\{\widetilde{M}_{\mathrm{T}}^{(i)}\}_{i=1}^{N}$, but no control over their training data or procedures.
(2) **Unknown Poisoning:** We do not know whether any $\widetilde{M}_{\mathrm{T}}^{(i)}$ is backdoored, nor any details of its trigger (e.g. size, location, visibility, spatial scope, or whether it is static vs. learned).
(3) **Trusted Distillation Set:** We assume a small clean distillation dataset is available and free of any backdoor injections.
(4) **Data Sharing:** We have no knowledge of the original outsourced subsets $\{\mathcal{D}_i\}_{i=1}^{N}$. Overlap between $\mathcal{D}_{\mathrm{aux}}$ and $\{\mathcal{D}_i\}$ before poisoning is possible but not guaranteed.
(5) **Compute Constraints:** Due to limited computational resources, we can not train large-scale models (e.g., ImageNet-scale ResNets) from scratch or process massive datasets. Instead, we leverage the $N = 7$ teacher models provided by external services to amortize compute costs.

4.6 Evaluation Metrics

We quantify defense performance using three primary metrics—Clean-Data Accuracy (CDA), Robust Accuracy (R-ACC), and Attack Success Rate (ASR)—along with their relative differences to the clean baseline. Additionally, we also report a model-complexity measure to demonstrate time and space overhead.

Clean-Data Accuracy (CDA). Let $\mathcal{D}_{\mathrm{test}}$ be the unmodified test set. For any model $\mathcal{M}$,

$$\mathrm{CDA}(\mathcal{M}) = \frac{1}{N_{\mathrm{test}}}\sum_{j=1}^{N_{\mathrm{test}}} \mathbf{1}[\mathcal{M}(x_j) = y_j]. \tag{7}$$

A high CDA indicates that the model behaves normally on clean inputs, correctly classifying non-poisoned data. To emphasize robustness, we also report the drop relative to the clean baseline:

$$\Delta\mathrm{CDA}(\mathcal{M}) = \mathrm{CDA}(M_{\mathrm{C}}) - \mathrm{CDA}(\mathcal{M}),$$

where M_{C} is the Stage 1 clean model baseline. A small ΔCDA indicates minimal degradation on clean inputs.

Attack Success Rate (ASR). For trigger transformation τ and target poison class $\tilde{y}$, define the poisoned test set

$$\widetilde{\mathcal{D}}_{\mathrm{test}}^{\tau} = \{(\tau(x_j), \tilde{y}) : (x_j, y_j) \in \mathcal{D}_{\mathrm{test}},\, y_j \neq \tilde{y}\}. \tag{8}$$

We excluded original class-$\tilde{y}$ samples to avoid label ambiguity. Then,

$$\mathrm{ASR}(M,\tau) = \frac{1}{|\widetilde{\mathcal{D}}_{\mathrm{test}}^{\tau}|}\sum_{(\tilde{x},\tilde{y})\in\widetilde{\mathcal{D}}_{\mathrm{test}}^{\tau}} \mathbf{1}[M(\tilde{x}) = \tilde{y}]. \tag{9}$$

A high ASR indicates strong vulnerability, meaning the model consistently misclassifies triggered inputs as the attacker's target class. To emphasize robustness, we also report the increase relative to the clean baseline:

$$\Delta\mathrm{ASR}(M,\tau) = \mathrm{ASR}(M,\tau) - \mathrm{ASR}(M_{\mathrm{C}},\tau),$$

Robust Accuracy (R-ACC). Let $\widetilde{\mathcal{D}}_{\mathrm{test}}^{\tau}$ be the poisoned version of the test set as shown in Equation 8, where $\tilde{x}_j$ is a triggered input with true label y_j. For any model $\mathcal{M}$,

$$\mathrm{RACC}(\mathcal{M}) = \frac{1}{N_{\mathrm{test}}}\sum_{j=1}^{N_{\mathrm{test}}} \mathbf{1}[\mathcal{M}(\tilde{x}_j) = y_j]. \tag{10}$$

A high RACC indicates that the model is robust to backdoor triggers and correctly predicts the true labels even when inputs are poisoned. To emphasize robustness, we also report the drop relative to the clean baseline:

$$\Delta\mathrm{RACC}(\mathcal{M}) = \mathrm{RACC}(M_{\mathrm{C}}) - \mathrm{RACC}(\mathcal{M}),$$

where M_{C} is the Stage 1 clean model baseline. A small ΔRACC indicates strong resistance to backdoor activations.

Model Complexity. We report and compare the number of trainable parameters in the models. One of the key goals of this work is to reduce the complexity of the model while preserving performance. In particular, we highlight the efficiency gains achieved by compressing multiple teacher models into a single student. Parameter counts serve as a proxy for both memory usage and computational cost, and reductions are quantified and compared across settings.

4.7 Backdoor Analysis

To visualize how backdoors and our defense affect internal representations, we perform the following analyses:

Feature-Map Inspection. For a model M and a poisoned input $\tilde{x}_j = \tau(x_j)$, let $f_\ell(x)$ denote the activation tensor at layer ℓ. We extract and visualize channel-wise activations:

$$\{\, f_\ell(\tilde{x}_j)[c,:,:]\}_{c=1}^{C_\ell} \quad \text{for } \ell \in \{\text{layer1}, \text{layer2}, \text{layer3}\}. \tag{11}$$

Discrepancies in these feature maps reveal whether the trigger still dominates intermediate representations.

Grad-CAM Visualization. We apply Grad-CAM [27] to the last convolutional layer (layer3). For target class t, we compute:

$$\alpha_c^t = \frac{1}{Z} \sum_{i,j} \frac{\partial y^t}{\partial f_3(x)_{c,i,j}},$$

$$\text{GradCAM}(x)_{i,j} = \text{ReLU}\Big(\sum_c \alpha_c^t \, f_3(x)_{c,i,j}\Big). \tag{12}$$

where:

- $f_3(x)_{c,i,j}$ is the activation at channel c, spatial location (i, j) in layer3.
- α_c^t is the channel-wise weight, computed as the spatial average of the gradient of the target logit y^t with respect to that channel's activations.
- Z is the total number of spatial positions (i.e., $Z = H \times W$).
- The final ReLU retains only positive contributions.

Grad-CAM visualizations reveal whether the model's attention is still hijacked by the trigger. This helps validate that our defense pipeline effectively suppresses trigger-induced activations.

Cross-Trigger Generalization. To assess vulnerability to unseen trigger transformations, we evaluate cross-trigger attack success. For each teacher model $\widetilde{M}_{\mathrm{T}}^{(i)}$ trained with transformation τ_i, we measure $\text{ASR}(\widetilde{M}_{\mathrm{T}}^{(i)}, \tau_j)$ for all $j \neq i$. Elevated cross-trigger ASR indicates overlapping feature reliance, which may weaken simple ensemble defenses by exposing shared vulnerabilities.

4.8 Training Details

All models across the pipeline—including the Stage 1 clean baseline, Stage 2 poisoned models, Stage 4.1 aggregation model, and Stage 4.2 student model—are trained using a unified configuration. Each is trained for 15 epochs, which we found sufficient for convergence on both MNIST and CIFAR-10.

We use a batch size of 64 for training and 128 for both validation and testing phases. All models are optimized using Adam with a fixed initial learning rate $\eta_0 = 0.01$ and PyTorch's default values for $(\beta_1, \beta_2, \epsilon)$. A learning-rate scheduler is applied with settings: `mode=‘‘min’’`, `factor=0.5`, `patience=2`, and `threshold=`10^{-3}, to reduce the learning rate upon plateau in validation loss.

All experiments are implemented in Python v3.12.8, primarily using PyTorch v2.5.1+cu124 and Torchvision v0.20.1+cu124, along with other supporting libraries where needed. Training and evaluation are conducted on a consumer-grade laptop (Victus 15-fa0xxx) equipped with a 12th Gen Intel® Core™ i5-12450H CPU and an NVIDIA GeForce GTX 1650 GPU (4 GB GDDR6 VRAM, driver version 566.36).

For reproducibility, we initialize all random seeds to 0 using PyTorch, NumPy, and Python's built-in `random` module. Nonetheless, PyTorch does not guarantee strict determinism across all operations or hardware configurations. Therefore, minor variations across runs or devices may still occur, even with fixed seeds.

5 Results and Discussion

5.1 Evaluation of Clean and Poisoned Models

To understand the impact of backdoor attacks on model performance, we compare a clean baseline model (Stage 1) with $N = 7$ individually poisoned models (Stage 2) trained on different poisoned subsets. The results for MNIST Table 1a and CIFAR-10 Table 1b are reported in Table 1. For the clean model, ASR and R-ACC are measured using all N poisoned test sets.

As shown in Table 1, poisoned models maintain clean accuracy (CDA $\approx$ 0.98–0.99 on MNIST, $\approx$ 0.73–0.79 on CIFAR-10), showing the stealthiness of backdoors. Meanwhile, their ASR remains high (ASR $\approx$ 1.0 on MNIST, $\approx$ 0.76–1.0 on CIFAR-10), and R-ACC drops to nearly zero (R-ACC $\approx$ 0.0 on MNIST, $\approx$ 0.0–0.2 on CIFAR-10), confirming that the models have memorized the backdoor and misclassify clean inputs with embedded triggers.

5.2 Evaluation of Aggregation-Based Defenses

To evaluate the effectiveness of our aggregation-based defense, we compare the performance of baseline ensemble methods—soft and hard majority voting (Stage 3)—against both of our proposed strategists, our teacher-aggregated model ($\widetilde{M}_{\mathrm{TA}}$) and light-weight

Table 1. Comparison of clean and poisoned models (Stage 1 vs Stage 2) on MNIST and CIFAR-10. Metrics are reported as CDA (Clean Accuracy), ASR (Attack Success Rate), R-ACC (Robust Accuracy). Δ values show the drop or increase relative to the clean baseline.

(a) MNIST Results

Model	**CDA**	**ASR**	**R-ACC**	ΔCDA	ΔASR	ΔR-ACC
$\widetilde{M}_{\mathrm{C}}$	**0.994**	**0.000**	**0.953**	–	–	–
$\widetilde{M}_{\mathrm{T}}^{(1)}$	0.993	1.000	0.000	0.001	1.000	0.953
$\widetilde{M}_{\mathrm{T}}^{(2)}$	0.993	1.000	0.000	0.001	1.000	0.953
$\widetilde{M}_{\mathrm{T}}^{(3)}$	0.989	0.999	0.001	0.005	0.999	0.952
$\widetilde{M}_{\mathrm{T}}^{(4)}$	0.991	1.000	0.000	0.003	1.000	0.953
$\widetilde{M}_{\mathrm{T}}^{(5)}$	0.989	1.000	0.000	0.005	1.000	0.953
$\widetilde{M}_{\mathrm{T}}^{(6)}$	0.984	0.999	0.001	0.010	0.999	0.952
$\widetilde{M}_{\mathrm{T}}^{(7)}$	0.993	1.000	0.000	0.001	1.000	0.953

(b) CIFAR-10 Results

Model	**CDA**	**ASR**	**R-ACC**	ΔCDA	ΔASR	ΔR-ACC
$\widetilde{M}_{\mathrm{C}}$	**0.853**	**0.029**	**0.767**	–	–	–
$\widetilde{M}_{\mathrm{T}}^{(1)}$	0.729	0.875	0.105	0.124	0.846	0.662
$\widetilde{M}_{\mathrm{T}}^{(2)}$	0.788	0.764	0.203	0.065	0.735	0.564
$\widetilde{M}_{\mathrm{T}}^{(3)}$	0.758	0.995	0.003	0.095	0.966	0.764
$\widetilde{M}_{\mathrm{T}}^{(4)}$	0.755	0.793	0.167	0.098	0.764	0.600
$\widetilde{M}_{\mathrm{T}}^{(5)}$	0.774	0.970	0.026	0.079	0.941	0.741
$\widetilde{M}_{\mathrm{T}}^{(6)}$	0.758	0.975	0.024	0.095	0.946	0.743
$\widetilde{M}_{\mathrm{T}}^{(7)}$	0.776	0.968	0.027	0.077	0.939	0.740

student model ($\widetilde{M}_{\mathrm{S}}$). All results are benchmarked against the clean model $\widetilde{M}_{\mathrm{C}}$. Table 2 presents the CDA, and the mean ASR and R-ACC values averaged over $N = 7$ independently poisoned subsets for MNIST Table 2a and CIFAR-10 Table 2b datasets.

As shown in Table 2 which compares the baseline ensemble defenses (soft and hard voting) with our proposed aggregation-based models on **MNIST** and **CIFAR-10**. On **MNIST**, while soft and hard voting moderately reduce ASR (to 17.2% and 5.6% respectively), they do so with notable degradation in R-ACC. In contrast, both of the proposed scenario, $\widetilde{M}_{\mathrm{TA}}$ and $\widetilde{M}_{\mathrm{S}}$ achieve near-zero ASR (0.1%) and even improve R-ACC beyond the clean baseline (from 95.3% to 98.6%), demonstrating strong robustness without compromising clean accuracy. This suggests that aggregation followed by distillation can neutralize backdoors more effectively than naive majority voting.

On **CIFAR-10**, soft and hard voting reduce ASR to 12.0% and 5.1% respectively, with hard voting slightly outperforming soft voting in terms of R-ACC (0.769 vs. 0.696). However, both suffer from a noticeable drop in clean accuracy (CDA). In contrast, our teacher-aggregated models $\widetilde{M}_{\mathrm{TA}}$ achieves the **lowest ASR** (2.6%), effectively neutralizing the backdoor. This results in a reduction in CDA and R-ACC, which is likely attributed to the increased complexity of the CIFAR-10 dataset and the limited size of the clean subset used for aggregation. The student model $\widetilde{M}_{\mathrm{S}}$ maintains a similar ASR (5.4%) as compared to hard vot- ing results, though it still lags behind in CDA and R-ACC. These results suggest that while our defense is effective in suppressing backdoor behavior, performance on complex datasets like CIFAR-10 is sensitive to the size of the clean subset used. A larger subset during aggregation and distillation may help improve generalization and preserve clean accuracy.

Table 2. Comparison of ensemble-based (Stage 3) and our aggregation-based (Stage 4) defenses against the clean model baseline. ASR and R-ACC are reported as mean values averaged over $N = 7$ independently poisoned subsets. The Δ columns show differences relative to the clean baseline.

(a) MNIST Results

Model	**CDA**	**ASR**	**R-ACC**	ΔCDA	ΔASR	ΔR-ACC
$\widetilde{M}_{\mathrm{C}}$	**0.994**	**0.000**	**0.953**	–	–	–
Soft Vote	**0.995**	0.172	0.824	**-0.001**	0.172	0.129
Hard Vote	**0.995**	0.056	0.939	**-0.001**	0.056	0.014
$\widetilde{M}_{\mathrm{TA}}$	0.989	**0.001**	**0.986**	0.005	**0.001**	**-0.033**
$\widetilde{M}_{\mathrm{S}}$	0.989	**0.001**	**0.986**	0.005	**0.001**	**-0.033**

(b) CIFAR-10 Results

Model	**CDA**	**ASR**	**R-ACC**	ΔCDA	ΔASR	ΔR-ACC
$\widetilde{M}_{\mathrm{C}}$	**0.853**	**0.029**	**0.767**	–	–	–
Soft Vote	**0.839**	0.120	0.696	**0.014**	0.091	0.071
Hard Vote	0.831	0.051	**0.769**	0.022	0.022	**-0.002**
$\widetilde{M}_{\mathrm{TA}}$	0.686	**0.026**	0.603	0.167	**-0.003**	0.164
$\widetilde{M}_{\mathrm{S}}$	0.668	0.054	0.582	0.185	0.025	0.185

5.3 Model Size Comparison

To evaluate the efficiency of our approach, we compare the number of parameters required by the baseline ensemble defense (Stage 3) and our proposed method (Stage 4). As shown in Table 3, Stage 4 achieves a significant reduction in model size. While the aggregated teacher model $\widetilde{M}_{\mathrm{TA}}$ retains comparable complexity to a single model, the final student model $\widetilde{M}_{\text{S}}$ is lightweight and suitable for deployment in resource-constrained settings.

Table 3. Comparison of total model parameters between the baseline ensemble (with $N = 7$ models) and the proposed defense components. The numbers are based on CIFAR-10 experiments. Minor variations exist across datasets due to differences in input dimensions.

Defense Method	Model(s)	Total Parameters
Baseline Ensemble	7 $\times\widetilde{\text{ResNet34}}$	7 × 8.2M
$\widetilde{M}_{\text{TA}}$	$\widetilde{\text{ResNet34}}$	8.2M
$\widetilde{M}_{\text{S}}$	$\widetilde{\text{ResNet18}}$	**2.8M**

5.4 Visual Analysis Using GradCAM

To gain deeper insight into how different models respond to trigger patterns, we employ GradCAM [27] to visualize class-discriminative regions. This analysis helps interpret the internal behavior of models trained under different stages of our pipeline.

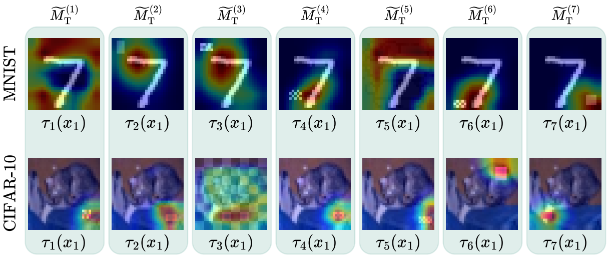

Figure 4. GradCAM visualizations for a representative poisoned model $\widetilde{M}_{\text{T}}^{(1)}$ reacting to poisoned samples. Trigger regions dominate attention across both MNIST and CIFAR-10 inputs.

Figure 4 presents GradCAM heatmaps for poisoned models from Stage 2, namely $\widetilde{M}_{\text{T}}^{(i)}$. We use one poisoned sample from MNIST and one from CIFAR-10 (typically the first test image) and apply all $N = 7$ trigger variants (as shown in Figure 2). The visualizations reveal that the poisoned model consistently attends to the trigger regions, regardless of the input's semantic content.

Figure 5 shows the corresponding GradCAM visualizations for our proposed defense models from stage 4: $\widetilde{M}_{\text{TA}}$ and $\widetilde{M}_{\text{S}}$. Compared to Stage 2, these models exhibit reduced attention to the trigger regions and greater focus on semantically meaningful parts of the input, indicating successful mitigation of backdoor behavior.

6 Conclusion

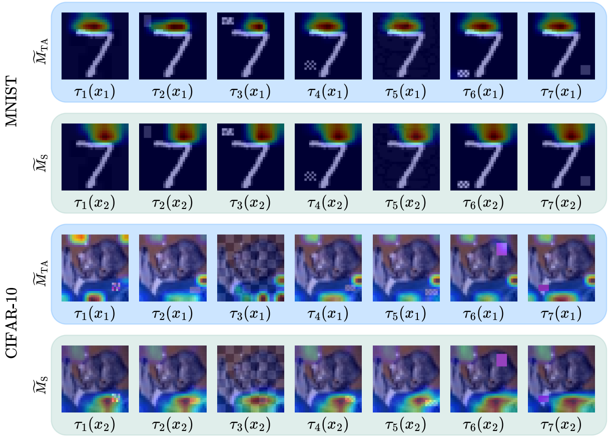

Figure 5. GradCAM visualizations for $\widetilde{M}_{\text{TA}}$ and $\widetilde{M}_{\text{S}}$, showing reduced attention to trigger regions in defended models.

In this paper, we proposed a simple yet effective two-stage defense pipeline against backdoor attacks, grounded in model aggregation and knowledge distillation. Our method requires no prior knowledge of the target model's training process or data and relies only on a small, trusted subset of clean samples. Experimental results demonstrate that it can reduce the attack success rate to near zero while maintaining acceptable clean accuracy. The resulting model is both lightweight and robust, making it well-suited for deployment in resource-constrained environments such as edge devices. In future work, we aim to explore the integration of state-of-the-art defense techniques with individually poisoned models prior to applying our pipeline, and to investigate strategies for preserving higher clean accuracy under limited clean data availability.

References

[1] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

[2] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019.

[3] Shihao Zhao, Xingjun Ma, Xiang Zheng, James Bailey, Jingjing Chen, and Yu-Gang Jiang. Clean-label backdoor attacks on video recognition models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14443–14452, 2020.

[4] Yuezun Li, Yiming Li, Baoyuan Wu, Longkang Li, Ran He, and Siwei Lyu. Invisible backdoor attack with sample-specific triggers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16463–16472, 2021.

[5] Yunfei Liu, Xingjun Ma, James Bailey, and Feng Lu. Reflection backdoor: A natural backdoor

attack on deep neural networks. In *European Conference on Computer Vision*, pages 182–199. Springer, 2020.

[6] Tong Wang, Yuan Yao, Feng Xu, Shengwei An, Hanghang Tong, and Ting Wang. An invisible black-box backdoor attack through frequency domain. In *European Conference on Computer Vision*, pages 396–413. Springer, 2022.

[7] Qiuyu Duan, Zhongyun Hua, Qing Liao, Yushu Zhang, and Leo Yu Zhang. Conditional backdoor attack via jpeg compression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19823–19831, 2024.

[8] Sheng Yang, Jiawang Bai, Kuofeng Gao, Yong Yang, Yiming Li, and Shu-Tao Xia. Not all prompts are secure: A switchable backdoor attack against pre-trained vision transfomers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24431–24441, 2024.

[9] Shuo Wang, Surya Nepal, Carsten Rudolph, Marthie Grobler, Shangyu Chen, and Tianle Chen. Backdoor attacks against transfer learning with pre-trained deep learning models. *IEEE Transactions on Services Computing*, 15(3):1526–1539, 2020.

[10] Sanghyun Hong, Michael-Andrei Panaitescu-Liess, Yigitcan Kaya, and Tudor Dumitras. Qu-anti-zation: Exploiting quantization artifacts for achieving adversarial outcomes. *Advances in Neural Information Processing Systems*, 34:9303–9316, 2021.

[11] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE symposium on security and privacy (SP)*, pages 707–723. IEEE, 2019.

[12] Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu 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*, pages 1265–1282, 2019.

[13] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International symposium on research in attacks, intrusions, and defenses*, pages 273–294. Springer, 2018.

[14] Yansong Gao, Change Xu, Derui Wang, Shiping Chen, Damith C Ranasinghe, and Surya Nepal. Strip: A defence against trojan attacks on deep neural networks. In *Proceedings of the 35th annual computer security applications conference*, pages 113–125, 2019.

[15] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. Anti-backdoor learning: Training clean models on poisoned data. *Advances in Neural Information Processing Systems*, 34:14900–14912, 2021.

[16] Kota Yoshida and Takeshi Fujino. Disabling backdoor and identifying poison data by using knowledge distillation in backdoor attacks on deep neural networks. In *Proceedings of the 13th ACM workshop on artificial intelligence and security*, pages 117–127, 2020.

[17] Zonghao Ying and Bin Wu. Nba: defensive distillation for backdoor removal via neural behavior alignment. *Cybersecurity*, 6(1):20, 2023.

[18] Yinshan Li, Hua Ma, Zhi Zhang, Yansong Gao, Alsharif Abuadbba, Minhui Xue, Anmin Fu, Yifeng Zheng, Said F Al-Sarawi, and Derek Abbott. Ntd: Non-transferability enabled deep learning backdoor detection. *IEEE Transactions on Information Forensics and Security*, 19:104–119, 2023.

[19] Boheng Li, Yishuo Cai, Haowei Li, Feng Xue, Zhifeng Li, and Yiming Li. Nearest is not dearest: Towards practical defense against quantization-conditioned backdoor attacks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 24523–24533, 2024.

[20] Huaibing Peng, Huming Qiu, Hua Ma, Shuo Wang, Anmin Fu, Said F Al-Sarawi, Derek Abbott, and Yansong Gao. On model outsourcing adaptive attacks to deep learning backdoor defenses. *IEEE Transactions on Information Forensics and Security*, 19:2356–2369, 2024.

[21] Baoyuan Wu, Hongrui Chen, Mingda Zhang, Zihao Zhu, Shaokui Wei, Danni Yuan, and Chao Shen. Backdoorbench: A comprehensive benchmark of backdoor learning. *Advances in Neural Information Processing Systems*, 35:10546–10559, 2022.

[22] Zhihuan Xing, Yuqing Lan, Yin Yu, Yong Cao, Xiaoyi Yang, Yichun Yu, and Dan Yu. Bdel: A backdoor attack defense method based on ensemble learning. In *Pacific Rim International Conference on Artificial Intelligence*, pages 221–235. Springer, 2024.

[23] Zijie Zhang, Xinyuan Miao, Chenyu Zhou, Chenming Shang, Xi Chen, Xianglong Kong, Wei Huang, and Yi Cao. Bdekd: mitigating backdoor attacks in nlp models via ensemble knowledge distillation. *Complex & Intelligent Systems*, 11(9):1–17, 2025.

[24] Yifan Wang, Wei Fan, Keke Yang, Naji Alhusaini, and Jing Li. A knowledge distillation-based backdoor attack in federated learning. *arXiv*

preprint arXiv:2208.06176, 2022.

[25] Chengcheng Zhu, Jiale Zhang, Xiaobing Sun, Bing Chen, and Weizhi Meng. Adfl: Defending backdoor attacks in federated learning via adversarial distillation. *Computers & Security*, 132:103366, 2023.

[26] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. Neural attention distillation: Erasing backdoor triggers from deep neural networks. *arXiv preprint arXiv:2101.05930*, 2021.

[27] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: visual explanations from deep networks via gradient-based localization. *International journal of computer vision*, 128:336–359, 2020.

Amirhossein Heydari received the B.S. degree in computer engineering from Guilan University, Guilan, Iran, in 2022. He is currently pursuing the M.S. degree in computer engineering with a specialization in artificial intelligence and robotics at Kharazmi University, Tehran, Iran. His research interests include backdoor attacks and defenses in deep neural networks, adversarial machine learning, robustness and security of AI systems.

Azadeh Mansouri received the B.S. degree in computer engineering from Shiraz University, Shiraz, Iran, in 2002, the M.S. degree from the Science and Research Branch, Azad University of Tehran, and Ph.D degree from the Department of Electrical and Computer Engineering, Shahid Beheshti University, Tehran, Iran, in 2010. In 2011, she joined the Department of Electrical and Computer Engineering, Kharazmi University, Tehran, Iran, as an assistant professor. Her research interests include image/video processing and machine learning algorithms with an emphasis on digital forensics, quality assessment and enhancement methods.

Ahmad Mahmoudi-Aznaveh is an assitant professor at cyberspace research institute at Shahid Beheshti University. He received the B.S. degree in computer engineering from Isfahan University of Technology, Isfahan, Iran, in 2003, and the M.S. and Ph.D. degrees from the Department of Electrical and Computer Engineering, Shahid Beheshti University, Tehran, Iran, in 2005 and 2010 respectively. His areas of research interest include image/video processing and machine learning, especially visual quality assessment, forensics applications and human action recognition.