Deep Learning Backdoor Defense via Adaptive Trigger Collisions in Latent Space

Zixun Xiong
zxiong9@stevens.edu
Stevens Institute of Technology
Hoboken, New Jersey, USA

Hao Wang
hwang9@stevens.edu
Stevens Institute of Technology
Hoboken, New Jersey, USA

Jian Li
jian.li.3@stonybrook.edu
Stony Brook University
Stony Brook, New York, USA

Yang Hua
Y.Hua@qub.ac.uk
Queen's University Belfast
Belfast, Northern Ireland, UK

Miao Pan
mpan2@uh.edu
University of Houston
Houston, Texas, USA

Xiaojiang Du
xdu16@stevens.edu
Stevens Institute of Technology
Hoboken, New Jersey, USA

Abstract

Backdoor attacks in data outsourcing settings pose severe risks to deep neural networks. Specifically, adversaries can manipulate externally sourced training data to implant hidden behaviors in target models (*e.g.*, incorrect predictions on triggered samples). Existing defenses are either pre-processing or post-processing. Since the two approaches are orthogonal and either one can independently strengthen real-world defenses, we focus on the latter in this paper. Yet current post-processing defenses face one or more of the following issues: overemphasis on output logits while overlooking rich information in intermediate layers, injection of uncertain new triggers while requiring alignment with the original triggers, and underuse of poisoned model representations. To overcome the aforementioned limitations, we propose *ATClean*, an adaptive post-processing defense based on feature collisions in latent space. Specifically, it leverages all layers rather than only output logits to capture backdoor-affected regions using an adaptive loss function, relaxes the need for exact trigger reconstruction by generating adversarial samples that only enforce feature collisions with a theoretical guarantee, and fully exploits poisoned representations with feature-collision-based fine-tuning. Experiments across benchmark datasets, multiple architectures, and seven representative attacks show that *ATClean* achieves state-of-the-art defense effectiveness with the lowest drop on clean data, including about a 20% improvement in DER, which measures the accuracy–defense trade-off.

CCS Concepts

• **Security and privacy** → **Artificial intelligence security**; • **Computing methodologies** → *Machine learning*; *Neural networks*.

Keywords

Backdoor defense, Trustworthy AI

ASIA CCS '26, Bangalore, India

ACM ISBN 979-8-4007-2356-8/26/06
https://doi.org/10.1145/3779208.3806081

ACM Reference Format:
Zixun Xiong, Hao Wang, Jian Li, Yang Hua, Miao Pan, and Xiaojiang Du. 2026. Deep Learning Backdoor Defense via Adaptive Trigger Collisions in Latent Space. In *Proceedings of the 21st ACM ASIA Conference on Computer and Communications Security (ASIA CCS '26)*. ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3779208.3806081

1 Introduction

Deep neural networks (DNNs) have been proven effective in diverse security-critical fields (*e.g.*, autonomous driving [18, 32], face recognition [19, 22]). However, training DNNs consumes large volumes of high-quality data samples, prompting developers to increasingly rely on *data outsourcing*, a practice that may introduce serious vulnerabilities to backdoor attacks [3, 15, 26, 36, 42, 59].

Specifically, in backdoor attacks targeting data outsourcing scenarios, the victim trains models using externally sourced data, potentially poisoned by attackers. Moreover, these attacks are particularly stealthy, as the backdoored model performs normally on clean inputs (*i.e.*, input samples without triggers injected), and only misclassifies poisoned samples (*i.e.*, input samples with triggers injected) that activate the backdoor [15, 26, 36, 39]. Consequently, these attacks pose a significant threat; for instance, A backdoored model can cause a vehicle to misidentify traffic signs, potentially leading to accidents [15]. Thus, backdoor attacks are regarded as one of the most severe threats to deep learning systems [39].

Extensive research has been devoted to mitigating backdoor attacks, with existing approaches broadly classified into two categories: *pre-processing* and *post-processing* defenses. Pre-processing defenses aim to remove poisoned samples or features from the training data before or during the training [7, 13, 43, 60], but they often suffer from incomplete detection [33] and limited robustness against diverse or adaptive attack strategies [33]. In this paper, we focus on an orthogonal defense, and improving it helps address the limitations of the pre-processing defenses, leading to stronger results when deployed together: Post-processing defenses focus on mitigating the effects of backdoors embedded in a poisoned model [16, 27, 28, 31, 33, 52, 56, 57, 61, 63, 68]. However, previous post-processing methods suffer from several limitations. First, they focus mainly on output logits while ignoring useful information embedded in other layers [58]. Second, some approaches require injecting new triggers, whose effectiveness depends on their uncertain alignment with the original ones [56]. Finally, others overlook

the usage of the poisoned model representations, resulting in sub-optimal defense performance [58].

Therefore, we propose *ATClean*, an adaptive post-processing defense. First, rather than focusing solely on output logits, *ATClean* adaptively explores all layers of the poisoned model to locate backdoor-affected regions, ensuring more comprehensive utilization of internal representations. Second, instead of injecting new triggers whose effectiveness depends on uncertain alignment with the original ones, *ATClean* fine-tunes the model using adversarially crafted samples from the identified layers, directly mitigating malicious features. Finally, *ATClean* fully exploits poisoned representations with feature-collision-based fine-tuning. Through this design, *ATClean* effectively removes backdoors without the weaknesses of existing post-processing approaches.

In practice, *ATClean* defends against backdoors through four key steps. First, we train a target model on the questioned dataset to obtain an initial backdoored model. Second, we select suspicious samples using the backdoored model and any detection method with minimal precision requirements (*i.e.*, precision > 0.5).[1] Third, we train a generator using suspicious samples with adaptive feature collisions to produce adversarial triggers that induce feature collisions at the most sensitive layers of the model. Finally, we fine-tune the backdoored model on clean samples and samples generated by the generator, effectively removing backdoors through feature collisions without significantly degrading normal performance. In summary, we mainly make the following contributions:

- We propose *ATClean*, the first post-processing backdoor defense method that leverages feature collisions in the latent feature space to disrupt backdoors and neutralize poisoned models.
- Unlike previous post-processing defenses that rely on accurate reconstruction or primarily focus on output logits, *ATClean* crafts triggers in latent space by optimizing adaptive feature collisions across all layers.
- We evaluate *ATClean*'s effectiveness against seven representative state-of-the-art (SOTA) backdoor attacks plus one adaptive attack. By comparing with seven representative SOTA backdoor defenses as baselines on various datasets and models, *ATClean* achieves the SOTA defense effectiveness and by far the lowest performance drop on clean data among successful defenses with approximately a 20% improvement of DER, which measures the accuracy-defense trade-off.

2 Preliminaries

2.1 Threat Model

Following is the threat model for backdoor attacks targeting data outsourcing scenarios.

Attackers. Following existing backdoor attacks targeting data-outsourcing scenarios [6, 15, 25, 26, 29, 45, 58, 59], we assume attackers may gain access to a portion of the training data and manipulate both the input images and their corresponding labels. However, the attacker cannot control the training process. The attacker's objective is to compel the target model f_w to classify data samples $x_i \in \mathcal{X}$ with embedded triggers p (*i.e.*, $b(x_i, p)$) into a specific incorrect label c or a set C of incorrect labels. This objective can be formalized as $f_w(b(x_i, p)) = c$. For simplicity, we only formalize the single-label attack scenario here without lacking generality.[2] This attack can be achieved by training the target model f_w on a poisoned dataset, defined as $X_{questioned} := \{(b(x_i, p), c) \mid (x_i, y_i) \in X_{train}\} \cup \{(x_i, y_i) \mid (x_i, y_i) \in X_{train}\}$ where X_{train} is the training dataset.

Defenders. The defender aims to train the target model f_w on $X_{questioned}$ without introducing the backdoor [25, 29, 58]. Existing post-processing defense assume that the defender control the training process and can use a clean dataset, also known as the *root dataset* X_{root}, as a reference to prevent backdoor attacks while maintaining the main task accuracy [12, 31, 41, 52, 54, 57, 67, 68]. Typically, the root dataset X_{root}'s size is 5%–10% of the training dataset $X_{questioned}$, assuming it has the same clean feature distribution as $X_{questioned}$ but has no overlap with $X_{questioned}$. To further enhance the generality of our defense, we also consider a more challenging setting where the defender has no control over the training process and only receives a pre-trained, potentially poisoned model. In this scenario, the goal is to remove the embedded backdoor without access to the original training pipeline and data (see Section 5).

2.2 Backdoor Attacks

A *backdoor attack* on DNNs in data outsourcing scenarios compromises a model to perform well on benign inputs while misbehaving on inputs with attacker-defined triggers. Such attacks can be categorized as either *perturbation-based* [2, 3, 6, 15, 26, 37, 42, 51, 66] or *transformation-based* [36, 59], depending on how the trigger is crafted.

Perturbation-based Attacks use small noise perturbations as triggers, making malicious samples hard to detect from the original ones. For example, BadNet [15] stamps a 5×5 white square to the training image samples as the backdoor trigger and mislabeled the stamped images with a designated target label to train and compromise a DNN. Following BadNet, Blend [6] emphasizes that poisoned images should look visually similar to benign ones to evade human detection. Instead of directly stamping the trigger, Blend generates poisoned samples by blending benign images. Observing that prior defenses against BadNet and Blend attacks assume uniform triggers, ISSBA [26] proposes an optimization-based method to generate invisible, sample-specific triggers. BELT [42] is another optimization-based attack targeting defenses that is based on trigger recovery. Besides, clean label attacks (*e.g.*, SIG [3]) poison training examples while leaving labels unchanged, making them more stealthy and resistant to data filtering and detection.

Transformation-based Attacks create triggers by moving and distorting training samples, making them less noticeable to humans compared to perturbation-based attacks. For instance, WaNet [36] subtly distorts clean samples as triggers, making poisoned images both invisible and sample-specific. Similarly, BATT [59] uses affine transformations, such as rotations and translations, to poison training samples, achieving both invisibility and sample-specificity.

[1] We also provide a detector-free and root-data-free version to cover more challenging scenarios in Sections 5 and C.3.

[2] It can be extended to multiple-label scenarios easily, and we also conduct multiple-label attacks in experiments.

2.3 Backdoor Defenses

Many backdoor defense methods have been proposed to eliminate backdoor attacks in data outsourcing scenarios. According to the stage at which backdoor defenses operate, there are two types: *pre-processing* and *post-processing* defenses [7, 57, 68].

Pre-processing Defenses [4, 8, 11, 13, 17, 20, 21, 27, 30, 35, 40, 50, 55, 60] focus on either removing poisoned data or poisoned features from the training data before or during the training. Thus, they can be further categorized as pre-training methods (*e.g.*, REFINE [7]) and in-training methods (*e.g.*, ABL [27]). Pre-training methods avoid learning backdoors before the training. This method alone cannot achieve a successful defense when a fraction of poisoned samples escapes detection [33]. In-training methods attempt to remove the backdoor feature during the training. These methods alone have limited effects as they also depend on the high precision of their backdoor detection [33]. Thereby, we focus on post-processing defenses in this paper.

Post-processing Defenses [16, 28, 31, 33, 41, 52, 54, 56, 57, 61, 63, 64, 67, 68] are aimed at eliminating the influence of a backdoor after a model is poisoned. Generally, post-processing defenses can be categorized into two types: trigger-inversion-based methods (*e.g.*, Neural Cleanse [52], I-BAU [61], SAU [57], NPD [68]) and zero-knowledge-based methods (*e.g.*, NAB [33], PDB [56]). Trigger-inversion-based methods approximate potential triggers and fine-tune the model on a clean dataset with the recovered triggers. Their effectiveness, however, relies heavily on perfect trigger recovery, which is often impractical under strong or adaptive attacks. Besides, these defenses only focus on the output logits, failing to exploit the information embedded in other poisoned layers, and thus lack comprehensive utilization of the model's internal representations. Zero-knowledge-based methods attempt to neutralize backdoors without reconstructing triggers. Early approaches fine-tune the poisoned model on clean data but achieve limited performance due to insufficient use of information embedded in the poisoned model. More recent works introduce defensive backdoors (*e.g.*, NAB, PDB) to suppress existing ones, but their effectiveness depends on the alignment between newly injected and original triggers, which may not always hold. To overcome the above limitations, we design *ATClean*, an adaptive post-processing defense. First, instead of relying on perfect trigger recovery as in trigger-inversion methods, *ATClean* only requires generating adversarial samples that induce feature collisions, thereby bypassing the impractical need for exact trigger reconstruction. Second, unlike prior defenses that focus solely on output logits, *ATClean* adaptively explores all layers of the poisoned model to locate backdoor-affected regions, ensuring a more comprehensive utilization of internal representations. Third, rather than injecting new triggers whose effectiveness depends on uncertain alignment with original ones, *ATClean* fine-tunes the model using adversarially crafted samples from the identified layers, directly targeting and mitigating the malicious features. Through this design, *ATClean* effectively removes backdoors without the weaknesses of existing post-processing approaches.

2.4 Feature Collisions

Feature collision is a phenomenon where two or more images are treated to share the same feature in the view of a model [24, 38, 45, 47]. For two different inputs x_i and x_j, where $x_i \neq x_j$, if the outputs of the k-th layer is same (*i.e.*, $f^{(k)}(x_i) = f^{(k)}(x_j)$), then we call it a feature collision. Studies have found that DNNs could be unexpectedly insensitive to adversarial samples with large magnitudes [24, 38]. Attackers can use such adversarial images to bypass identification verification based on DNN. Previous studies [45, 47] primarily emphasize the application of feature collision as a strategy for attacking, whereas our study employs it as a mechanism for defense.

Through motivation experiments below and theoretical analysis later—modeling the phenomenon as feature collisions [24]—we show that maximizing collisions between approximated and actual backdoor triggers enables effective backdoor removal, without requiring precise recovery of the original triggers.

We first train a model with one backdoor. Then we try to fine-tune it on different backdoor triggers with clean labels to see if it can

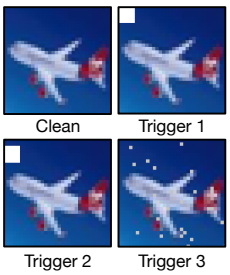

Figure 1: Backdoor Triggers 1–3.

Case	Triggers	ACC (%)	ASR (%)
1	Trigger 1	83.02	99.64
2	Triggers 1 & 2	83.50	3.00
3	Trigger 1 & Clean	74.56	98.63
4	Triggers 1 & 3	77.93	97.71

Table 1: Case 1: Backdoor attack with Trigger 1 alone. Case 2: Trigger 1 first and then Trigger 2. Case 3: Trigger 1 first and then clean samples. Case 4: Trigger 1 first and then Trigger 3. Case 2 shows the feature collision between Triggers 1 and 2.

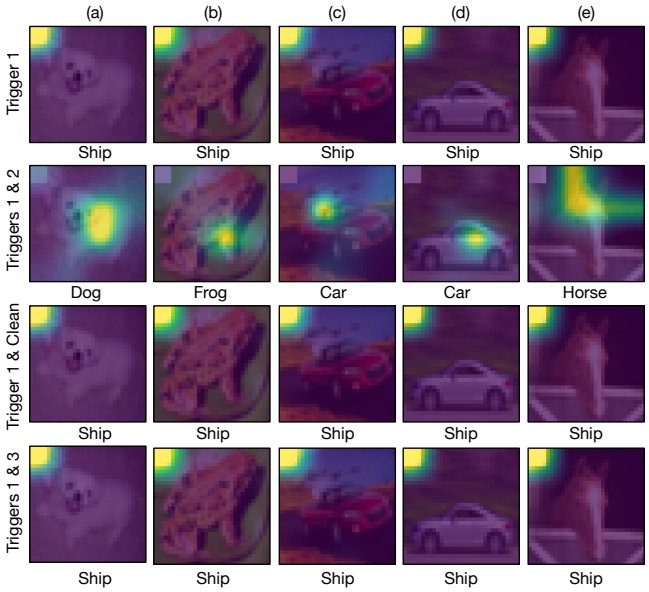

Figure 2: The original backdoor's region of interest in Grad-CAM (first row) is effectively removed by the second backdoor (second row). In contrast, fine-tuning the model on clean data or a random trigger does not affect the original backdoor, as shown in the last two rows. The caption below each image is its predicted label.

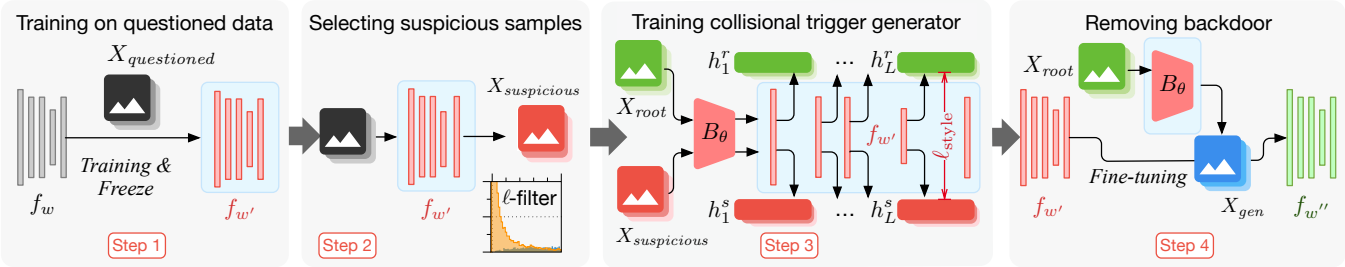

Figure 3: *ATClean*'s workflow: 1) Train a target model f_w on the questioned dataset $X_{questioned}$ to get the poisoned model $f_{w'}$; 2) Select suspicious samples $X_{suspicious}$; 3) Train a trigger generator B_θ by maximizing feature collisions between generated samples $B_\theta(X_{root})$ and suspicious samples $X_{suspicious}$; 4) Remove the learned backdoor through fine-tuning $f_{w'}$ on collisional samples X_{gen} generated by B_θ to get the clean model $f_{w''}$.

remove the existing backdoor. Specifically, we train a ResNet-18 on the CIFAR-10 dataset following the four cases presented in Figure 1 and Table 1. In Case 1, we train a poisoned model by a backdoor attack named BadNet [6] with Trigger 1. For Case 2, when the same model is trained on samples embedded with Trigger 1 (with poisoned labels) and Trigger 2 (with clean labels), the attack success rate (ASR) of the existing backdoor (3%) is significantly reduced compared to training the backdoor with Trigger 1 alone in Case 1 (99.64%), while main task accuracy (ACC) does not change much. Additionally, using Grad-CAM [46] to visualize the neuron pathway leading to the backdoor, we observe that the backdoor is effectively removed after training on the samples embedded with the second trigger, as the first two rows of Figure 2 illustrate. However, Case 3 and Case 4 do not show the same observation, indicating that clean data alone and random triggers cannot eliminate the backdoor.[3] In conclusion, some backdoor triggers can remove existing backdoors, while arbitrary new triggers or clean images cannot.

This phenomenon can be modeled as feature collisions. Let the new triggered input be x_i and the existing backdoor trigger be x_j, where $x_j \neq x_i$, and the model predictions of x_i and x_j are identical. We prove that such triggered inputs can be located by maximizing feature collisions, which is equivalent to recovering the same backdoor trigger even if the new trigger is significantly different from the original one (see formal proofs in Section A). Therefore, we can achieve optimal backdoor removal without high-quality trigger recovery by maximizing feature collisions.

The unique challenge here lies in the asymmetry between benign and poisoned features: applying feature collisions as an attack is feasible since benign features are accessible, while poisoned features are unknown and difficult to isolate. Therefore, identifying the layer where feature collisions occur is crucial. To address this, we use an attention-based generative model with adaptive feature collisions, which we prove later in Section 3.4 can automatically locate and optimize potential triggers that induce maximal feature collisions with existing backdoor triggers.

[3]Moreover, backdoor triggers of different shapes, number of pixels, and positions can produce the same effect as Case 2, as illustrated in Section C.1 of the Appendix.

3 *ATClean*'s Design

3.1 Objectives and Challenges

Objectives. We aim to achieve three objectives: 1) effectively eliminate backdoors within a model, minimizing ASR, 2) maintain high ACC, and 3) defend against potential backdoor attacks without requiring prior knowledge.

Challenges. To achieve these objectives, we face the following challenges: *First*, backdoor attacks are designed to be stealthy and well-camouflaged, making them difficult to eliminate without sufficient knowledge of the attack. *Second*, backdoor triggers are typically entangled with normal training sample features, creating a dilemma to trade off between ACC and ASR. *Lastly*, the diversity of backdoor attacks makes it non-trivial to effectively counter all types with no prior knowledge.

Even though feature collisions provide promising solutions for the aforementioned challenges, as discussed later, there are still issues to be addressed before applying them as a practical backdoor defense: 1) Due to the diverse forms of backdoor attacks, it is impractical to design heuristic backdoor triggers capable of removing backdoors via feature collisions; 2) Unlike feature-collision-based attacks, the collisional features are unknown in feature-collision-based defenses. Moreover, backdoor triggers are often entangled with benign features, making it more challenging to recover the collisional triggers. 3) The hidden layer where feature collisions take place is unknown, making it hard to retrieve such backdoor triggers via optimization.

3.2 Overview

We propose to address the aforementioned challenges through *ATClean*'s four steps, as illustrated in Figure 3:

Step 1: Training on the Questioned Dataset. We train a target model f_w on the poisoned dataset $X_{questioned}$, which contains malicious data, to obtain the backdoored model $f_{w'}$. $f_{w'}$'s model weights are then frozen for reuse in Steps 2 and 3.

Step 2: Selecting Suspicious Data Samples. The model $f_{w'}$ in Step 1 is used to select k% data samples of $X_{questioned}$ as suspicious dataset $X_{suspicious}$ using any detection method with a precision greater than 0.5. *ATClean* does not require exhaustively detecting all suspicious samples; instead, it works with any detection method

satisfying the mild precision requirement, and can seamlessly incorporate future techniques as they emerge.[4]

Step 3: Training the Collisional Trigger Generator. We train a trigger generator B_θ on the root dataset X_{root} and $X_{suspicious}$ using adaptive feature collisions and a generative model with the attention mechanism to locate and generate collisional triggers (Section 3.4), which addresses the challenge of locating the layer where feature collisions take place. B_θ is designed to learn and mimic potential collisional backdoor triggers without prior knowledge, addressing the challenges of designing heuristic triggers for different attacks. Moreover, since our generative model is conditioned on clean images, we save the trouble of extracting collisional triggers alone.

Step 4: Removing Backdoor via Feature Collisions. We generate X_{gen} with collisional features using B_θ by feeding X_{root}. Then, the backdoored model $f_{w'}$ is fine-tuned on X_{gen} to remove the backdoor via feature collisions (Section 3.5). Since the model is fine-tuned solely on clean samples, we can reduce the ASR without significantly degrading clean performance.

3.3 Step 2: Selecting Suspicious Data Samples

To effectively optimize collisional features and mitigate the backdoor, it is crucial to operate on a dataset where poisoned samples are more prevalent. Starting from the questioned dataset $X_{questioned}$, we employ a lightweight backdoor detection algorithm to extract a subset $X_{suspicious}$ containing the k% samples most likely to be poisoned. This detection step can be instantiated with any off-the-shelf backdoor detector, as long as its precision exceeds 0.5. This mild requirement ensures that a sufficient portion of poisoned samples are retained in $X_{suspicious}$, allowing our feature collision optimization to concentrate on the corrupted representations rather than being dominated by clean features. As a result, our framework significantly lowers the reliance on accurate detection, enabling robust performance even with imperfect detectors.

We conduct a sensitivity analysis on the selection of k in Section 4.4, examining how variations in the detection performance (*i.e.*, precision) of Step 2 impact the overall effectiveness of *ATClean*. Our results demonstrate that *ATClean* consistently removes the backdoor as long as the detection precision exceeds 0.5, validating our claim that *ATClean* significantly lowers the requirement on detector quality. We emphasize that although Step 2 builds on existing detection methods, our ablation studies show that applying these detectors alone for unlearning is insufficient. The success of *ATClean* stems from its end-to-end design that integrates feature collision optimization with minimal assumptions on detection quality. Moreover, we further introduce a detection-free variant of *ATClean* based on adversarial learning in Section 5, eliminating the need for Step 2.

3.4 Step 3: Training the Trigger Generator

First of all, it is hard to find a backdoor trigger that can have general feature collisions with all attacks. To address this challenge, we employ a generative model B_θ with attention mechanisms shown in Figure 4 for trigger generation. By training on the poisoned model, we can derive adaptive collisional triggers for different attacks without requiring prior knowledge. Specifically, we train a generator B_θ to craft collisional triggered samples that achieve maximal feature collisions with the backdoored samples in the poisoned target model $f_{w'}$.

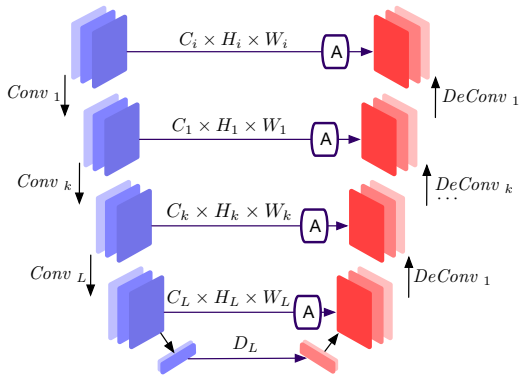

Figure 4: Structure of B_θ: 1) $Conv_k$ and $DeConv_k$ denote the k-th convolutional and deconvolutional layers, respectively; 2) A is the attention module for focusing on the trigger's position; 3) $C_k \times H_k \times W_k$ represents the output dimensions of the k-th convolutional or $(L-k)$-th deconvolutional layer; 4) The right arrow indicates skip connections to preserve benign features.

The Design of B_θ: We propose a trigger generator B_θ to maximize feature collisions as discussed above. We utilize UNet [44] as the base model to preserve benign features in generated samples. This also avoids generating random noise. We then add an attention module in the middle layer to make it capable of focusing on the backdoor trigger position, the effectiveness of which is demonstrated in ablation studies of the Section 5.

Why UNet: U-Net with attention mechanisms (Figure 4) preserves input shape due to its symmetric architecture and shape-preserving kernel design, making it adaptable to datasets with varying input dimensions. Moreover, attention mechanisms can capture position-related information in collisional backdoor triggers. In contrast, diffusion models [9] lose positional and clean feature information because of their noise-based input mechanism. Thereby, we apply UNet shown in Figure 4 instead of other popular generative models (*e.g.*, diffusion model). [5]

Another challenge here is that the hidden layer of feature collision is unknown, making it difficult to retrieve collisional triggers via optimization. We design a loss function ℓ_{locate}, that incorporates per-layer collision terms across all hidden layers. These terms are designed to adaptively emphasize layers where feature collisions are more likely to occur during optimization. By aggregating collision signals from multiple layers and allowing the optimization to focus on the most responsive ones, B_θ learns to generate collisional-triggered samples that embed backdoor patterns into clean inputs. The exponentially decaying weights in the loss function are inspired by findings in [14], which show that deeper layers typically have a greater influence on the model's output. The following proposition

[4] In Section 5, we also present an alternative *ATClean* that removes the need for Step 2 by leveraging adversarial learning without explicit detection.

[5] Conditional diffusion models require target labels, which are impractical against attacks with multiple labels (*e.g.*, all-to-all attacks [57]).

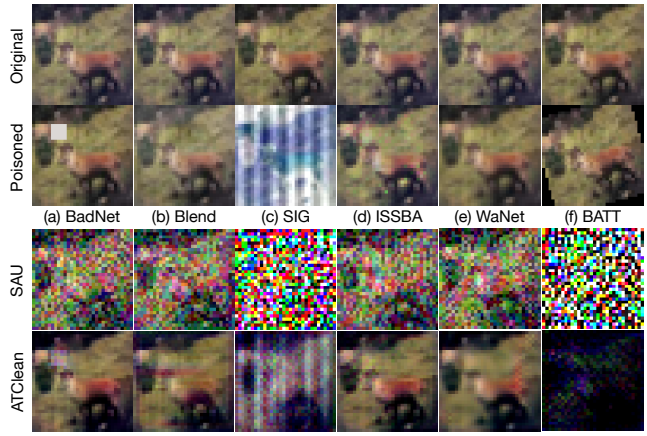

Figure 5: Original, poisoned, and generated images from CIFAR-10. Row 1 presents the original image. Row 2 shows the images poisoned by six attacks (a)-(f), respectively. Rows 3 and 4 show images generated by SAU [57] and *ATClean* with collisional backdoor features for Row 2, respectively.

of the definition of feature collisions proves that optimizing l_{locate} is a feasible method to address the challenge of locating the feature collision layer.

Proposition 1. *Minimizing ℓ_{locate} is equivalent to locating the target layer and maximizing feature collisions. The definition of l_{locate} is as follows:*

$$l_{locate} = \frac{1}{4N_L^2 M_L^2} \sum_k 0.1^{k-1} \sum_{i,j} (h^r_{L,k/i,j} - h^s_{L,k/i,j})^2, \tag{1}$$

where $h^r_{L,k} \in \mathbb{R}^{N_L \times M_L}$ and $h^s_{L,k} \in \mathbb{R}^{N_L \times M_L}$ are the feature maps from the k-th last convolutional layer of the backdoored model when encoding the root data X_{root} and the suspicious data $X_{suspicious}$, respectively. Here, N_L and M_L denote the number of channels and the spatial dimension of the feature maps. Moreover, in practice, we characterize the distance using distributional measures (e.g., Maximum Mean Discrepancy (MMD) loss), without requiring each suspicious sample to align with its corresponding sample.

Proof. If $\ell_{locate} \approx 0$, we have $f^{(L)}_{w'}(B_\theta(x_i)) \approx f^{(L)}_{w'}(x_j)$ for $(x_i, y_i) \in X_{root}$ and $(x_j, y_j) \in X_{suspicious}$, where $f^{(L)}$ is the output of the L-th layer ($1 \le L \le M$, where M is the number of layers). Since $B_\theta(x_i) \neq x_j$, setting $\ell_{locate} = 0$ aligns with the definition of feature collisions. *Thus, minimizing ℓ_{locate} is equivalent to automatically locating all possible target layers (i.e., L, where $1 \le L \le M$) and maximizing feature collisions.*

Moreover, to maximize the feature collisions while maintaining the content representation of the original clean sample $(x_i, y_i) \in X_{root}$ to avoid further reducing performance on clean samples, we add a regularization loss term ℓ_{MSE}:

$$\ell_{MSE} = \frac{1}{|X_{root}|} \sum_{(x_i, y_i) \in X_{root}} (x_i - B_\theta(x_i))^2. \tag{2}$$

The total loss function is finalized as:

$$\ell_{total} = \ell_{MSE} + \alpha * \ell_{locate}. \tag{3}$$

Figure 5 shows the images generated by B_θ. For BadNet, the generated image recovers the square trigger's position, shape, and size. For Blend, B_θ recovers the blended car image's contour. For SIG and ISSBA, the attack pattern is recovered. For WaNet, the blurring transformation is restored. For BATT, the rotation transformation is not fully recovered, yet the generated image appears at a similar angle. We further investigate the extent of feature collision and facilitate a more quantitative discussion as discussed in Section 4.2. By comparing the cosine similarity of output logits between generated data and ground-truth poisoned samples, we find that *ATClean* significantly amplifies the feature collisions.

3.5 Step 4: Removing Backdoor

We construct a generated dataset X_{gen} from the root data X_{root} using the trigger generator B_θ from the previous Section:

$$X_{gen} = \{(B_\theta(x_i), y_i) \mid (x_i, y_i) \in X_{root}\}, \tag{4}$$

where X_{root} consists of root images and their corresponding labels. Next, we train the poisoned model $f_{w'}$ on the generated dataset X_{gen} to get a repaired model. We then present the following proposition to establish the relationship between training on X_{gen} and minimizing the backdoor risk l_{risk} defined below.

Proposition 2. *If there exist feature collisions[6] between $B_\theta(x_i)$ for $(x_i, y_i) \in X_{root}$ and x_j for $(x_j, y_j) \in X_{suspicious}$, fine-tuning the poisoned model $f_{w'}$ on X_{gen} is same as minimizing l_{risk} defined below which measures the effectiveness of the backdoor attack and failure of the model on clean samples:*

$$\ell_{risk} := \frac{1}{|X_{test}|} \sum_{i=1}^{|X_{test}|} \mathcal{I}(f_w(x_i) \neq y_i) + \frac{1}{|X^{-c}_{test}|} \sum_{j=1}^{|X^{-c}_{test}|} \mathcal{I}(f_w(b(x_j, p)) = c), \tag{5}$$

where $\mathcal{I}$ is the indicator function, X_{test} is the test dataset, X_{root} is the root dataset which is collected by the defender and does not overlap with X_{test}, X^{-c}_{test} denotes the subset of X_{test} containing samples with labels other than c, $(x_i, y_i) \in X_{test}$, $(x_j, y_j) \in X^{-c}_{test}$, $b(x_j, p)$ is the function to generate malicious samples in backdoor attacks, and c is backdoor attacks' target label.

Since optimizing ℓ_{risk} can reduce ASR while maintaining a high ACC, according to Proposition 2, the backdoor can be effectively removed by training $f_{w'}$ on X_{gen} we construct in Step 4.[7]

4 Evaluation

Our evaluation examines *ATClean*'s effectiveness in removing backdoors, verification of our generated samples maximize feature collisions and the performance of *ATClean* does not depend on perfect approximation of backdoor triggers, sensitivity to factors (*e.g.*, detection performance, different poisoning ratios, and root data sizes), robustness against adaptive attacks, and ablation studies on each component of *ATClean*.

[6] The existence of feature collisions is proved in the Appendix.
[7] The proof of the above proposition is provided in Section A of the Appendix.

ATClean achieves the SOTA defense effectiveness and by far the lowest performance drop on clean data among successful defenses with approximately a 20% improvement of DER, which measures the accuracy-defense trade-off.

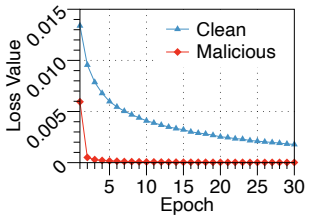

Figure 6: Average loss comparison of training ResNet-18 on CIFAR-10 between clean samples and malicious samples backdoored by BadNet.

4.1 Experimental Setup

Models and Datasets. We evaluate *ATClean* with ResNet-18 [19] on two popular datasets: CIFAR-10 and Tiny ImageNet [23]. We also evaluate *ATClean* with another representative model structure T2T-ViT on CIFAR-10.

Backdoor Attacks. Seven iconic SOTA backdoor attacks are considered: 1) *visible trigger attack* using unrelated patterns (BadNet [15]), 2) *multi-label attack* using All to All attacks (BadNet-A2A [57]), 3) *invisible trigger attack* by blending the trigger with clean samples (Blend [6]), 4) *clean label attack* with a frequency domain trigger (SIG [3]), 5) *sample-specific invisible trigger attack* generated through optimization (ISSBA [26]), and 6) *sample-specific invisible trigger attack* created by manipulating the geometric structure of original images (WaNet [36] and BATT [59]). All attacks poison 10% of the training data, except for SIG on Tiny ImageNet, where we poison only 0.5% of the training data in this scenario.[8] We also evaluate defend effectiveness of *ATClean* with potential adaptive attacks in Section 4.5 and BELT attacks in Section 4.3 (Additional details of attacks in Section E.1).

Backdoor Defenses. We compare *ATClean* with seven representative SOTA backdoor defense methods: 1) ABL [27] minimizes updates of detected poisoned samples during the training and applies inverse updates on such samples to remove backdoor after convergence of the model; 2) NAB [33] introduces a new backdoor on a poisoned model by training it on detected poisoned samples and rejecting inputs that produce different outputs before and after applying the new backdoor trigger during inference; 3) FT [31] fine-tunes poisoned model on clean samples to mitigate the backdoor; 4) i-BAU [61] applies adversarial training to improve the robustness of a poisoned model; 5) SAU [57] uses shared adversarial unlearning to break the relationship between backdoor triggers and target labels; 6) NPD [68] applies a plugin neural network trained by adversarial learning to filter poisoned feature during the inference of a poisoned model; 7) PDB [56] incorporate a defensive backdoor during the training to suppress effects of potential backdoor attacks. All defenses have access to 10% of benign training data on CIFAR-10 and 5% on Tiny ImageNet, respectively. For *ATClean*, we apply ABL [27], which assumes inputs with lower loss values as poisoned, as shown in Figure 6. *ℓ-filter* selects 1% of data as $X_{suspicious}$ (*i.e.*, $k = 1$) in Step 2, and we set $\alpha = 0.1$ for CIFAR-10 and $\alpha = 10$ for Tiny ImageNet in Step 3. Further details on the defense are provided in the Supplementary Material E.2. Moreover, comparisons with SAU and i-BAU can be viewed as ablation studies on the effectiveness of feature collisions in post-processing defenses.

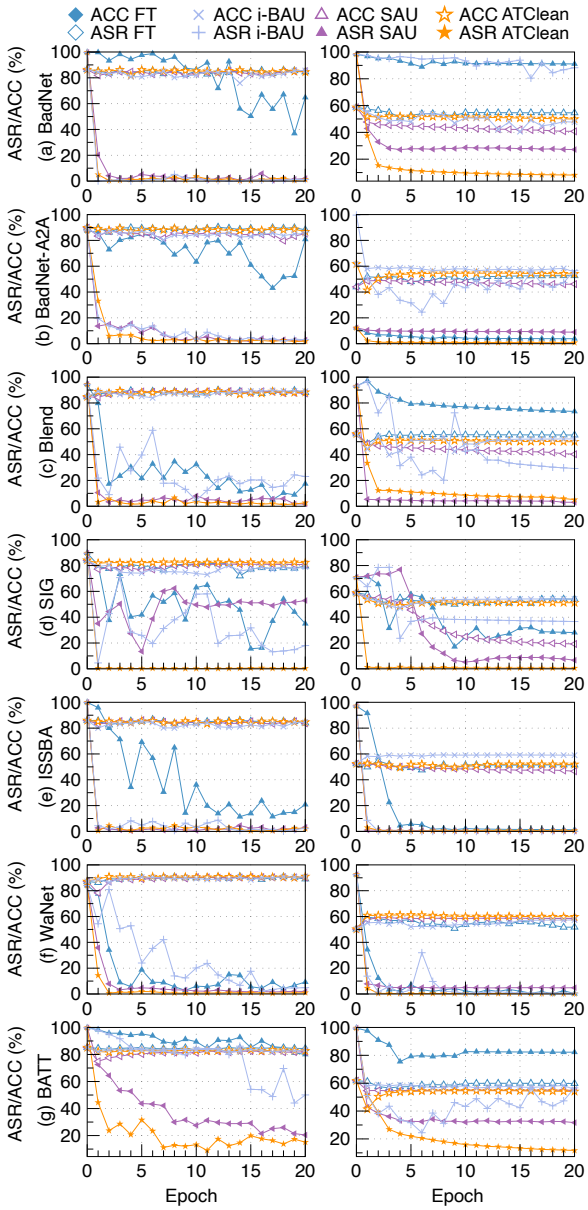

Figure 7: Fine-tuning curves of different defenses on CIFAR-10 (left) and Tiny ImageNet (right) using ResNet-18 trained with 10% poisoned data. All attacks are extended to 20 rounds. Beyond the end-to-end results, this further demonstrates the superiority and robustness of *ATClean* throughout the entire defense process.

Metrics. We use four metrics to evaluate defenses' effectiveness: ACC, ASR, DER, and R-ACC.[9] DER is defined as $[\max\{0, \Delta\text{ASR}\} -$

[8] The clean-label attack SIG uses only 0.5% poisoned data on Tiny ImageNet due to insufficient samples per class to poison 10% of the dataset. This includes all samples from the target label.

[9] ACC: Main task accuracy, *i.e.*, the performance of the target model on clean data; ASR: Attack success rate, *i.e.*, the ratio of poisoned samples classified as the target label; DER: Defense effectiveness rating, *i.e.*, the trade-off measurement between ACC and

Table 2: Effectiveness (%) of various defenses on a poisoned ResNet-18 trained on CIFAR-10 (Tiny ImageNet) with 10% poisoned data. Heatmaps visualize key metrics: ASR, ACC, and DER, for easy comparison: values closer to 0 for ASR, and closer to 100 for ACC and DER, are shaded yellow, indicating better performance; less optimal values shift toward cyan. "BN" is short for BadNet.

Defenses	No Defense						FT						ABL					
Attacks	ASR		ACC		DER		ASR		ACC		DER		ASR		ACC		DER	
BN	99.7	(98.4)	86.4	(58.7)	50.0	(50.0)	64.8	(91.1)	84.6	(54.6)	66.6	(51.6)	0.0	(0.1)	53.8	(32.5)	83.2	(73.5)
BN-A2A	87.4	(12.3)	89.8	(44.0)	50.0	(50.0)	80.8	(3.9)	88.2	(52.7)	52.5	(54.2)	0.0	(1.3)	34.0	(30.9)	74.3	(49.0)
Blend	94.5	(93.1)	84.8	(56.1)	50.0	(50.0)	17.4	(73.5)	88.2	(55.2)	88.5	(59.4)	0.0	(8.5)	54.9	(28.2)	83.8	(78.3)
SIG	89.6	(70.9)	84.1	(58.8)	50.0	(50.0)	35.0	(28.1)	79.6	(54.0)	75.1	(69.0)	0.2	(0.8)	34.4	(15.2)	69.8	(63.2)
ISSBA	99.9	(97.2)	85.9	(52.5)	50.0	(50.0)	20.8	(1.3)	84.0	(51.5)	88.6	(97.5)	0.0	(0.5)	39.1	(30.0)	76.6	(87.6)
WaNet	84.0	(92.6)	87.2	(50.3)	50.0	(50.0)	9.2	(0.2)	89.1	(51.8)	87.4	(96.1)	0.0	(19.1)	51.6	(39.5)	74.2	(81.3)
BATT	84.1	(99.6)	89.6	(61.8)	50.0	(50.0)	80.1	(82.3)	84.4	(59.8)	59.5	(57.7)	0.0	(0.1)	34.0	(13.5)	73.5	(75.0)
Defenses	i-BAU						SAU						*ATClean* (ours)					
Attacks	ASR		ACC		DER		ASR		ACC		DER		ASR		ACC		DER	
BN	0.4	(95.9)	80.0	(48.0)	96.9	(45.9)	2.0	(27.2)	85.1	(40.8)	98.2	(76.2)	0.3	(8.8)	86.0	(50.9)	99.5	(90.9)
BN-A2A	11.4	(1.5)	87.3	(40.5)	86.8	(53.6)	2.8	(8.9)	85.2	(46.2)	90.0	(51.7)	1.9	(0.8)	86.7	(47.6)	91.2	(55.7)
Blend	39.7	(38.0)	85.1	(51.2)	77.4	(75.1)	1.8	(3.1)	87.4	(40.3)	96.3	(87.1)	1.6	(7.3)	86.8	(50.1)	96.4	(89.9)
SIG	25.9	(36.7)	74.1	(50.0)	76.8	(62.7)	52.9	(6.7)	80.8	(19.3)	66.7	(62.3)	0.4	(0.6)	82.1	(51.2)	93.6	(81.4)
ISSBA	4.9	(0.0)	85.1	(59.1)	97.1	(98.6)	3.2	(0.3)	83.7	(46.6)	97.3	(95.5)	1.6	(0.7)	85.7	(52.0)	99.1	(98.0)
WaNet	24.0	(3.1)	90.6	(52.1)	80.0	(94.6)	1.9	(4.8)	90.2	(58.4)	91.0	(93.8)	0.8	(0.3)	90.7	(60.1)	91.6	(96.1)
BATT	84.0	(31.6)	83.6	(58.7)	57.2	(82.4)	20.4	(31.7)	81.9	(54.8)	88.1	(80.5)	15.1	(13.9)	82.8	(54.6)	91.2	(89.3)

Table 3: Effectiveness (%) of various defenses on a poisoned T2T-ViT trained on CIFAR-10 with 10% poisoned data. Heatmaps visualize key metrics: ASR, ACC, and DER, for easy comparison: values closer to 0 for ASR, and closer to 100 for ACC and DER, are shaded yellow, indicating better performance; less optimal values shift toward cyan. "BN" is short for BadNet.

Defenses	No Defense			FT			ABL			i-BAU			SAU			*ATClean* (ours)		
Attacks	ASR	ACC	DER	ASR	ACC	DER	ASR	ACC	DER	ASR	ACC	DER	ASR	ACC	DER	ASR	ACC	DER
BN	81.6	76.4	50.0	69.1	78.7	56.3	0.0	56.1	81.2	9.8	77.2	85.9	18.3	75.3	81.1	2.2	75.3	89.2
Blend	75.8	63.2	50.0	72.7	64.8	52.1	9.1	45.7	74.5	70.4	63.6	52.7	21.6	56.8	73.8	9.4	71.1	83.2
SIG	98.6	60.4	50.0	62.9	54.5	67.2	7.7	30.7	80.3	59.9	53.6	59.4	4.2	39.4	80.2	5.2	55.1	94.1
ISSBA	85.8	70.2	50.0	78.8	69.2	52.9	30.0	48.7	66.9	78.0	67.9	52.5	1.1	60.5	87.3	3.8	65.9	87.4
WaNet	72.9	42.7	50.0	65.1	43.6	53.9	18.0	49.2	77.5	60.7	48.4	56.1	26.7	63.7	73.1	5.2	64.4	83.3
BATT	85.9	71.5	50.0	85.3	71.3	50.2	4.9	47.5	78.5	83.1	72.5	51.4	0.4	53.2	83.4	6.4	66.5	87.3

Table 4: Defense effectiveness (%) of latest post-processing defenses against three attacks on Tiny ImageNet: each cell presents results as ASR/ACC/DER ("No def." indicates no defense).

	BadNet	Blend	SIG
No def.	98.36/58.74/50.0	93.06/56.07/50.0	70.87/58.81/50.0
NAB	19.04/39.51/80.05	16.66/42.91/81.62	69.87/**53.53**/47.86
NPD	11.46/40.63/84.39	**3.94**/44.81/88.93	1.43/37.36/74.00
PDB	24.5/**51.9**/83.51	8.21/50.09/89.44	1.04/50.90/80.56
ATClean	**8.81**/50.90/**90.86**	7.25/**50.10**/**89.92**	**0.60**/51.20/**81.30**

Table 5: Defense effectiveness (%) of latest post-processing defenses against three attacks on Tiny ImageNet: each cell presents results as ASR/ACC/R-ACC ("No def." indicates no defense).

	BadNet	Blend	SIG
No def.	98.36/58.74/1.64	93.06/56.07/6.94	70.87/58.81/0.0
NAB	19.04/39.51/0.0	16.66/42.91/0.0	69.87/**53.53**/6.90
NPD	11.46/40.63/31.02	**3.94**/44.81/34.10	1.43/37.36/33.90
PDB	24.5/**51.9**/47.78	8.21/50.09/43.36	1.04/50.90/47.89
ATClean	**8.81**/50.90/**47.86**	7.25/**50.10**/**43.52**	**0.60**/51.20/**47.90**

$\max\{0, \Delta\text{ACC}\}+1]/2$, where $\Delta\text{ASR} \in [-1, 1]$, $\Delta\text{ACC} \in [-1, 1]$, and thereby DER $\in [0, 1]$. ΔASR and ΔACC indicate the changes before and after applying defense methods in ASR and ACC, respectively. Intuitively, a low ASR with a high ACC results in a high DER, indicating that the defense effectively removes backdoor attacks without degrading (or even improving) the main task accuracy.

4.2 Key Results

Defense Effectiveness of *ATClean*. Table 2 shows that: 1) *ATClean* achieves the highest DER in all seven attacks for CIFAR-10 and ranks highest DER in six out of the seven attacks for Tiny ImageNet, demonstrating the best balance between reducing ASR and maintaining ACC; 2) *ATClean* achieves the second-lowest ASR values across all seven attacks on CIFAR-10 and ranks among the Top 2 for the lowest ASR in five out of seven attacks on Tiny ImageNet. Although ABL achieves the lowest ASR across nearly all attacks for both datasets, it incurs an ACC drop exceeding 20%. To demonstrate the effectiveness of *ATClean* on an alternative model structure besides ResNet-18, we also present experimental results on T2T-ViT [34] using CIFAR-10. T2T-ViT applies a vision transformer encoder [1], which is widely used in SOTA DNNs, for image classification and has been used on the backdoor defense papers [63] recently as a representative model. As shown in Table 3,

ASR [57, 67, 68]; R-ACC: Recovered Accuracy, *i.e.*, the accuracy of the target model on poisoned samples [57, 67, 68].

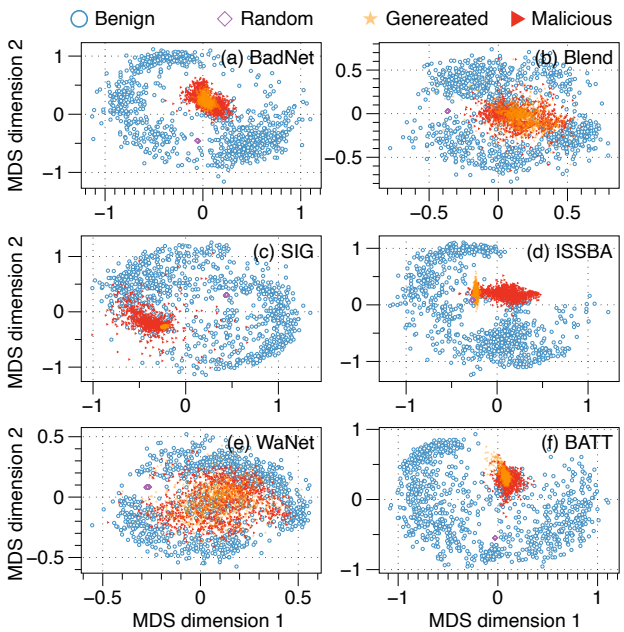

Figure 8: 2D-MDS projection of logits on the CIFAR-10 test dataset: 1) Benign samples are the original clean samples; 2) Random samples are generated from clean samples using a randomly initialized B_θ; 3) Generated samples are derived from clean samples and B_θ learned in Step 3 of *ATClean*; 4) Malicious samples are created from clean samples poisoned with the malicious trigger. *ATClean* better reconstructs the ground-truth poisoned features and significantly improves feature collision.

Table 6: DoF across different attacks: 1) Random stands for the case where B_θ is randomly initialized; 2) *ATClean* stands for the case where Step 3 of *ATClean* trains B_θ. *ATClean* achieves a higher quantitative measure of feature collision.

	BadNet	Blend	SIG	ISSBA	WaNet	BATT
Random	0.47	0.47	0.33	0.53	0.63	0.35
ATClean	**0.77**	**0.92**	**0.76**	**0.86**	**0.82**	**0.86**

ATClean achieves the highest DER against all six SOTA attacks using T2T-ViT, demonstrating its effectiveness across different model architectures.

Table 4 demonstrates that *ATClean* not only achieves the highest DER, compared to latest post-processing defenses, such as NAB [33], NPD [68], and PDB [56], but also attains the lowest ASR in three out of four attacks. This highlights its ability to provide a comparable level of security while maintaining superior model performance. Moreover, as demonstrated in Table 5, *ATClean* also achieves the highest R-ACC, indicating *ATClean*'s superiority in recovering incorrect predictions on poisoned samples. In summary, compared to the latest SOTA defenses with partially similar motivations, *ATClean* exhibits clear superiority. This is because *ATClean* is a layerwise defense with finer granularity, which allows it to achieve better protection against the attack in more scenarios.

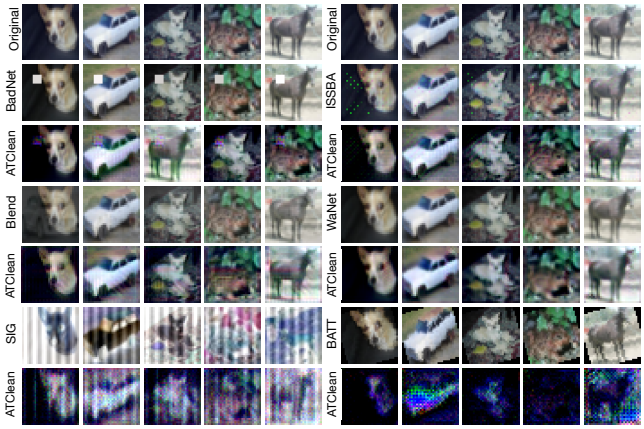

Figure 9: Original, poisoned, and generated images from CIFAR-10. Row 1 represents original images, Rows 2,4,6 stand for poisoned images, and Rows 3,5,7 are generated images. This shows that *ATClean* achieves a better reconstruction of the backdoor trigger.

Table 7: Defense effectiveness (%) of our method against the BELT attack with ResNet-18 on Tiny ImageNet. Each cell is presented as ACC/ASR.

Attacks/Defenses	No Defense	MOTH [48]	*ATClean* (ours)
BELT+BadNet	97.83/54.37	80.98/39.95	**10.13**/51.92
BELT+Blend	92.89/57.63	92.83/56.74	**8.93**/52.31
BELT+ISSBA	98.88/53.98	74.83/52.78	**0.78**/52.49

Figure 7 further examines the unlearning dynamics of various defense methods over 20 rounds, which reveal that, for *ATClean*, the ASR exhibits the most rapid decline as the number of epochs increases across nearly all attacks while maintaining an ACC comparable to fine-tuning on the clean dataset, in contrast to the two adversarial training methods, i-BAU and SAU. This is because the finer-grained defense makes the overall robustness stronger, preventing performance fluctuations caused by misclassification.

Visualization and Quantitative Analysis of Feature Collisions. We visualize the logits of different samples by Multi-dimensional scaling (MDS) [49] which maps logits into 2D-samples. Figure 8 shows that *ATClean* maps clean samples to a space where their logits overlap with those of malicious samples, demonstrating that *ATClean* successfully induces feature collisions with the injected backdoor, effectively aiding in the backdoor's neutralization in Step 4. For quantitative analysis, we define the degree of feature collisions (DoF) as:

$$DoF := \sum_{x_i \in X_{test}} \frac{f_{w'}(b(x_i, p)) \cdot f_{w'}(B_\theta(x_i))}{|X_{test}| \|\mathbf{f_{w'}}(\mathbf{b(x_i, p)})\| \|\mathbf{f_{w'}}(\mathbf{B_\theta(x_i)})\|}, \tag{6}$$

where X_{test}, which is the test dataset. A higher Dof indicates more feature collisions. Table 6 shows that *ATClean* evidently improves feature collisions between generated samples and poisoned samples. Moreover, Figure 5 and Figure 9 shows that B_θ effectively captures the pattern of the backdoor trigger. For example, in the case of BadNet, B_θ identifies both the shape and position of the backdoor trigger. In contrast, SAU generates only global noise, failing to

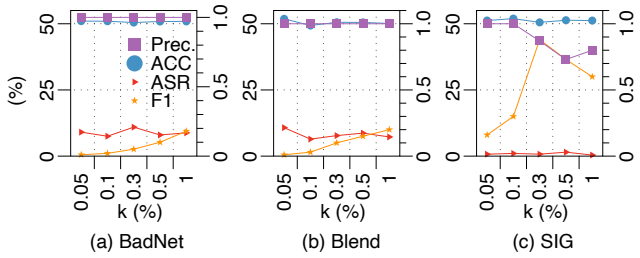

Figure 10: Effectiveness (%) of *ATClean* under varying selections of k on Tiny ImageNet across different attacks. F1 and Prec. are F1-score and Precision of Step 2.

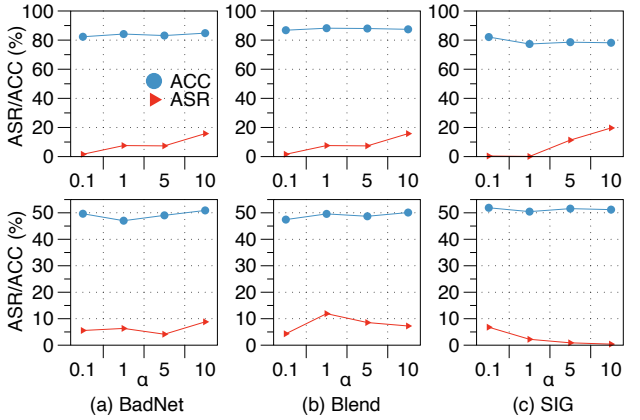

Figure 11: Effectiveness (%) of *ATClean* under varying selections of α on CIFAR-10 (top) and Tiny ImageNet (bottom) across different attacks.

Table 8: Effectiveness (%) of *ATClean* on CIFAR-10 with different poisoning ratios.

Poisoning Ratios		10%	30%	50%
Attacks	**Metrics**	No defense (*ATClean*)	No defense (*ATClean*)	No defense (*ATClean*)
BadNet	ASR	99.71 (0.34)	99.86 (3.88)	99.98 (3.32)
	ACC	86.35 (85.98)	86.83 (88.53)	88.08 (85.66)
ISSBA	ASR	99.99 (1.59)	99.99 (1.40)	99.98 (1.43)
	ACC	85.94 (85.65)	88.03 (90.78)	85.41 (89.77)

capture the trigger's position or shape. This suggests that trigger collisions can even be achieved directly on RGB images for some attacks, a capability that SOTA methods lack. Last but not least, even though we do not achieve perfect trigger recovery, *ATClean* still has the highest DER, indicating our methods are not so sensitive to perfect backdoor trigger recovery as previous works.

Table 9: Effectiveness of *ATClean* on CIFAR-10 with different root data sizes (root ratios).

Root Ratios	1%	3%	5%	7%
Attacks	ASR/ACC	ASR/ACC	ASR/ACC	ASR/ACC
BadNet	1.4/83.4	0.6/86.3	0.7/87.8	1.6/85.2
ISSBA	2.1/82.6	1.8/85.1	2.8/84.6	1.6/85.2

Table 10: Defense effectiveness (%) of our method against the adaptive BadNet attack on Tiny ImageNet.

$\|\|\delta\|\|$	No Defense	SAU	*ATClean* (ours)
1	70.99/51.92	80.98/39.95	**4.29**/45.82
5	99.09/51.33	51.14/40.83	**3.12**/49.19
10	98.16/53.03	34.12/46.2	**1.15**/50.38

Table 11: Ablation studies of *ATClean* on Tiny ImageNet: 1) In "No Step 3", B_θ is replaced with a randomly initialized U-Net in Step 4; 2) In "No Step 4", X_{gen} is replaced with the root dataset in Step 4.

	Metrics	BadNet	Blend	SIG
No defense	ASR	98.36	93.06	70.87
	ACC	58.74	56.07	58.81
No Step 3	ASR	98.09	58.93	28.88
	ACC	52.66	51.96	50.96
No Step 4	ASR	98.92	91.53	58.33
	ACC	57.03	55.22	58.26
ATClean	ASR	**8.81**	**7.25**	**0.60**
	ACC	50.90	50.10	51.20

4.3 Resistance to BELT Attacks

The exclusivity enhancement mechanism of BELT [42] is specifically designed to prevent trigger inversion, and evaluating the effectiveness of Step 3 against BELT could further validate the robustness of the defense.[10] Table 7 shows that *ATClean* is effective against such attacks: *ATClean* reduces ASR from around 97.8% to 10.1%, which further proves the robustness of *ATClean*. This is because *ATClean* is an adaptive algorithm, enabling it to defend even against such specialized, tailored attacks.

4.4 Sensitivity Test

Sensitivity of Normal Sample Filter Threshold k (detection performance). We conduct a sensitivity analysis of normal sample filter threshold k, which is the ratio of data selected as $X_{suspicious}$ in Step 2. ASR in Figure 10 demonstrates that *ATClean* remains consistently effective across different k values even when F1-score is low. Precision in Figure 10 also indicates that lower k values result in a higher proportion of poisoned samples, facilitating training B_θ to maximize feature collisions with the backdoor.

Sensitivity of Loss Weighting Coefficient α. In Step 3, we use a weighting coefficient α to balance the contributions of two loss

[10] BELT-specific regularization is omitted to keep the evaluation consistent with the defender-controlled assumption in our threat model.

terms. We now assess the sensitivity of α, with $\{0.1, 1, 5, 10\}$ as candidate values. Figure 11 shows that *ATClean* effectively reduces ASR while maintaining high ACC across all α values. For smaller images with fewer pixels, such as those in CIFAR-10, lower α values result in lower ASR, whereas for larger images with more pixels, such as those in Tiny ImageNet, higher α values are more effective. This is because a higher α introduces more manipulations to RGB image pixels, which are more suitable for datasets with larger figures. To streamline our approach, we set $\alpha = 0.1$ for CIFAR-10 and $\alpha = 10$ for Tiny ImageNet, respectively.

Influence of Different Poisoning Ratios and Sizes of Root Dataset X_{root}. To examine the impact of the poisoning ratio, we evaluate *ATClean* with poisoning ratios ranging from 10% to 50%. Table 8 shows that *ATClean* remains effective even when half of the data is poisoned, both for global attacks (*e.g.*, BadNet) and sample-specific attacks (*e.g.*, ISSBA). Table 9 indicates *ATClean* is effective even with only 1% clean samples with acceptable degrade on performance.

4.5 Resistance to Potential Adaptive Attacks

An adversary may attempt to generate a more stealthy backdoor that cannot be triggered by triggers with similar features to break *ATClean*. We simulate such attacks by designing the following adaptive attack framework, where the attacker constructs the poisoned data as: $D_{poisoned} = \{(b(x_i, p), c) | (x_i, y_i) \in X_{train}\} \cup \{(b(x_i, p) + \delta, y_i) | (x_i, y_i) \in X_{train}\}$, where $b(\cdot, p)$ is the backdoor function and δ is a random noise generated via Gaussian distribution. Table 10 shows that *ATClean* is still effective even under the adaptive version of BadNet under this framework by achieving a DER above 80%. Increasing $||\delta||$ boosts ASR for adaptive attacks but reduces stealth, making them easier for *ATClean* to remove (Tiny ImageNet, ResNet-18). Moreover, this adaptive attack effectively bypasses other trigger-inversion defenses (*e.g.*, SAU) with a better performance than BadNet (Table 2 of our *ATClean* submission). Thereby, we conclude that our adaptive attack is effectively developed and *ATClean* can successfully defend against it. WaNet can also be considered one adaptive attack under this framework, and *ATClean* outperforms previous methods as illustrated in Table 2.

4.6 Ablation Studies

We examine each component of *ATClean* through ablation studies. *ATClean* consists of two parts: training the trigger generator (denoted as Step 3) and removing the backdoor via feature collisions (denoted as Step 4). To demonstrate the effectiveness of each component, we conduct ablation studies for each part. Table 11 shows that "Randomly Weighted B_θ" cannot substitute for Step 3 (No Step 3), and "Fine-tuning on clean samples" cannot substitute for Step 4 (No Step 4), as *ATClean* achieves a significantly lower ASR.

Our Contributions are Beyond Step 2. As shown in both Table 2 and Figure 12, *ATClean* achieves a higher trade-off between ACC and ASR throughout the fine-tuning compared with ABL, which uses the same detection method in Step 2. Furthermore, Figure 13 shows that even $X_{suspicious}$ contains clean samples in our experiments of Table 2, *ATClean* still can successfully remove backdoors.

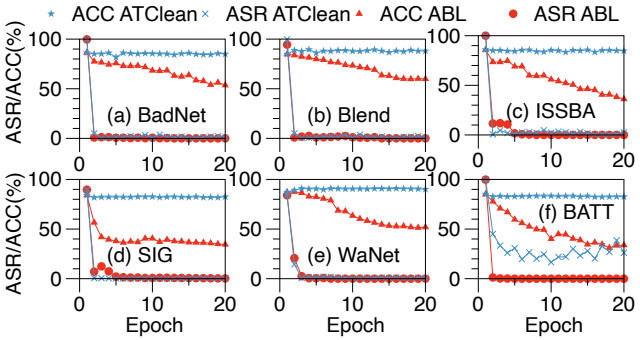

Figure 12: Fine-tuning curves of *ATClean* and ABL on Tiny-ImageNet using ResNet-18. This highlights the advantage of *ATClean* compared to directly relying on a detector.

Table 12: Comparison of training requirements and computational cost. Re-train indicates that the method requires retraining the model from scratch, which incurs substantially higher cost than Fine-tune, where the primary overhead comes from model fine-tuning. Trigger Recovery denotes that the method involves optimizing the trigger, with its detailed overhead reported in a separate table. Pre-processing refers to input-level defenses, while Post-processing refers to prior post-processing defenses.

Defenses	Re-train	Fine-tune	Trigger Recovery	Cost
Pre-processing	✓	✗	✗	High
Post-precessing	✗	✓	✓ & ✗	Medium
ATClean	✗	✓	✓	Medium

Table 13: Training overhead (s) of different post-processing defenses. This shows that, compared to prior works, the overhead of *ATClean* is entirely acceptable.

Defenses	i-BAU	SAU	NPD	*ATClean* (ours)
Overhead (s)	**43.69**	726.04	869.34	**154.82**

Moreover, previous trigger recovery-based defenses (*e.g.*, SAU) cannot be combined with detection methods like Step 2 directly, as they cannot be conditioned on the poisoned input and they need the clean label of poisoned samples. This observation highlights that our contributions are beyond Step 2.

4.7 Training Overhead Analysis

In this section, we show that ATClean incurs lower or comparable computational overhead compared to both post-processing and pre-processing SOTA defenses. 1) Specifically, although ATClean trains a U-Net-based generator, this cost is modest because the generator is trained only once before fine-tuning in Step 4. In contrast, as shown in Table 13, post-processing methods such as SAU and NPD repeatedly optimize perturbations or polarizers in each fine-tuning round, resulting in higher overall time costs despite using simpler architectures. Furthermore, the initial poisoned model training time is common to all methods in this setting and is not additional overhead for ATClean. 2) Compared to pre-processing defenses, which typically require retraining the entire model from scratch

on a cleaned dataset, ATClean avoids such costly retraining. To further reduce overhead, ATClean can adopt a lighter U-Net-based generator with fewer layers without changing the defense pipeline. The finer-grained defense is precisely the underlying reason.

5 Discussion and Clarifications

We further discuss *ATClean*'s design decision in this section:

Q: What if the detection is failed? This subsection aims to address concerns about scenarios where the detection module in Step 2 fails (*i.e.*, precision < 1.0). First, we included a clean label attack (*i.e.*, SIG [3]) in our experiments, which presents a challenging backdoor design where loss-based detection methods fail. As shown in Figure 10, SIG breaks the lower-loss assumption of ABL, which assumes that poison samples have lower loss values; yet, our method remains effective in this setting, demonstrating robustness to the challenges posed by methods such as Narcissus [62] that can bypass loss-based detections. Second, we also note that our approach is not inherently tied to loss-based detection; as stated previously in the paper, the loss-based detector can be replaced with any alternative detection method. The core contribution of our work lies in leveraging feature collisions, and the inclusion of SIG in our evaluation already addresses the critical failure mode of the detection module.

Q: What if the detection is infeasible? In our previous *ATClean* framework, the trigger generator B_θ is trained using both root data X_{root} and suspicious data $X_{suspicious}$ to maximize feature collisions in intermediate layers of the poisoned target model $f_w^{'}$. In this section, we remove the dependency on $X_{suspicious}$ entirely, enabling applicability in *black-box* settings where collecting poisoned samples is infeasible (*e.g.*, when defenders only have access to the poisoned model). Instead of relying on real suspicious data, we adopt an adversarial learning paradigm, where B_θ generates collision-inducing inputs directly from X_{root}. The generator is optimized to produce inputs that minimize raw data distance while maximizing divergence in intermediate representations. This enables the model's feature space to form concentrated clusters that mimic backdoor-triggered behavior, without knowing the actual trigger or having access to poisoned samples. The backdoored model is then fine-tuned on these adversarially generated examples to remove the malicious behavior. Table 14 shows that this design preserves the core strength of the collision-based purification while improving generalization to unseen attack types and black-box scenarios, further addressing the concerns of ATClean's dependence on Step 2. We also provide a truly data-free variant and analysis demonstrating how the detector-free variant generated samples capture sample-specific triggers in Sections C.3 and C.7 of the supplementary material.

Q: Why we need the attention module in trigger generator? The attention blocks let the generator preserve local spatial cues and emphasize collision-sensitive regions, which a vanilla U-Net tends to overlook; this makes the reconstructed trigger sharper and more faithful. In our ablation on BadNet with ResNet-18, removing attention modules causes the post-defense ASR to remain around 20%, whereas the full model with attention reduces ASR substantially further (*i.e.*, 0.3%).

Table 14: Effectiveness (%) of *ATClean* without the dependence on Step 2. Results are collected on Tiny ImageNet. "w/o" is the detector-free variant of *ATClean*. "No" is the result without any defense. This shows that the detector-free version of *ATClean* still delivers strong performance.

	Metric	BN	BN-A2A	Blend	SIG	ISSBA	WaNet	BATT
No	ASR	98.4	12.3	93.1	70.9	97.2	92.6	99.6
	ACC	58.7	44.0	56.1	58.8	52.5	50.3	61.8
w/o	ASR	12.2	1.2	**2.89**	6.4	6.1	5.8	17.1
	ACC	49.5	46.9	51.1	50.6	49.4	47.6	58.5
ATClean	ASR	**8.8**	**0.8**	7.3	**0.6**	**0.7**	**0.3**	**13.9**
	ACC	50.9	47.6	50.1	51.2	52.0	60.1	54.6

6 Limitations and Future Works

Although *ATClean* shows strong performance and we provide simplified variants (*e.g.*, detector-free and root-data-free), the method remains relatively complex and is currently tested only on small models. Scaling layerwise defense to larger architectures and adaptively selecting optimal layers is beyond the scope of this paper, and we leave it for future work. Reinforcement learning may be a promising direction for enabling scalable layerwise defense.

7 Conclusion

This paper proposes a post-processing defense, *ATClean*, to counter backdoor attacks in data outsourcing scenarios. Unlike prior defenses that focus only on output logits, rely on perfect trigger recovery, inject uncertain triggers, or overlook poisoned representations, *ATClean* leverages adaptive feature collisions to disrupt backdoors and neutralize poisoned models. Specifically, rather than focusing solely on output logits, it explores all layers to locate backdoor-affected regions; it relaxes the need for perfect trigger recovery by generating adversarial samples that only enforce guaranteed collisions; instead of injecting uncertain triggers, it suppresses malicious features through collisions; and it fully exploits poisoned representations via feature-collision-based fine-tuning. Experiments demonstrate that *ATClean* outperforms prior methods, achieving SOTA defense effectiveness with the lowest clean-data performance drop among successful defenses, and about a 20% improvement in DER, which measures the accuracy–defense trade-off.

8 Acknowledgments

We thank the anonymous reviewers for their valuable feedback. The work of Z. Xiong and H. Wang was supported in part by the United States National Science Foundation (NSF) under grants 2534286, 2523997, 2315612, and 2332638 and by the AWS Cloud Credit for Research program. The work of J. Li was supported in part by NSF under grant 2315614. The work of M. Pan was supported in part by the NSF under grants CNS-2107057, CNS-2318664, CSR-2403249, and CNS-2431596. The work of X. Du was supported in part by the NSF under grants CNS-2204785, CNS-2205868, and 2409212. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

References

[1] Dosovitskiy Alexey. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv: 2010.11929* (2020).

[2] Eugene Bagdasaryan and Vitaly Shmatikov. 2021. Blind backdoors in deep learning models. In *Proc .30th USENIX Security Symposium (USENIX Security 21)*.

[3] Mauro Barni, Kassem Kallas, and Benedetta Tondi. 2019. A new backdoor attack in cnns by training set corruption without label poisoning. In *Proc. 26th IEEE International Conference on Image Processing (ICIP)*.

[4] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. 2018. Detecting backdoor attacks on deep neural networks by activation clustering. In *Proc. 33th Conference on Artificial Intelligence Workshop (AAAI Workshop)*.

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

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

[7] Yukun Chen, Shuo Shao, Enhao Huang, Yiming Li, Pin-Yu Chen, Zhan Qin, and Kui Ren. 2025. Refine: Inversion-free backdoor defense via model reprogramming. *arXiv preprint arXiv:2502.18508* (2025).

[8] Yiming Chen, Haiwei Wu, and Jiantao Zhou. 2024. Progressive poisoned data isolation for training-time backdoor defense. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 11425–11433.

[9] Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. 2023. Diffusion models in vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 9 (2023), 10850–10869.

[10] P Kingma Diederik. 2014. Adam: A method for stochastic optimization. *(No Title)* (2014).

[11] Kuofeng Gao, Yang Bai, Jindong Gu, Yong Yang, and Shu-Tao Xia. 2023. Backdoor defense via adaptively splitting poisoned dataset. In *Proc. 22nd Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[12] Yansong Gao, Yeonjae Kim, Bao Gia Doan, Zhi Zhang, Gongxuan Zhang, Surya Nepal, Damith C Ranasinghe, and Hyoungshick Kim. 2021. Design and evaluation of a multi-domain trojan detection method on deep neural networks. In *Proc. IEEE Transactions on Dependable and Secure Computing (DSC)*.

[13] Yansong Gao, Change Xu, Derui Wang, Shiping Chen, Damith C Ranasinghe, and Surya Nepal. 2019. STRIP: A Defence Against Trojan Attacks on Deep Neural Networks. In *Proc. 35th Annual Computer Security Applications Conference (ACSAC)*.

[14] Leon Gatys, Alexander S Ecker, and Matthias Bethge. 2015. Texture synthesis using convolutional neural networks. *Advances in neural information processing systems* 28 (2015).

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

[16] Jiyang Guan, Jian Liang, and Ran He. 2024. Backdoor Defense via Test-Time Detecting and Repairing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 24564–24573.

[17] Junfeng Guo, Yiming Li, Xun Chen, Hanqing Guo, Lichao Sun, and Cong Liu. 2023. Scale-up: An efficient black-box input-level backdoor detection via analyzing scaled prediction consistency. In *Proc. 11th International Conference on Learning Representations (ICLR)*.

[18] Xuyao Guo, Feng Jiang, Quanzhen Chen, Yuxuan Wang, Kaiyue Sha, and Jing Chen. 2025. Deep learning-enhanced environment perception for autonomous driving: MDNet with CSP-DarkNet53. *Pattern Recognition* 160 (2025), 111174.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proc. 29th Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

[20] Linshan Hou, Ruili Feng, Zhongyun Hua, Wei Luo, Leo Yu Zhang, and Yiming Li. 2024. IBD-PSC: Input-level Backdoor Detection via Parameter-oriented Scaling Consistency. In *Proc. 41st The International Conference on Machine Learning (ICML)*.

[21] Kunzhe Huang, Yiming Li, Baoyuan Wu, Zhan Qin, and Kui Ren. 2022. Backdoor defense via decoupling the training process. *arXiv preprint arXiv:2202.03423* (2022).

[22] Lamyanba Laishram, Muhammad Shaheryar, Jong Taek Lee, and Soon Ki Jung. 2025. Toward a privacy-preserving face recognition system: A survey of leakages and solutions. *Comput. Surveys* 57, 6 (2025), 1–38.

[23] Ya Le and Xuan Yang. 2015. Tiny imagenet visual recognition challenge. *CS 231N* 7, 7 (2015), 3.

[24] Ke Li, Tianhao Zhang, and Jitendra Malik. 2019. Approximate feature collisions in neural nets. In *Proc. 33th Advances in Neural Information Processing Systems (NeurIPS)*.

[25] Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. 2022. Backdoor learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[26] Yuezun Li, Yiming Li, Baoyuan Wu, Longkang Li, Ran He, and Siwei Lyu. 2021. Invisible backdoor attack with sample-specific triggers. In *Proc. 18th th the IEEE/CVF international conference on computer vision(ICCV)*.

[27] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. 2021. Anti-Backdoor Learning: Training Clean Models on Poisoned Data. In *Proc. 35th Advances in Neural Information Processing Systems (NeurIPS)*.

[28] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. 2021. Neural attention distillation: Erasing backdoor triggers from deep neural networks. In *Proc. 9th International Conference on Learning Representations (ICLR)*.

[29] Yiming Li, Mengxi Ya, Yang Bai, Yong Jiang, and Shu-Tao Xia. 2023. BackdoorBox: A Python Toolbox for Backdoor Learning. In *ICLR Workshop*.

[30] Yiming Li, Tongqing Zhai, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. 2021. Backdoor attack in the physical world. In *ICLR Workshop*.

[31] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2018. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *Proc. 21st International symposium on research in attacks, intrusions, and defenses (RAID)*.

[32] Liangkai Liu, Sidi Lu, Ren Zhong, Baofu Wu, Yongtao Yao, Qingyang Zhang, and Weisong Shi. 2020. Computing systems for autonomous driving: State of the art and challenges. *IEEE Internet of Things Journal* 8, 8 (2020), 6469–6486.

[33] Min Liu, Alberto Sangiovanni-Vincentelli, and Xiangyu Yue. 2023. Beating backdoor attack at its own game. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 4620–4629.

[34] Yahui Liu, Enver Sangineto, Wei Bi, Nicu Sebe, Bruno Lepri, and Marco Nadai. 2021. Efficient training of visual transformers with small datasets. *Advances in Neural Information Processing Systems* 34 (2021), 23818–23830.

[35] Yuntao Liu, Yang Xie, and Ankur Srivastava. 2017. Neural trojans. In *2017 IEEE International Conference on Computer Design (ICCD)*. IEEE, 45–48.

[36] Anh Nguyen and Anh Tran. 2021. WaNet–Imperceptible Warping-based Backdoor Attack. In *Proc. 9th International Conference on Learning Representations (ICLR)*.

[37] Tuan Anh Nguyen and Anh Tran. 2020. Input-aware dynamic backdoor attack. *Advances in Neural Information Processing Systems* 33 (2020), 3454–3464.

[38] Utku Ozbulak, Manvel Gasparyan, Shodhan Rao, Wesley De Neve, and Arnout Van Messem. 2022. Exact Feature Collisions in Neural Networks. *arXiv preprint arXiv:2205.15763* (2022).

[39] Anum Paracha, Junaid Arshad, Mohamed Ben Farah, and Khalid Ismail. 2024. Machine learning security and privacy: a review of threats and countermeasures. *EURASIP Journal on Information Security* 2024, 1 (2024), 10.

[40] Fanchao Qi, Yangyi Chen, Mukai Li, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2020. Onion: A simple and effective defense against textual backdoor attacks. *arXiv preprint arXiv:2011.10369* (2020).

[41] Ximing Qiao, Yukun Yang, and Hai Li. 2019. Defending neural backdoors via generative distribution modeling. In *Proc. 32nd Advances in Neural Information Processing Systems (NeurIPS)*.

[42] Huming Qiu, Junjie Sun, Mi Zhang, Xudong Pan, and Min Yang. 2024. BELT: Old-School Backdoor Attacks can Evade the State-of-the-Art Defense with Backdoor Exclusivity Lifting. In *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2124–2141.

[43] Arezoo Rajabi, Surudhi Asokraj, Fengqing Jiang, Luyao Niu, Bhaskar Ramasubramanian, James Ritcey, and Radha Poovendran. 2023. MDTD: A Multi-Domain Trojan Detector for Deep Neural Networks. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 2232–2246.

[44] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer, 234–241.

[45] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. 2020. Hidden trigger backdoor attacks. In *Proc. 33th AAAI conference on artificial intelligence*.

[46] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proc. 16th Proceedings of the IEEE international conference on computer vision (ICCV)*.

[47] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. 2018. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Proc. 32th Advances in Neural Information Processing Systems (NeurIPS)*.

[48] Guanhong Tao, Yingqi Liu, Guangyu Shen, Qiuling Xu, Shengwei An, Zhuo Zhang, and Xiangyu Zhang. 2022. Model orthogonalization: Class distance hardening in neural networks for better security. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1372–1389.

[49] Warren S. Torgerson. 1952. Multidimensional Scaling: I. Theory and Method. *Psychometrika* 17, 4 (1952), 401–419. doi:10.1007/BF02288916

[50] Brandon Tran, Jerry Li, and Aleksander Madry. 2018. Spectral signatures in backdoor attacks. In *Proc. 32th Advances in Neural Information Processing Systems (NeurIPS)*.

[51] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. 2018. Clean-label backdoor attacks. (2018).

[52] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. 2019. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *Proc. 40th IEEE symposium on security and privacy (SP)*.

[53] Jiahao Wang, Xianglong Zhang, Xiuzhen Cheng, Pengfei Hu, and Guoming Zhang. 2024. A Practical Trigger-Free Backdoor Attack on Neural Networks. *arXiv preprint arXiv:2408.11444* (2024).

[54] Ren Wang, Gaoyuan Zhang, Sijia Liu, Pin-Yu Chen, Jinjun Xiong, and Meng Wang. 2020. Practical Detection of Trojan Neural Networks: Data-Limited and Data-Free Cases. In *Proc. 16th European Conference on Computer Vision (ECCV)*.

[55] Zhenting Wang, Kai Mei, Juan Zhai, and Shiqing Ma. 2023. Unicorn: A unified backdoor trigger inversion framework. *arXiv preprint arXiv:2304.02786* (2023).

[56] Shaokui Wei, Hongyuan Zha, and Baoyuan Wu. 2024. Mitigating backdoor attack by injecting proactive defensive backdoor. *arXiv preprint arXiv:2405.16112* (2024).

[57] Shaokui Wei, Mingda Zhang, Hongyuan Zha, and Baoyuan Wu. 2023. Shared adversarial unlearning: Backdoor mitigation by unlearning shared adversarial examples. In *Proc. 37th Advances in Neural Information Processing Systems (NeurIPS)*.

[58] Baoyuan Wu, Hongrui Chen, Mingda Zhang, Zihao Zhu, Shaokui Wei, Danni Yuan, and Chao Shen. 2022. BackdoorBench: A Comprehensive Benchmark of Backdoor Learning. In *Proc. 36th Conference on Neural Information Processing Systems Datasets and Benchmarks Track (NeurIPS)*.

[59] Tong Xu, Yiming Li, Yong Jiang, and Shu-Tao Xia. 2023. Batt: Backdoor attack with transformation-based triggers. In *Proc. 48th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

[60] Xiong Xu, Kunzhe Huang, Yiming Li, Zhan Qin, and Kui Ren. 2024. Towards reliable and efficient backdoor trigger inversion via decoupling benign features. In *The Twelfth International Conference on Learning Representations*.

[61] Yi Zeng, Si Chen, Won Park, Z Morley Mao, Ming Jin, and Ruoxi Jia. 2020. Adversarial unlearning of backdoors via implicit hypergradient. In *Proc. 8th International Conference on Learning Representations (ICLR)*.

[62] Yi Zeng, Minzhou Pan, Hoang Anh Just, Lingjuan Lyu, Meikang Qiu, and Ruoxi Jia. 2023. Narcissus: A practical clean-label backdoor attack with limited information. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 771–785.

[63] Zaixi Zhang, Qi Liu, Zhicai Wang, Zepu Lu, and Qingyong Hu. 2023. Backdoor defense via deconfounded representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12228–12238.

[64] Pu Zhao, Pin-Yu Chen, Payel Das, Karthikeyan Natesan Ramamurthy, and Xue Lin. 2021. Bridging mode connectivity in loss landscapes and adversarial robustness. In *Proc. 9th International Conference on Learning Representations (ICLR)*.

[65] Runkai Zheng, Rongjun Tang, Jianze Li, and Li Liu. 2022. Data-free backdoor removal based on channel lipschitzness. In *European Conference on Computer Vision*. Springer, 175–191.

[66] Nan Zhong, Zhenxing Qian, and Xinpeng Zhang. 2022. Imperceptible backdoor attack: From input space to feature representation. *arXiv preprint arXiv:2205.03190* (2022).

[67] Mingli Zhu, Shaokui Wei, Li Shen, Yanbo Fan, and Baoyuan Wu. 2023. Enhancing fine-tuning based backdoor defense with sharpness-aware minimization. In *Proc. 19th CVF International Conference on Computer Vision (ICCV)*.

[68] Mingli Zhu, Shaokui Wei, Hongyuan Zha, and Baoyuan Wu. 2024. Neural polarizer: A lightweight and effective backdoor defense via purifying poisoned features. In *Proc. 38th Advances in Neural Information Processing Systems (NeurIPS)*.

Appendices

A Theoretical Analysis

A.1 Proof of Proposition 2

In this section, we aim to demonstrate that training $f_{w'}$ on X_{gen} is equivalent to minimizing l_{risk}, and we establish the existence of feature collisions between backdoor triggers to support this proposition.

Proof. According to the definition of feature collisions, if feature collisions exist between $B_\theta(x_i)$ for $(x_i, y_i) \in X_{root}$, and x_j for $(x_j, y_j) \in X_{suspicious}$, then we have $f(B_\theta(x_i)) = f(x_j)$. This implies that our generated backdoor samples are indistinguishable from suspicious samples in the model's perspective. By labeling these samples with the clean label y_i, we minimize $p(f_w(x_j) = c)$, where c is the target label. Furthermore, since l_{total} also maintains the distance between generated and clean samples, it preserves the clean samples-clean labels relationship without degrading the model's performance on clean data. Thereby, training on X_{gen} is equivalent as minimizing l_{risk} of Equation 5.

Remark. For cases where X_{sus} contains clean samples, Equation (5) can be decomposed as follows:

$$\frac{1}{|X_{root}^{-c}|}\Big[\sum_{i=1}^{|X_{clean}^{-c}|} \mathcal{I}\left(f_w(B_\theta(x_i)) = y_i\right) + \tag{7}$$

$$\sum_{j=1}^{|D_{sus}^{-c}|} \mathcal{I}\left(f_w(B_\theta(x_j)) = y_j\right)\Big]. \tag{8}$$

This decomposition can still remove the backdoor as long as the second term is dominant, a condition that is consistently met as discussed in the related works and our sensitivity analysis. Here, X_{clean}^{-c} represents the portion of X_{root}^{-c} that is not transformed into potential predicted backdoor samples by B_θ, while X_{sus}^{-c} represents the remaining part. This distinction arises because the loss function in Step 3 learns the distribution of X_{sus} where part of the data is poisoned, and thereby Proposition 2 is still true.

Proposition 3. *Minimizing* $l_{locate} = \frac{1}{4N_L{}^2 M_L{}^2}\sum_{i,j}(h^r_{L/i,j} - h^s_{L/i,j})^2$ *is equivalent to maximizing feature collisions.*

Proof. If $l_{locate} = 0$, then $f^{(L)}_{w'}(B_\theta(x_i)) = f^{(L)}_{w'}(x_j)$ for $x_i \in X_{root}$ and $x_j \in X_{suspicious}$. Given that $x_i \neq x_j$, setting $l_{locate} = 0$ aligns with the definition of feature collisions. Therefore, minimizing l_{locate} is equivalent to maximizing feature collisions.

Theorem 1. *$\forall x_i \in X$, there exists one or more non-zero backdoor triggers ϕ_i, satisfying $f(x_i + \phi_i) = f(x_i)$ as long as the input dimension of the model is bigger than the output size of the model. [38]*

Proof According to the definition of feature collisions, if $f(x_i + \phi_i) = f(x_i)$, there must exist a layer k such that $f^{(k)}(x_i + \phi_i) = f^{(k)}(x_i)$. To demonstrate this, we will consider the following two cases separately:

Case I: If the k-th layer is a fully connected layer, $f^{(k)}(x_i)$ can be rewritten as $v_k^T W_k$, where v_k is the output of the $(k-1)$-th layer, and W_k is the weight matrix of the k-th layer with shape (q, p). Therefore, if the input dimension is larger than the output dimension, W_k must have zero eigenvalue(s), implying that $rank(W_k) < q$.

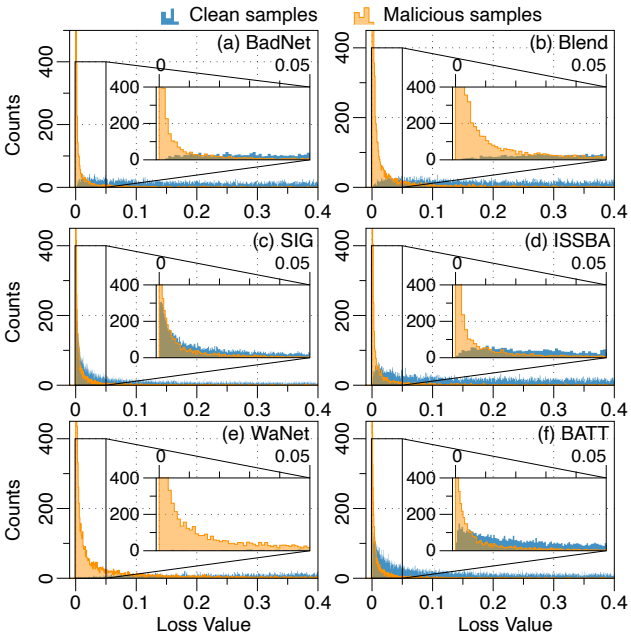

Figure 13: Loss value distribution of clean samples v.s. malicious samples. This shows that the attacker can mostly be distinguished directly through the loss, and combined with the previous results, it demonstrates that even if the detector fails, *ATClean* remains effective.

Consequently, $rank(W_k^T) < q$. From the fundamental theorem of linear algebra, we know: $nullity(W_k^T) + rank(W_k^T) = q$.

Since $nullity(W_k^T) = dim(Ker(W_k^T))$, it follows that $dim(Ker(W_k^T)) \geq 1$, which further implies the existence of a non-zero basis vector $\Phi_i \in Ker(W_k^T)$ that satisfies $\phi_i^T W_k = 0$. Therefore we have the following equation:

$$(v_k + \Phi_i)^T W_k = v_k^T W_k + \Phi_i^T W_k = v_k^T W_k.$$

This can be rewritten as $f^{(k)}(x_i + \phi_i) = f^{(k)}(x_i)$, where ϕ_i is defined as $\Phi_i^T W_{k-1} = f^{(k-1)}(x_i + \phi_i) - f^{(k-1)}(x_i)$. Since $\Phi_i \neq 0$, $\phi_i \neq 0$.

Case II: If the k-th layer is a convolutional layer, we can rewrite $f^{(k)}(x_k)$ as the matrix multiplication $V_k W_k$, where $V_k \in R^{l \times q}$. Following the approach in Case (1), we can derive $(V_k + P)W_k = V_k W_k$, where P is defined as $P = [\phi_i, \cdots, \phi_i]^T$, with ϕ_i repeated l times, where l is the number of rows in the linearized matrix for the convolutional operation in the k-th layer.

Remark. Theorem 1 establishes the existence of feature collisions between backdoor triggers, while Proposition 3 demonstrates that minimizing the loss function is equivalent to maximizing feature collisions. Additionally, Proposition 2 shows that if feature collisions are present, B_θ can be used to generate X_{gen}, effectively removing the backdoor in Step 4.

B Algorithm

This section provides the pseudocode for Steps 3–4 of *ATClean*. See the accompanying Python files for additional details.

Algorithm 1: *ATClean*

Input: Root data X_{root}, suspicious data $X_{suspicious}$, poisoned model $f_{w'}$, N_1, N_2
Output: Trigger generator B_θ, purified model $f_{w''}$
For N_1 rounds:
    For x_i, x_j in X_{root} and $X_{suspicious}$:
        Update B_θ by minimizing $l_{total}(B_\theta, x_i, x_j)$
Freeze B_θ and unfreeze $f_{w'}$
For N_2 rounds:
    For $x \in X_{root}$
        Generate $x_{gen} = B_\theta(x)$
        Update $f_{w'}$ by fine-tuning on x_{gen} to get $f_{w''}$
Return the B_θ, and $f_{w''}$

C Additional Experiments

C.1 More Results of motivation experiments

As indicated in Case 4 of Table 15, even trigger 4 and trigger 1 have different sizes and shapes, and there's no overlap between two triggers, ASR can still be significantly reduced in a single round.

C.2 More Performance Analysis

As shown in Table 5, NAB fails to recover the ground-truth labels for all poisoned samples. In contrast, *ATClean* achieves the highest R-ACC, demonstrating its superiority in preserving model performance on both clean and malicious samples.

Figure 14: Feature collisions of triggers 1,2, and 4.

Table 15: Case 1: Backdoor attack with Trigger 1 alone. Case 2: Trigger 1 first and then Trigger 2. Case 3: Trigger 1 first and then Trigger 3. Case 4: Trigger 1 first and then Trigger 4. Case 5: Trigger 1 first and then Trigger 5. Case 4 shows the feature collision between Triggers 1 and 4.

Case	Triggers	ACC (%)	ASR (%)
1	Trigger 1	81.69	80.14
2	Triggers 1 & 2	35.34	67.33
3	Trigger 1 & 3	80.08	74.63
4	Triggers 1 & 4	28.89	77.14
5	Triggers 1 & 5	79.85	74.01

C.3 Data-free Variant

As shown in Table 16, we use a pre-trained diffusion model (not trained on CIFAR-10) to generate the root dataset, enabling a truly data-free version of ATClean (ATClean w/o root data), and the results below show a better performance than CLP.

Table 16: Performance comparison of the truly data-free version.

Attack (ACC/ASR)	No Defense	CLP [65]	ATClean w/o root data
BadNet	58.7/98.4	57.3/7.32	58.09/6.83
Blend	56.1/93.1	51.68/0.32	55.52/6.53
SIG	58.8/70.9	50.80/35.22	58.19/5.04
ISSBA	52.5/97.2	49.72/10.89	51.95/6.12
WaNet	50.3/92.6	49.99/5.90	49.98/5.85
BATT	61.8/99.6	61.74/34.22	61.36/17.19

Table 17: Failure cases analysis of *ATClean*.

Attack (ACC/ASR)	No Defense	SAU	layer-wise SAU	ATClean
BATT	61.8/99.6	54.8/31.7	58.53/18.57	54.6/13.9

C.4 Failure Cases Analysis of *ATClean*

Our failure analysis in Table 17 (combined with Figure 5 and Table 2) reveals that when ATClean fails to reconstruct poisoned samples, such as those of BATT, the full pipeline collapses into a layer-wise SAU procedure (SAU is optimized on the output logit; layer-wise SAU is the SAU optimized on every layer) in these cases, but it is still more effective than SAU.

C.5 Test against Trigger-free Attacks

As shown in the Table 19, our ATClean was able to successfully defend against the trigger-free backdoor attack [53]. ATClean succeeds because its layerwise adaptive adversarial learning (verified in Q2) actively searches for and suppresses unstable or malicious decision-boundary expansions across intermediate representations, making the model's classification regions more robust and thereby invalidating the trigger-free attack's attempt to covertly enlarge the target class boundary (similar to SAU).

C.6 A More Fine-grained Root Data Size Sensitivity Analysis.

Our granular experiments in Table 18 show that ATClean has a moderate performance drop at 0.5 percent clean root data, and its performance at 0.1 percent remains similar to the 0.5 percent setting, yet both still perform better than SAU. This suggests that although ATClean benefits from clean root data, its finer-grained layerwise adversarial learning provides stronger robustness. In the extreme case of zero clean root data, ATClean naturally reduces to a more fine-grained form of SAU, consistent with the results of our detector-free version.

C.7 More Analysis of the Detector-free Variant

We printed several optimized images produced by the detector-free version on CIFAR-10. Although these images do not match the original patterns, their DoF values (as shown in Table 20) and defense performance exceed those of SAU, showing that our method still achieves a finer-grained defense through feature collision.

C.8 Incremental Benefits Analysis

We compare vanilla PIPD [8] with the combination of PIPD and ATClean in the Table 21. ATClean further reduces ASR because it explicitly strengthens the model's decision boundary, making

Table 18: Effectiveness of *ATClean* with different root data sizes.

Attack	0% (w/o detector)	0.1%	0.5%	SAU
BadNet	0/12.21/49.47	0.8/9.17/50.1	0.9/8.5/50.3	0/27.2/40.8
Blend	0/2.89/51.07	0.7/7.4/52.3	0.8/8.8/50.1	0/31.4/40.3
SIG	0/6.44/50.58	0.6/4.6/49.4	0.7/0.3/50.7	0/6.7/19.3

Table 19: Test against trigger-free attacks.

Attack (ACC/ASR)	Before	SAU	ATClean w/o root data
Trigger-free Attack [53]	61.75/98.54	57.48/8.92	58.01/3.41

Table 20: Feature collisions of different defenses.

Dof	BadNet	Blend	SIG	ISSBA	WaNet	BATT
SAU	0.62	0.59	0.56	0.61	0.65	0.42
ATClean w/o detector	0.74	0.87	0.74	0.82	0.89	0.78
ATClean	0.77	0.92	0.76	0.86	0.82	0.86

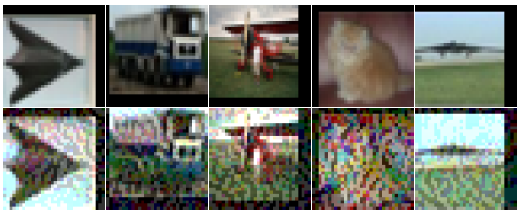

Figure 15: Detector-free variant generated samples. The first row is clean images, and the second row is generated samples of the detector-free variant against 5 attacks (*i.e.*, BadNet, Blend, SIG, ISSBA, WaNet).

the poisoned model more robust; backdoor attacks succeed largely when this robustness is insufficient. In addition, PIPD is a loss-based defense, and its filtering stage can be bypassed by attacks that deliberately shape poisoned samples to have loss values similar to clean data (*e.g.*, Grond [60]). By reinforcing robustness at a finer granularity, ATClean mitigates these bypass cases more effectively, leading to a more substantial ASR reduction.

D More Discussions

D.1 Novelty Concerns in Steps 2 and 3

Our method does not build upon prior works but instead only incorporates certain auxiliary elements in Step 2 and Step 3, none of which constitute our core contribution. Specifically, in Step 2, we employ existing detection methods with minimal precision requirements (*i.e.*, precision > 0.5), in contrast to ABL's loss-based isolation that fails even when precision drops to 0.8 under the lower-loss violation [68]. In Step 3, although both our framework and DeepInspect [5] utilize generative models, DeepInspect critically depends on knowing the target labels of backdoors, an unrealistic assumption in all-to-all settings that would require a perfect detector. Our approach makes no such assumptions and remains effective under diverse attack scenarios. These shared elements are not the source of our contribution; rather, our ablation studies and analysis above confirm that the effectiveness of *ATClean* arises from its collision-driven optimization framework, not from merely reusing detection modules or generative components.

Table 21: Incremental benefits analysis of *ATClean*.

Attack (ACC/ASR)	Vanilla PIPD	PIPD & ATClean
SIG	55.29/0.43	56.09/0.23
BadNet	58.91/0.43	58.52/0.53
BATT	60.38/10.42	60.19/5.04
Grond	61.82/29.42	61.47/9.42

D.2 Coverage of One-to-one Attacks

Our experiments already evaluate all-to-all attacks as shown in Table 2. Since an all-to-all attack is strictly harder than a one-to-one attack due to its more dispersed target labels and weaker label majority signals, a method that works under all-to-all inherently covers the one-to-one setting. *ATClean* relies on feature collisions rather than label dominance or low-loss assumptions, so mixed labels do not hinder suspicious sample selection or generator training. As shown in our results of Table 2, *ATClean* successfully reconstructs triggers and reduces the backdoor effect under all-to-all attacks, which subsumes the one-to-one case.

E Details of Experiment Setup

Unless otherwise specified, all attacks and defenses follow the original paper's settings. Except for the main experiments, all results are measured on Tiny ImageNet using a ResNet-18 backbone.

E.1 Attack Methods and Settings

BadNet [15]: We apply a 5X5 square trigger at position (6, 6).
ISSBA [26]: We follow the setting of the original paper: we pick up 20 as the secret size; The encoder is trained on 1% of the whole training set for 20 rounds with tiny ImageNet, and 10% of the whole training set for 20 rounds on CIFAR-1; For the encoder, we choose the same structure as the original paper [26].
WaNet [36]: We pick up k=4, s=0.5, noise scale factor=2, and grid re-scale factor=1 following [29, 58].
Blend [6]: We select image indexed 656 in the test dataset as the trigger and we set α=0.2 following the original paper.
BATT [59]: We choose rotation degree as 16°.
SIG [3]: We follow the same setting of [58]: frequency $f = 6$, $\delta = 40$.
BELT [42]: We follow the same setting of the original paper.

E.2 Defense Methods and Settings

FT [31]: We use Adam [10] as the optimizer with a learning rate of $\eta = 10^{-4}$, running it for 20 rounds.
ABL [27]: We follow the settings of the original paper, with an isolation rate of $p = 0.01$, $\gamma = 0.5$, and 20 rounds.
SAU [57]: We follow the same settings of [58] and we use SGD as the optimizer for 20 rounds.
i-BAU [61]: We follow the same settings of [58].
NAB [33]: We follow the same settings of the original paper.
NPD [68]: We follow the same settings of the original paper.
PDB [56]: We follow the same settings of the original paper.
ATClean: In Step 4, we select Adam [10] as the optimizer, using a learning rate of $\eta = 10^{-4}$ for CIFAR-10 and $\eta = 10^{-5}$ for Tiny ImageNet, and run it for 15 epochs. The structure of B_θ is illustrated in Figure 4.

Received 20 February 2007; revised 12 March 2009; accepted 2026