Mitigating Backdoor Attack by Injecting Proactive Defensive Backdoor

Shaokui Wei[1] **Hongyuan Zha**[1,2] **Baoyuan Wu**[1]*
[1]School of Data Science,
The Chinese University of Hong Kong, Shenzhen, Guangdong, 518172, P.R. China
[2]Shenzhen Key Laboratory of Crowd Intelligence Empowered Low-Carbon Energy Network

Abstract

Data-poisoning backdoor attacks are serious security threats to machine learning models, where an adversary can manipulate the training dataset to inject backdoors into models. In this paper, we focus on in-training backdoor defense, aiming to train a clean model even when the dataset may be potentially poisoned. Unlike most existing methods that primarily detect and remove/unlearn suspicious samples to mitigate malicious backdoor attacks, we propose a novel defense approach called PDB (**P**roactive **D**efensive **B**ackdoor). Specifically, PDB leverages the "home field" advantage of defenders by proactively injecting a defensive backdoor into the model during training. Taking advantage of controlling the training process, the defensive backdoor is designed to suppress the malicious backdoor effectively while remaining secret to attackers. In addition, we introduce a reversible mapping to determine the defensive target label. During inference, PDB embeds a defensive trigger in the inputs and reverses the model's prediction, suppressing malicious backdoor and ensuring the model's utility on the original task. Experimental results across various datasets and models demonstrate that our approach achieves state-of-the-art defense performance against a wide range of backdoor attacks. The code is available at https://github.com/shawkui/Proactive_Defensive_Backdoor.

1 Introduction

In recent years, deep neural networks (DNNs) have become ubiquitous across diverse fields, powering applications such as face recognition, self-driving vehicles, and medical image analysis [1, 13, 25, 38]. However, alongside these advancements, the vulnerability of DNNs to malicious attacks presents a critical challenge to their safety and reliability. A particularly alarming threat arises from backdoor attacks, where adversaries secretly introduce backdoors into DNN models during training by subtly altering a fraction of the dataset. This manipulation ensures the model's standard performance on uncontaminated data but erroneously assigns a pre-determined label to any input carrying a specific trigger. Considering its real threats to machine learning systems, especially in security-critical scenarios, it's a practical necessity to investigate and propose effective defense strategies against such attacks to safeguard real-world applications.

To mitigate the threats posed by backdoor attacks, researchers have actively explored various backdoor defense techniques throughout the life cycle of machine learning systems [45]. In this paper, we specifically delve into **in-training backdoor defense** [44–46], which aims to train machine learning models using datasets that may be contaminated with poisoned data. Most existing methods in this field primarily focuses on identifying suspicious samples through various means, along with mitigating the backdoor effect by directly removing [3, 48] or applying some techniques ($e.g.$, unlearning [20, 4], or relabel [15, 26, 60]) to the suspicious samples. Despite achieving remarkable performance in

*Corresponds to Baoyuan Wu (wubaoyuan@cuhk.edu.cn).

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

backdoor defense, these methods face certain limitations and challenges. First, most existing works rely on specific assumptions such as the latent separability [3] or the early learning of poisoned samples [20, 60, 51] to identify the poisoned samples. However, these assumptions may not hold under more sophisticated attacks [31]. As accurately detecting poisoned samples is crucial for those methods, any deviation from their underlying assumptions could lead to performance degradation and compromise their effectiveness. Second, some methods, such as DBD [15], NAB [26], and V&B [60], necessitate complex modifications to the training process, resulting in a substantial increase in training costs.

In this paper, instead of following the traditional detection-and-mitigation pipeline, we propose a proactive approach that leverages the "*home field*" advantage of defenders. Our method, called PDB (short for **P**roactive **D**efensive **B**ackdoor), aims to fight malicious backdoor attacks by injecting a proactive defensive backdoor introduced by the defenders themselves. The primary objective of PDB is to suppress the malicious backdoor with a defensive backdoor while keeping the model's utility on original task. Specifically, When the defensive trigger is presented, the defensive backdoor will dominate the prediction of the proactively backdoored model, effectively suppressing the malicious backdoor's impact. Importantly, our defensive backdoor allows for the restoration of the ground truth label to maintain the model's utility on the original task. To achieve this goal, we first analyze the objective for an effective defensive backdoor and introduce four essential design principles, including **reversibility**, **inaccessibility to attackers**, **minimal impact on model performance**, and **resistance against other backdoors**. Then, we construct an additional defensive poisoned dataset, subsequently utilizing such dataset and the whole poisoned dataset to train the model. Consequently, if only the malicious trigger is present, the model remains under the control of the malicious backdoor. However, when the defensive trigger appears, the defensive backdoor is activated, mitigating the malicious backdoor effect. To evaluate its effectiveness, we compare PDB with five state-of-the-art (SOTA) in-training defense methods across seven SOTA data-poisoning backdoor attack methods involving different model structures and datasets. Our experimental results demonstrate that PDB achieves comparable or even superior performance compared to existing baselines.

Our main contributions are threefold: **1)** We break away from the traditional detection-and-mitigation pipeline by proposing a novel mechanism that injects a proactive defensive backdoor during training, which suppresses the malicious backdoor while preserving the model's utility on the original task, without any specific assumptions about potential malicious backdoor attacks. **2)** By analyzing the primary objective, we introduce essential design principles for an effective defensive backdoor and propose a practical algorithm to implement the defensive backdoor. **3)** We conduct extensive experiments to evaluate the effectiveness of our method and compare it with five SOTA defense methods across seven challenging backdoor attacks, spanning diverse model structures and datasets, demonstrating the superior performance of the proposed method.

2 Related work

Backdoor attacks. DNNs face significant security threats from backdoor attacks, which are designed to maintain normal performance on regular inputs while forcing the network to output a predetermined target when a specific trigger is introduced. These attacks can be generally categorized into two types based on the property of the trigger: static-pattern backdoor attacks and dynamic-pattern backdoor attacks. The seminal instance of static-pattern backdoors, BadNets [12], employed fixed triggers like white squares. To enhance trigger stealthness, the Blended approach [5] was introduced, which merges the trigger with the host image in a subtle manner. Recognizing the potential for detection in fixed-pattern triggers, the research has pivoted towards dynamic-pattern backdoor attacks. Innovations in this direction, such as SSBA [22], WaNet [30], LF [49], WPDA [35], IRBA [10], VSSC [41] and TAT [6], have focused on crafting sample-specific triggers that are more challenging to identify. Techniques to refine the stealthness of triggers have been furthered by works like Sleeper-agent [36] and Lira [8], which optimize the output to be more covert. The sophistication of backdoor attacks has recently been advanced by strategies for learning-based poisoning sample selection [58] and re-activation attack [57]. To execute attacks without altering the consistency between the image and its label, 'clean label' attacks have been introduced. For example, LC [33] and SIG [2] employed counterfactual methods and additional techniques to modify the image while maintaining label consistency subtly.

Backdoor defenses. The main purpose of backdoor defense is to alleviate the vulnerabilities of DNNs to backdoor attacks by employing various strategies during different stages of the model lifecycle. Therefore, backdoor defenses are typically categorized into three types: pre-training, in-training, and post-training. Pre-training defenses concentrate on the detection and removal of poisoned data points before training. For example, AC [3] leverages unusual activation patterns to weed out poisoned data, while Confusion Training [32] relies on a model trained specifically to recognize poisoned instances. VDC [59] utilizes the capabilities of large multimodal language models for the same purpose. Post-training defenses are applied after a model has been trained. A line of works in this direction focusing on pruning [24, 47, 53, 52, 23] or fine-tuning [55, 28] to neutralize the backdoor. Besides, I-BAU [50], NPD [56], and SAU [43] reverse potential backdoor triggers by adversarial techniques to cleanse the model. NAD [21] employs a slightly poisoned model to assist in retraining a heavily compromised one.

This paper mainly focuses on the in-training defenses that aim to prevent backdoor insertion during the training phase. Along this direction, ABL [20] utilizes the observation that the poisoned samples are easier to learn than normal samples, resulting in the different learning speeds between benign and poisoned samples, to detect and unlearn the poisoned samples. Based on similar observation, V&B [60] first trains a backdoored model to capture the backdoor effect and utilizes the backdoored model to train a benign model by detecting and applying a series of operations on the suspicious samples. Similarly, CBD [51] first trains a backdoored model for a few epochs and trains a benign model by reweighting the samples and deconfounding the representation. DBD [15] splits the training process into three steps and employs self-supervised learning techniques to detect suspicious samples and train a benign model. D-ST [4] leverages the fact that benign samples are less sensitive to image transformations to detect suspicious samples and employs semi-supervised learning to train a benign model. Recently, a few attempts have been made to defend against malicious attacks by incorporating proactive attacks [54, 26]. The work most closely aligned with our approach is NAB [26], which first identifies and then relabels potentially poisoned samples in the dataset, subsequently embedding non-adversarial triggers into the suspicious samples to mitigate the backdoor effect. In contrast to their methodology, our technique offers a more straightforward solution, eliminating the need for costly detection and relabeling processes, thus reducing overall costs and complexity. In essence, we demonstrate that injecting a defensive backdoor alone is sufficient to defend against backdoor attacks without requiring detection and relabeling of the poisoned samples. We refer readers to [45] for more defense in adversarial machine learning.

3 Method

In Section 3.1, we introduce the essential notations and define the threat model in this paper. Subsequently, we explore the principles behind effective defensive backdoors, illustrated with practical examples in Section 3.2. We present the overall pipeline for our proposed method in Section 3.3.

3.1 Problem setting

Notations. Considering a sample $\boldsymbol{x} \in \mathcal{X}$ with label $y \in \mathcal{Y}$, a DNN model $f_{\boldsymbol{\theta}}$ parameterized by $\boldsymbol{\theta}$ is trained to classify $\boldsymbol{x}$. The space $\mathcal{Y} = [1, \cdots, K]$ denotes the space of candidate labels ($K \geq 2$), and $\mathcal{X}$ represents the sample space. In the context of backdoor attack, we denote the trigger by Δ and the trigger injection operator by $\oplus$. Consequently, given a benign sample $\boldsymbol{x}$, the poisoned sample can be generated by $\boldsymbol{x} \oplus \Delta$. It's important to note that the injection operator $\oplus$ can vary according to the type of trigger Δ.

Threat model. We consider a data poisoning scenario for **malicious backdoor** attack where the attacker can only manipulate a portion of the training dataset to plant trigge but cannot control the training process. By poisoning the dataset, the model trained on the manipulated dataset $\mathcal{D}_{tr}$ would normally perform for benign input but classify the inputs with malicious trigger Δ to predefined target $\hat{y}$. Besides, we define the portion of manipulated samples as the **poisoning ratio** of backdoor attack.

The defender faces a situation where a potentially poisoned dataset is given. The defender aims to train a model where the malicious backdoor fails to be activated by the malicious trigger, and the model's utility on the original task is maintained. We assume a small benign dataset $\mathcal{D}_{cl}$ is reserved

for the defender, which can be obtained by various means, including but not limited to purchase from reputable data vendors, generation via state-of-the-art generative models [7, 11, 16], collection from the internet, or applying data cleansing methods [45]. Moreover, we assume that the defender does not have knowledge of either the malicious trigger Δ or the malicious target label $\hat{y}$.

3.2 Proactive defensive backdoor

In this paper, we aim to defend the unknown malicious backdoor with the trigger Δ, by inserting a proactive defensive backdoor with a trigger Δ_1 into the model. Our primary objective is to ensure that when Δ_1 is presented, the model's output will be controlled by Δ_1 rather than Δ, thereby suppressing the malicious backdoor. Besides, the model's utility on the original task should be preserved, $i.e.$, user can still get the true prediction of the benign sample with the defensive trigger. To achieve such a defense goal, the desired defensive backdoor attack should follow the principles below:

- **Principle 1: Reversibility.** The defensive backdoor must be reversible, such that the ground truth label can be restored from the prediction of benign samples attached with Δ_1. Such a requirement is crucial for preserving the model performance on benign inputs with Δ_1.
- **Principle 2: Inaccessibility to attackers.** The defensive trigger Δ_1 should be meticulously designed to be non-replicable and undiscoverable by potential attackers. By doing so, we prevent adversaries from exploiting the same trigger or using inversion techniques to identify it.
- **Principle 3: Minimal impact on model performance.** While stealth is not a strict requirement for the defensive trigger, modified samples should retain sufficient characteristics of the original data. This ensures accurate label recovery from the model's predictions in the presence of Δ_1.
- **Principle 4: Resistance against other backdoors.** To effectively mitigate malicious backdoors, the defensive backdoor should be resistant to various backdoor attacks, not only known attacks but also potential future backdoors.

In light of the principles outlined above, we delve into the practical design of our defensive backdoor[2].

Following Principle 1. For the first principle, we propose to assign the target label by a bijective mapping $h : \mathcal{Y} \to \mathcal{Y}$, such that the target label of a sample with label y is $h(y)$ and the ground truth label of a poisoned image with label y is $h^{-1}(y)$. A typical choice of h and h^{-1} is $h(y) = (y+1) \mod K$ and $h^{-1}(y) = (y-1) \mod K$ where $\mod$ represents the modulo operation and K is the number of classes. It's worth noting that in the context of DNNs, h can also be formulated as a function of logits or features such as $h(\phi(\boldsymbol{x})) = -\phi(\boldsymbol{x})$ and $h^{-1}(\phi(\boldsymbol{x})) = -\phi(\boldsymbol{x})$ where $\phi(\boldsymbol{x})$ corresponds to the features or logits of input. This flexibility allows for a broader range of target label assignment strategies.

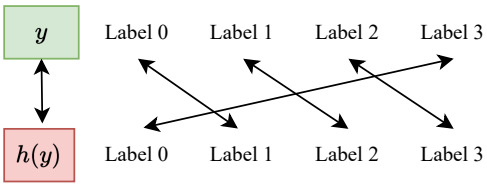

Figure 1: Illustration of bijective mapping with $h(y) = (y+1) \mod K$, with $K = 4$.

Following Principle 2 & 3. To follow the second and third principles, the design of the trigger is essential. Consider the patched trigger as an illustrative example, which can be constructed by carefully determining its position and pattern. Regarding the trigger's position, it should be crafted to preserve the core visual patterns of the original image, ensuring that the primary content remains unaltered. As for the trigger's pattern, we leverage the *"home field"* advantage of the defender, designing a trigger that operates beyond the conventional pixel space. Specifically, for an image with pixel values in the range of $[0, 1]$, the trigger is engineered to modify regions

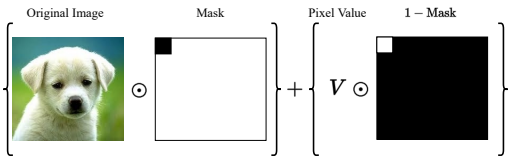

Figure 2: Demonstration of generating a defensive poisoned sample. $V \notin [0, 1]$ is the pixel value of trigger, $\odot$ is the element-wise product. For the mask, 0 is represented by black, while 1 is represented by white.

[2]It's important to acknowledge that alternative designs may also adhere to these principles.

to values beyond this range. This modification renders the trigger infeasible and not invertible by attackers, given the natural constraints of image data.

Following Principle 4. Following the fourth principle, the defensive backdoor is required to be resistant against other backdoors in the dataset. To meet such requirements, the key point is that the defender can control the training process, a *"home filed"* advantage that attackers lack. On the one hand, the defender can design a strong defensive backdoor, $e.g.$, adopting a large trigger. On the other hand, the defensive backdoor can be further enhanced by controlling the training process, $e.g.$, applying data augmentation or adjusting the weight of defensive poisoned samples. More discussion and empirical findings are presented in **Appendix** C.7.

3.3 Backdoor injection

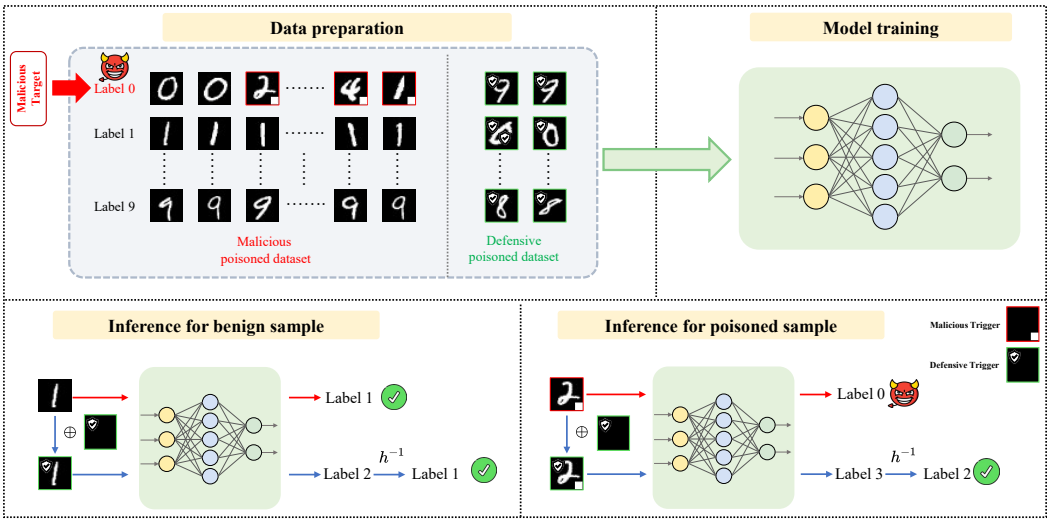

Figure 3: Overview of the proposed method. The trigger of the malicious backdoor is a white square, and its target label is 0. The trigger of the defensive backdoor is represented by a white shield, and the target label mapping is $h(y) = (y+1) \mod 10$ and $h^{-1}(y) = (y-1) \mod 10$.

As depicted in Figure 3, our proposed method involves three key steps:

Data preparation. Given a well-designed defensive backdoor with trigger Δ_1 and a target label mapping h, a defensive poisoned dataset is first constructed by

$$\hat{\mathcal{D}}_{def} = \{(\boldsymbol{x} \oplus \Delta_1, h(y)) | \forall (\boldsymbol{x}, y) \in \mathcal{D}_{cl}\}. \tag{1}$$

Model training. Now, a model can be trained on the combination of the malicious poisoned dataset $\mathcal{D}_{tr}$ and the defensive poisoned dataset $\hat{\mathcal{D}}_{def}$. Then, a well-trained model will normally perform for benign inputs while controlled by the corresponding backdoor when either the trigger Δ or Δ_1 is presented. However, if both Δ and Δ_1 are simultaneously presented, the model may become confused due to the lack of such samples in the training dataset. As aforementioned, to ensure that the defensive trigger Δ_1 effectively defeats an unknown trigger Δ, some *backdoor enhancement strategies* such as data augmentation or increasing sample weight can be adopted to enhance the defensive backdoor. In summary, the overall training objective is formulated as follows:

$$\min_{\boldsymbol{\theta}} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_{tr}} L_0(f_{\boldsymbol{\theta}}(\boldsymbol{x}), y) + \sum_{(\boldsymbol{x},y)\in\hat{\mathcal{D}}_{def}} \lambda_1 L_1(f_{\boldsymbol{\theta}}(\boldsymbol{x}), y) + \lambda_2 L_2(f_{\boldsymbol{\theta}}(\tau(\boldsymbol{x})), y), \tag{2}$$

where $\mathcal{D}_{tr}$ and $\hat{\mathcal{D}}_{def}$ are the maliciously poisoned training dataset and the defensive poisoned dataset, respectively. The operation τ enhances the defensive backdoor by applying operation on the defensive poisoned samples ($e.g.$, adding noise: $\tau(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(0, 1)$).

In (2), the first term stands for the loss on the poisoned dataset, the second term stands for the loss of injecting our defensive backdoor, and the third loss aims to enhance the defensive backdoor. We use L_0, L_1, and L_2 to represent the loss function for each term, which are usually Cross-Entropy losses if not specified. The parameters λ_1 and λ_2 are introduced to balance the contributions of the respective loss components. More details for the model training and implementation can be found in **Appendix** A.

Inference. During the inference, each input sample $\boldsymbol{x}$ is initially embedded with the defensive trigger, and the model's prediction $f_{\boldsymbol{\theta}}(\boldsymbol{x} \oplus \Delta_1)$ is obtained. Subsequently, the authentic prediction is reconstructed via the inverse mapping $h^{-1}(f_{\boldsymbol{\theta}}(\boldsymbol{x} \oplus \Delta_1))$.

Below, we provide a high-level pseudocode representation of our proposed method for training and inference:

Algorithm 1 Proactive Defensive Backdoor (PDB)

Input: Model $f_{\boldsymbol{\theta}}$, poisoned training set $\mathcal{D}_{tr}$, reserved benign dataset $\mathcal{D}_{cl}$, defensive trigger Δ_1, defensive target mapping h, max iteration number T.
Initialize $f_{\boldsymbol{\theta}}$.
▷ Data preparation
Construct the defensive poisoned dataset $\hat{\mathcal{D}}_{def} = \{(\boldsymbol{x} \oplus \Delta_1, h(y)) | (\boldsymbol{x}, y) \in \mathcal{D}_{cl}\}$.
▷ Model training
for $t = 0, ..., T-1$ **do**
  for each mini-batch in $\mathcal{D}_{tr} \cup \hat{\mathcal{D}}_{def}$ **do**
    Update $\boldsymbol{\theta}$ *w.r.t.* objective in (2).
  end for
end for
▷ Inference
for each input sample $\boldsymbol{x}$ **do**
  Predict its label by $h^{-1}(f_{\boldsymbol{\theta}}(\boldsymbol{x} \oplus \Delta_1))$.
end for

4 Experiments

4.1 Experiment setting

Backdoor attack. To assess our method, we consider seven leading backdoor attacks: BadNets [12], Blended method [5], Sinusoidal Signal (SIG) attacks [2], Sample-Specific Backdoor Attacks (SSBA) [22], WaNet [30], BPP attack [42] and TrojanNN attack [27]. Note that to expand our evaluation scope, we have modified certain attacks originally intended for training-controllable scenarios by excluding their training control components and we postpone the details to **Appendix** A. For a consistent and reliable evaluation, we utilize configurations from the BackdoorBench framework [44, 46], which offers a standardized backdoor attack assessment platform. Each attack is implemented with a 5% poisoning rate, targeting the 0^{th} label if not specified. The performance of these attacks is measured across three benchmark dataset, *i.e.*, CIFAR-10 [17], Tiny ImageNet [18], and GTSRB [37], and analyzed using three neural network architectures, *i.e.*, PreAct-ResNet18 [14] VGG19-BN [34] and ViT-B-16 [9]. Due to limitations in space, we present results for GTSRB and VGG19-BN in **Appendix** B. It is important to note that the clean label attack SIG is only applicable to CIFAR-10 with the set poisoning ratio. Additional information on these attacks is available in **Appendix** A.

Backdoor defense. In this paper, we benchmark our approach against popular and advanced backdoor defense methods, including AC [3], Spectral signatures [39], ABL [20], DBD [15], NAB [26]. For a fair comparison, we adopt the configurations recommended by the BackdoorBench framework [44, 46]. Note that we were unable to achieve satisfactory results for DBD on Tiny ImageNet with ViT-B-16, so it has been excluded in this case. For our method, we set the reserved dataset size to 10% of the training dataset unless otherwise specified. The chosen parameters are $\lambda_1 = 1$ and $\lambda_2 = 1$. To enhance the defensive backdoor, each defensive poisoned sample is sampled five times in an epoch, and we set $\tau(\boldsymbol{x}) = \boldsymbol{x} + 0.1 \cdot \boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(0, 1)$. The defensive backdoor utilizes a target mapping function $h(y) = (y + 1) \mod K$, along with a patch trigger with pixel

value 2 as illustrated in Figure 2. More details on the defense methods and supplementary experiments are postponed in **Appendix** A and **Appendix** B.

Metrics. To measure the effectiveness of each defense method, we employ three key metrics: Accuracy on benign data (**ACC**), Attack Success Rate (**ASR**), and Defense Effectiveness Rating (**DER**). ACC is the metric indicating model's performance for predicting the benign samples correctly, while ASR quantifies the proportion of poisoned samples that are incorrectly classified to the attacker's intended target label. For our method, the ACC is measured by predicting the benign samples with a defensive trigger to the defensive target, or equivalently reversing the prediction of the benign sample with a defensive trigger to the true label. A higher ACC and a lower ASR signify successful backdoor mitigation.

The DER, used in [55, 43], is a metric ranging from 0 to 1, designed to evaluate the trade-off between maintaining ACC and reducing ASR. It is defined by the following equation:

$$\text{DER} = [\max(0, \Delta\text{ASR}) - \max(0, \Delta\text{ACC}) + 1]/2, \tag{3}$$

where ΔASR and ΔACC represent the respective decreases in ASR and ACC between model without defense and model with defense.

Note: Superior defense methods are characterized by higher **ACC**, lower **ASR**, and higher **DER**. In the forthcoming experimental results, the best and second-best performing methods are denoted with **boldface** and underline, respectively.

4.2 Main results

Table 1: Results (%) on CIFAR-10 with PreAct-ResNet18 and poisoning ratio 5.0%.

Defense →	No Defense		AC [3]			Spectral [39]			ABL [20]			DBD [15]			NAB [26]			PDB (**Ours**)		
Attack ↓	ACC	ASR	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [12]	92.64	88.74	90.27	75.61	55.38	**91.21**	69.4	58.96	84.08	**0.00**	90.09	87.68	2.11	90.84	79.36	0.33	87.57	91.08	0.38	**93.40**
Blended [5]	93.67	99.61	**91.54**	98.93	49.27	91.43	99.42	48.97	65.78	**0.00**	85.86	75.0	99.99	40.66	90.21	0.34	97.90	91.36	0.70	**98.30**
SIG [2]	93.64	97.09	90.24	93.32	50.18	91.79	96.36	49.44	46.14	0.00	74.79	74.86	95.58	41.37	90.71	**0.00**	97.08	**91.79**	0.06	**97.59**
SSBA [22]	93.27	94.91	88.47	92.64	48.73	**92.01**	93.19	50.23	81.6	**0.09**	91.58	72.19	11.34	81.24	90.52	1.07	95.55	91.58	0.46	**96.38**
WaNet [30]	91.76	85.5	**91.96**	88.72	50.0	91.47	83.84	50.68	69.49	95.20	38.86	72.22	9.93	78.01	85.17	2.16	88.38	91.47	**0.92**	**92.14**
BPP [42]	91.47	99.34	89.64	97.96	49.78	**92.10**	99.82	50.0	82.89	99.93	45.71	81.71	99.98	45.12	82.86	76.94	56.90	90.43	**1.90**	**98.20**
Trojan [27]	93.79	99.99	89.40	99.93	47.83	86.30	99.38	46.56	18.64	100.00	12.42	72.34	100.0	39.27	87.41	1.16	96.23	**91.78**	**0.58**	**98.70**
Average	92.89	95.03	90.22	92.45	50.17	90.90	91.63	50.69	64.09	42.17	62.76	76.57	59.85	59.50	86.61	11.71	88.51	**91.36**	**0.71**	**96.39**

Table 2: Results (%) on Tiny ImageNet with ViT-B-16 and poisoning ratio 5.0%.

Defense →	No Defense		AC [3]			Spectral [39]			ABL [20]			NAB [26]			PDB (**Ours**)		
Attack ↓	ACC	ASR	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [12]	76.15	99.72	75.66	99.49	49.87	74.18	99.47	49.14	**78.19**	99.79	50.00	28.88	99.83	26.36	73.71	**0.00**	**98.64**
Blended [5]	76.00	99.83	77.58	99.77	50.03	76.30	99.62	50.11	**78.17**	99.93	50.00	38.43	99.79	31.24	72.70	**0.00**	**98.26**
SSBA [22]	75.30	98.86	76.01	97.01	50.93	76.96	98.73	50.07	**78.40**	99.59	50.00	42.03	99.37	33.36	72.65	**0.00**	**98.11**
WaNet [30]	60.90	99.74	75.68	92.46	53.64	74.27	95.91	51.91	**77.62**	95.37	52.18	21.89	99.76	30.50	72.82	**0.01**	**99.86**
BPP [42]	63.08	99.69	76.89	95.23	52.23	76.58	95.92	51.88	**78.19**	96.55	51.57	30.37	96.76	35.11	73.30	**0.00**	**99.84**
Trojan [27]	74.98	99.77	77.94	99.78	50.00	75.36	99.84	50.00	**78.40**	99.92	50.00	24.15	100.00	24.58	73.00	**0.00**	**98.89**
Average	71.07	99.60	76.63	97.29	51.12	75.61	98.25	50.52	**78.16**	98.52	50.63	30.96	99.25	30.19	73.03	**0.00**	**98.94**

Table 1 and Table 2 show the proposed method's defense performance compared with other methods, from which we can find:

PDB achieves consistent efficacy in mitigating backdoor threats across various attacks, datasets and models. Specifically, PDB achieves the top-2 lowest ASR across five out of seven attacks on the CIFAR-10 dataset. It also ranks top-2 across all attacks on the GTSRB (Table 7) and Tiny ImageNet. This consistent performance underscores PDB's ability to generalize well across different datasets and attacks. For AC and Spectral, both methods rely on the latent representation of images to detect

poisoned samples. AC identifies poisoned samples through clustering in the latent space, considering smaller clusters as likely to contain poisoned data. Spectral detects outliers in the latent space to identify such samples. However, with a poisoning ratio of 5% for Tiny ImageNet (200 classes, each class accounts for 0.5%), the poisoned samples become the majority within the target class, breaking the underlying assumptions of both methods and resulting in high ASR values. Additionally, while ABL, DBD, and NAB can defend against certain attacks, they fall short against more sophisticated adversaries, highlighting PDB's robust defense performance.

PDB achieves an excellent balance between defense performance and model utility. Apart from its robust defensive performance, PDB distinguishes itself through its ability to preserve benign accuracy. Unlike ABL, DBD, and NAB, which often sacrifice considerable benign accuracy in exchange for reduced ASR, leading to lower DER values, PDB maintains a high DER by effectively managing this trade-off. The preservation of model utility, without compromising defense effectiveness, further solidifies PDB's status as a promising strategy in backdoor defense.

The results demonstrate the superiority of PDB in defending against backdoor attacks. By effectively reducing ASR and maintaining a high DER, PDB stands out as a valuable defense approach for backdoor attack.

4.3 Analysis

Understanding the effect of PDB. To elucidate the underlying mechanism of PDB, we delve into the impact of the defensive backdoor by analyzing the T-SNE embeddings and the Trigger Activation Change (TAC). TAC, adapted from Zheng et al. [52], is designed to measure the change of activation values for each neuron when comparing maliciously poisoned samples to their benign counterparts. Let ϕ be a feature extractor which maps an input image x to the latent activations. For an input image x, we can construct the malicious poisoned sample $x \oplus \Delta$. In PDB, a defensive trigger is added to the malicious poisoned sample, crafting sample $x \oplus \Delta \oplus \Delta_1$, aiming to suppress the malicious backdoor. Therefore, for dataset D, we define

$$\text{TAC w/o } \Delta_1 = \frac{\sum_{x \in D} (\phi(x \oplus \Delta) - \phi(x))}{|D|}, \tag{4}$$

$$\text{TAC w/ } \Delta_1 = \frac{\sum_{x \in D} (\phi(x \oplus \Delta \oplus \Delta_1) - \phi(x))}{|D|}. \tag{5}$$

In Figure 4, we present the visualization results for the BadNets attack on the CIFAR-10 dataset, utilizing a poisoning ratio of 5% alongside a PreAct-ResNet architecture. The illustration reveals that planting a defensive trigger to the inputs prompts a shift in the feature space, resulting in the formation of new clusters and effectively alleviating the backdoor effect. Moreover, the TAC analysis for both the initial and final blocks demonstrates that the incorporation of a defensive trigger substantially mitigates the activation changes triggered by the malicious backdoor.

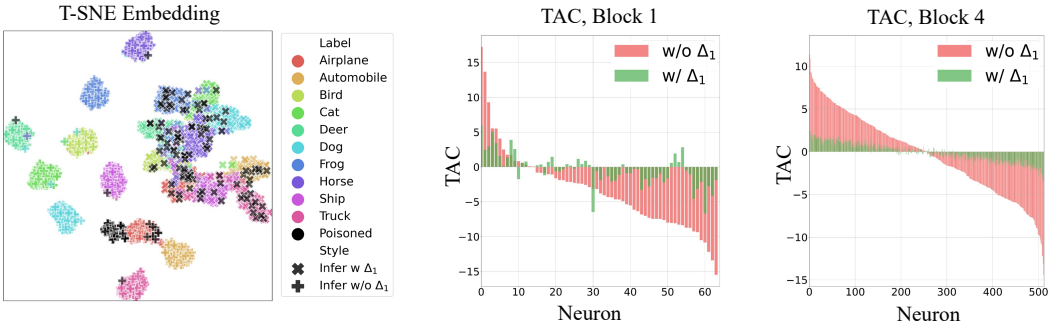

Figure 4: Visualization of T-SNE and TAC for the BadNets attack on CIFAR-10 with a poisoning ratio of 5% and PreAct-ResNet. The T-SNE visualizes features in the 4th block of PreAct-ResNet18, and TAC is calculated for both the 1st and the 4th blocks (4 blocks in total). Neurons are indexed in descending order based on their TAC values without Δ_1.

Defense effectiveness under different poisoning ratios. To investigate the influence of poisoning ratios on defense performance, we evaluate our method against attacks with poisoning ratios ranging from 1% to 40% on CIFAR-10 with PreAct-ResNet18. The results are summarized in Table 3, from which we can find that the proposed method can consistently mitigate malicious backdoor effect across a wide range of poisoning ratios. For a more comprehensive evaluation of the influence of the poisoning ratio, please refer to **Appendix** B.

Table 3: Defense results (%) under different poisoning ratios on CIFAR-10 and PreAct-ResNet18.

Poisoning ratio →	1%				5%				10%				20%				40%			
Defense →	No Defense		PDB (**Ours**)		No Defense		PDB (**Ours**)		No Defense		PDB (**Ours**)		No Defense		PDB (**Ours**)		No Defense		PDB (**Ours**)	
Attack ↓	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
BadNets [12]	93.14	74.73	91.59	0.31	92.64	88.74	91.08	0.38	91.32	95.03	90.25	0.40	90.17	96.12	89.18	0.16	86.16	97.77	86.32	0.28
Blended [5]	93.76	94.88	91.77	1.20	93.67	99.61	91.36	0.70	93.47	99.92	91.21	0.92	92.92	99.92	90.74	1.51	91.74	99.98	89.04	0.27
BPP [42]	90.81	87.23	90.73	1.38	91.47	99.34	90.43	1.90	90.69	99.78	90.47	1.11	91.45	99.71	90.44	1.29	90.66	99.99	89.22	0.49
Average	92.57	85.61	91.36	0.96	92.59	95.90	90.96	0.99	91.83	98.24	90.64	0.81	91.51	98.58	90.12	0.99	89.52	99.25	88.19	0.34

Training cost comparison. We first analyze the training complexity of PDB and we refer readers to BackdoorBench[44] for the training complexity of other methods. Let C_{sl} be the supervised training complexity. Then, we denote the size of the training dataset and the size of the defensive poisoned dataset by N_{tr} and N_{def}, respectively. Let F be the frequency of sampling defensive poisoned samples. The training complexity of PDB is given by: $O\left(\left(1+\frac{F\cdot N_{def}}{N_{tr}}\right)\cdot C_{sl}\right)$.

To evaluate the empirical runtime, $i.e.$, training time of different defense methods, we conduct experiments against the BadNets attack for the PreAct-ResNet18 architecture on CIFAR-10 and GTSRB, ViT-B-16 for Tiny ImageNet, all with a poisoning ratio of 5%. The experiments are conducted on an RTX 4090Ti GPU, and the results are summarized in Table 4. From Table 4, We can find since $\frac{F\cdot N_{def}}{N_{tr}}$ is set as a small value, the runtime of PDB is not much larger than the baseline ($i.e.$, No Defense). In contrast, the runtime of DBD and NAB are significantly higher due to their reliance on self-supervised and semi-supervised training techniques.

Table 4: Running time (s) comparison of defense methods.

Defense →	No Defense	AC [3]	Spectral [39]	ABL [20]	DBD [15]	NAB [26]	PDB (**Ours**)
GTSRB	801	637	1642	1634	7495	3081	1573
CIFAR-10	919	658	2278	1805	8351	3679	1853
Tiny	2938	2524	9952	6034	26932	11698	4913

Resistance to ALL2ALL attack. We also evaluate PDB for ALL2ALL attacks on CIFAR-10 using PreAct-ResNet18. The poisoning ratio is set to 5% and the target labels for samples with labels y are $(y+2) \mod K$ (different from the defensive target). The experimental results are summarized in Table 5. Notably, PDB achieves the best defending performance, demonstrating superior effectiveness in defending against backdoor attacks with multiple targets.

Table 5: ALL2ALL attack results (%) on CIFAR-10 with PreAct-ResNet18 and poisoning ratio 5.0%.

Defense →	No Defense		AC [3]			Spectral [39]			ABL [20]			DBD [15]			NAB [26]			PDB (**Ours**)		
Attack ↓	ACC	ASR	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [12]	92.50	61.33	90.10	53.7	52.61	**92.33**	57.73	51.72	52.46	59.96	30.66	87.10	4.52	75.70	80.51	62.74	44.00	90.68	**2.72**	**78.40**
Blended [5]	93.51	83.87	91.36	78.56	51.58	**93.72**	84.66	50.00	68.04	35.62	61.39	75.24	26.62	69.49	90.34	79.09	50.80	91.87	**3.95**	**89.14**
SIG [2]	93.52	88.15	91.49	83.07	51.52	**94.02**	88.77	50.00	67.20	59.67	51.08	76.19	20.26	75.28	82.65	83.19	47.04	91.73	**3.13**	**91.62**

Resistance to adaptive attack. In our previous experiments, we assumed that attackers had no knowledge of the defense method. However, when attackers are aware of the deployment of PDB, they may design adaptive attacks to bypass the defense. One straightforward approach is to strengthen the malicious backdoor to counteract the defensive backdoor. To assess our method's resistance to

such adaptive attacks, we evaluate it against BadNets with varying trigger sizes and poisoning ratios, representing different strengths of backdoor attacks. The results, summarized in Table 6, demonstrate that PDB can consistently mitigate backdoor against adaptive attacks with various malicious trigger size and poisoning ratios. Note that to keep the stealthness of malicious backdoor, its poisoning ratio and trigger size is expected to be constrained. However, the defensive backdoor can utilize a large trigger size and high sampling frequency to meet the Principle 4, therefore, mitigating the malicious backdoor.

Table 6: Defense results (%) against adaptive attacks with different poisoning ratios.

Poisoning ratio →	10%				20%				30%			
Defense →	No Defense		PDB (**Ours**)		No Defense		PDB (**Ours**)		No Defense		PDB (**Ours**)	
Malicious trigger size ↓	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
4x4	92.39	96.83	90.66	0.18	91.14	97.67	90.01	0.21	90.38	98.13	89.65	0.49
5x5	93.11	97.69	91.28	0.29	92.79	97.98	90.97	0.28	92.20	98.30	90.02	0.56
6x6	93.26	98.16	91.62	0.27	92.48	98.68	90.83	0.33	92.01	98.83	90.03	0.69
7x7	93.65	98.66	91.46	0.31	93.07	99.03	91.03	0.56	92.56	99.23	90.48	0.67
8x8	93.51	99.24	91.16	0.37	92.82	99.38	91.14	0.58	92.53	99.50	90.27	0.74
9x9	93.45	99.53	91.12	0.51	92.76	99.67	90.84	0.56	92.15	99.72	90.39	0.67
10x10	93.20	99.66	91.37	0.54	93.17	99.74	90.76	0.78	92.58	99.81	90.45	0.82

Appendix structure. Due to page limitations, more experiments and analyses have been moved to the Appendix. The Appendix is structured as follows: In **Appendix** A, we provide the details for the experiments, including the implementation of our method, the parameters, and the setting for all attacks and defense methods. In **Appendix** B, we provide a more comprehensive comparison between our method and baselines across different datasets, poisoning ratios, and model structures. In **Appendix** C, we discuss the influence of key components for PDB, such as triggers, targets, and reserved datasets, and make comparisons to more baselines.

5 Conclusion

In this paper, we propose a proactive approach to defend against malicious backdoor attacks in DNNs. Rather than relying on traditional detection and mitigation pipeline, our method, PDB, leverages the *"home field"* advantage of defenders to inject a defensive backdoor to fight against malicious backdoor. To achieve such a goal, we introduce four essential design properties for an effective defensive backdoor: reversibility, inaccessibility to attackers, minimal impact on model performance, and resistance to other backdoors. By incorporating a defensive backdoor during training, we suppress the impact of malicious backdoors when the defensive trigger is present. Our approach offers several advantages over existing methods. First, it does not rely on accurate detection of poisoned samples and any assumption for attacks, avoiding performance degradation when some poisoned samples evade detection. Second, PDB does not require complex modifications to the training process, minimizing training cost. In summary, PDB provides a novel and effective defense method against backdoor attacks, enhancing the safety and reliability of DNNs.

Limitations and future work. Currently, PDB faces several key limitations. First, its reliance on clean samples presents a practical challenge, prompting the exploration of alternative sources, such as generated data. Second, investigating PDB across diverse machine learning tasks is essential for broader applicability. Addressing these limitations through future research will enhance the defense's effectiveness and facilitate its widespread adoption in safeguarding machine learning systems against backdoor attacks.

Broader impacts. The broader impacts can be considered from both positive and negative perspectives. On the positive side, PDB enhances the security and reliability of DNNs, thereby contributing to the trustworthiness of AI technologies. However, there are potential negative implications that should be considered. The technique could potentially be misused if it falls into the wrong hands, who might use the defensive backdoor mechanism for nefarious purposes.

Acknowledgments and Disclosure of Funding

Baoyuan Wu is supported by Guangdong Basic and Applied Basic Research Foundation (No. 2024B1515020095), National Natural Science Foundation of China (No. 62076213), Shenzhen Science and Technology Program under grants (No. RCYX20210609103057050), and Longgang District Key Laboratory of Intelligent Digital Economy Security. Hongyuan Zha is supported in part by the Shenzhen Key Lab of Crowd Intelligence Empowered Low-Carbon Energy Network (No. ZDSYS20220606100601002). This work is supported by Shenzhen Science and Technology Program under grant No. GXWD20201231105722002-20200901175001001, and No. ZDSYS20211021111415025, and No. JCYJ20210324120011032, and the Guangdong Provincial Key Laboratory of Big Data Computing, the Chinese University of Hong Kong, Shenzhen.

References

[1] Insaf Adjabi, Abdeldjalil Ouahabi, Amir Benzaoui, and Abdelmalik Taleb-Ahmed. Past, present, and future of face recognition: A review. *Electronics*, page 1188, 2020. 1

[2] Mauro Barni, Kassem Kallas, and Benedetta Tondi. A new backdoor attack in cnns by training set corruption without label poisoning. In *International Conference on Image Processing*, 2019. 2, 6, 7, 9, 16, 18, 19, 21, 22, 23

[3] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. Detecting backdoor attacks on deep neural networks by activation clustering. In *Workshop on Artificial Intelligence Safety*, 2019. 1, 2, 3, 6, 7, 9, 16, 17, 18, 19, 23

[4] Weixin Chen, Baoyuan Wu, and Haoqian Wang. Effective backdoor defense by exploiting sensitivity of poisoned samples. In *Conference on Neural Information Processing Systems*, 2022. 1, 3

[5] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv e-prints*, pages arXiv–1712, 2017. 2, 6, 7, 9, 16, 17, 18, 19, 20, 21, 22, 23

[6] Ziyi Cheng, Baoyuan Wu, Zhenya Zhang, and Jianjun Zhao. Tat: Targeted backdoor attacks against visual object tracking. *Pattern Recognition*, 2023. 2

[7] Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. Diffusion models in vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 4

[8] Khoa Doan, Yingjie Lao, Weijie Zhao, and Ping Li. Lira: Learnable, imperceptible and robust backdoor attacks. In *International Conference on Computer Vision*, 2021. 2

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020. 6

[10] Kuofeng Gao, Jiawang Bai, Baoyuan Wu, Mengxi Ya, and Shu-Tao Xia. Imperceptible and robust backdoor attack in 3d point cloud. *IEEE Transactions on Information Forensics and Security*, 19:1267–1282, 2023. 2

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 2020. 4

[12] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, pages 47230–47244, 2019. 2, 6, 7, 9, 16, 17, 18, 19, 20, 21, 22, 23

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, 2016. 1

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, 2016. 6

[15] Kunzhe Huang, Yiming Li, Baoyuan Wu, Zhan Qin, and Kui Ren. Backdoor defense via decoupling the training process. In *International Conference on Learning Representations*, 2022. 1, 2, 3, 6, 7, 9, 16, 17, 18, 19, 23

[16] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv e-prints*, 2013. 4

[17] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

[18] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 2015. 6

[19] Songze Li and Yanbo Dai. Backdoorindicator: Leveraging ood data for proactive backdoor detection in federated learning. *arXiv e-prints*, 2024. 25, 26

[20] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. Anti-backdoor learning: Training clean models on poisoned data. In *Conference on Neural Information Processing Systems*, 2021. 1, 2, 3, 6, 7, 9, 16, 17, 18, 19, 23

[21] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. Neural attention distillation: Erasing backdoor triggers from deep neural networks. In *International Conference on Learning Representations*, 2021. 3, 17, 22

[22] Yuezun Li, Yiming Li, Baoyuan Wu, Longkang Li, Ran He, and Siwei Lyu. Invisible backdoor attack with sample-specific triggers. In *International Conference on Computer Vision*, 2021. 2, 6, 7, 16, 17, 18, 19, 20, 21, 22

[23] Weilin Lin, Li Liu, Shaokui Wei, Jianze Li, and Hui Xiong. Unveiling and mitigating backdoor vulnerabilities based on unlearning weight changes and backdoor activeness. *arXiv e-prints*, 2024. 3

[24] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *Research in Attacks, Intrusions, and Defenses*, 2018. 3, 17, 22

[25] Liangkai Liu, Sidi Lu, Ren Zhong, Baofu Wu, Yongtao Yao, Qingyang Zhang, and Weisong Shi. Computing systems for autonomous driving: State of the art and challenges. *IEEE Internet of Things Journal*, pages 6469–6486, 2020. 1

[26] Min Liu, Alberto Sangiovanni-Vincentelli, and Xiangyu Yue. Beating backdoor attack at its own game. In *International Conference on Computer Vision*, 2023. 1, 2, 3, 6, 7, 9, 16, 17, 18, 19, 23

[27] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *Network and Distributed System Security Symposium*, 2018. 6, 7, 16, 17, 18, 19, 20, 22

[28] Rui Min, Zeyu Qin, Li Shen, and Minhao Cheng. Towards stable backdoor purification through feature shift tuning. In *Advances in Neural Information Processing Systems*, 2023. 3

[29] Preetum Nakkiran, Behnam Neyshabur, and Hanie Sedghi. The deep bootstrap framework: Good online learners are good offline generalizers. In *International Conference on Learning Representations*, 2021. 21

[30] Tuan Anh Nguyen and Anh Tuan Tran. Wanet - imperceptible warping-based backdoor attack. In *International Conference on Learning Representations*, 2021. 2, 6, 7, 16, 17, 18, 19, 20, 22

[31] Xiangyu Qi, Tinghao Xie, Yiming Li, Saeed Mahloujifar, and Prateek Mittal. Revisiting the assumption of latent separability for backdoor defenses. In *International Conference on Learning Representations*, 2023. 2

[32] Xiangyu Qi, Tinghao Xie, Jiachen T Wang, Tong Wu, Saeed Mahloujifar, and Prateek Mittal. Towards a proactive {ML} approach for detecting backdoor poison samples. In *USENIX Security Symposium*, 2023. 3

[33] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Conference on Neural Information Processing Systems*, 2018. 2

[34] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015. 6

[35] Zhengyao Song, Yongqiang Li, Danni Yuan, Li Liu, Shaokui Wei, and Baoyuan Wu. Wpda: Frequency-based backdoor attack with wavelet packet decomposition. *arXiv e-prints*, 2024. 2

[36] Hossein Souri, Liam Fowl, Rama Chellappa, Micah Goldblum, and Tom Goldstein. Sleeper agent: Scalable hidden trigger backdoors for neural networks trained from scratch. In *Conference on Neural Information Processing Systems*, 2022. 2

[37] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *International Joint Conference on Neural Networks*, 2011. 6

[38] J-Donald Tournier, Robert Smith, David Raffelt, Rami Tabbara, Thijs Dhollander, Maximilian Pietsch, Daan Christiaens, Ben Jeurissen, Chun-Hung Yeh, and Alan Connelly. Mrtrix3: A fast, flexible and open software framework for medical image processing and visualisation. *Neuroimage*, page 116137, 2019. 1

[39] Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. In *Advances in Neural Information Processing Systems*, 2018. 6, 7, 9, 16, 17, 18, 19, 23

[40] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *Symposium on Security and Privacy*, 2019. 17, 22

[41] Ruotong Wang, Hongrui Chen, Zihao Zhu, Li Liu, Yong Zhang, Yanbo Fan, and Baoyuan Wu. Robust backdoor attack with visible, semantic, sample-specific, and compatible triggers. *arXiv e-prints*, 2023. 2

[42] Zhenting Wang, Juan Zhai, and Shiqing Ma. Bppattack: Stealthy and efficient trojan attacks against deep neural networks via image quantization and contrastive adversarial learning. In *Conference on Computer Vision and Pattern Recognition*, 2022. 6, 7, 9, 16, 17, 18, 19, 20, 22, 23

[43] Shaokui Wei, Mingda Zhang, Hongyuan Zha, and Baoyuan Wu. Shared adversarial unlearning: Backdoor mitigation by unlearning shared adversarial examples. In *Advances in Neural Information Processing Systems*, 2023. 3, 7

[44] Baoyuan Wu, Hongrui Chen, Mingda Zhang, Zihao Zhu, Shaokui Wei, Danni Yuan, and Chao Shen. Backdoorbench: A comprehensive benchmark of backdoor learning. In *Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. 1, 6, 9, 16

[45] Baoyuan Wu, Shaokui Wei, Mingli Zhu, Meixi Zheng, Zihao Zhu, Mingda Zhang, Hongrui Chen, Danni Yuan, Li Liu, and Qingshan Liu. Defenses in adversarial machine learning: A survey. *arXiv e-prints*, 2023. 1, 3, 4

[46] Baoyuan Wu, Hongrui Chen, Mingda Zhang, Zihao Zhu, Shaokui Wei, Danni Yuan, Mingli Zhu, Ruotong Wang, Li Liu, and Chao Shen. Backdoorbench: A comprehensive benchmark and analysis of backdoor learning. *arXiv e-prints*, 2024. 1, 6

[47] Dongxian Wu and Yisen Wang. Adversarial neuron pruning purifies backdoored deep models. In *Conference on Neural Information Processing Systems*, 2021. 3

[48] Danni Yuan, Shaokui Wei, Mingda Zhang, Li Liu, and Baoyuan Wu. Activation gradient based poisoned sample detection against backdoor attacks. *arXiv e-prints*, 2023. 1

[49] Yi Zeng, Won Park, Z. Morley Mao, and Ruoxi Jia. Rethinking the backdoor attacks' triggers: A frequency perspective. In *International Conference on Computer Vision*, 2021. 2

[50] Yi Zeng, Si Chen, Won Park, Zhuoqing Mao, Ming Jin, and Ruoxi Jia. Adversarial unlearning of backdoors via implicit hypergradient. In *International Conference on Learning Representations*, 2022. 3, 17, 22

[51] Zaixi Zhang, Qi Liu, Zhicai Wang, Zepu Lu, and Qingyong Hu. Backdoor defense via deconfounded representation learning. In *Conference on Computer Vision and Pattern Recognition*, 2023. 2, 3

[52] Runkai Zheng, Rongjun Tang, Jianze Li, and Li Liu. Data-free backdoor removal based on channel lipschitzness. In *European Conference on Computer Vision*, 2022. 3, 8

[53] Runkai Zheng, Rongjun Tang, Jianze Li, and Li Liu. Pre-activation distributions expose backdoor neurons. In *Conference on Neural Information Processing Systems*, 2022. 3

[54] Hong Zhu, Shengzhi Zhang, and Kai Chen. Ai-guardian: Defeating adversarial attacks using backdoors. In *Symposium on Security and Privacy*, pages 701–718. IEEE, 2023. 3

[55] Mingli Zhu, Shaokui Wei, Li Shen, Yanbo Fan, and Baoyuan Wu. Enhancing fine-tuning based backdoor defense with sharpness-aware minimization. In *International Conference on Computer Vision*, 2023. 3, 7, 26

[56] Mingli Zhu, Shaokui Wei, Hongyuan Zha, and Baoyuan Wu. Neural polarizer: A lightweight and effective backdoor defense via purifying poisoned features. In *Advances in Neural Information Processing Systems*, 2023. 3

[57] Mingli Zhu, Siyuan Liang, and Baoyuan Wu. Breaking the false sense of security in backdoor defense through re-activation attack. In *Conference on Neural Information Processing Systems*, 2024. 2

[58] Zihao Zhu, Mingda Zhang, Shaokui Wei, Li Shen, Yanbo Fan Fan, and Baoyuan Wu. Boosting backdoor attack with a learnable poisoning sample selection strategy. *arXiv e-prints*, 2023. 2

[59] Zihao Zhu, Mingda Zhang, Shaokui Wei, Bingzhe Wu, and Baoyuan Wu. Vdc: Versatile data cleanser for detecting dirty samples via visual-linguistic inconsistency. In *International Conference on Learning Representations*, 2024. 3

[60] Zixuan Zhu, Rui Wang, Cong Zou, and Lihua Jing. The victim and the beneficiary: Exploiting a poisoned model to train a clean model on poisoned data. In *International Conference on Computer Vision*, 2023. 1, 2, 3

A Experiment details

In our experiments, we adapted all baselines and settings from BackdoorBench [44]. Moreover, all checkpoints of attack methods are sourced from BackdoorBench and the defense results are aligned with the leaderboard in BackdoorBench if applicable. Below, we outline the details of various backdoor attacks:

A.1 Attack details

- BadNets [12] is one of the earliest works for backdoor learning, which inserts a small patch of fixed pattern to replace some pixels in the image. We use the default setting in BackdoorBench.
- Blended backdoor attack (Blended) [5] uses an alpha-blending strategy to fuse images with fixed patterns. We set $\alpha = 0.2$ as the default in BackdoorBench. Note that a large α causes visual-perceptible changes to clean samples, making the Blended Attack challenging for defense methods.
- Sinusoidal signal backdoor attack (SIG) [2] is a clean-label attack that perturbs clean images in the target label using a sinusoidal signal as the trigger. We use the default setting in BackdoorBench.
- Sample-specific backdoor attack (SSBA) [22] uses an auto-encoder to fuse a trigger into clean samples and generate poisoned samples. We use the default setting in BackdoorBench.
- Warping-based poisoned networks (WaNet) [30] is also a training-controllable attack that perturbs clean samples using a warping function to construct poisoned samples. We use the default setting in BackdoorBench.
- Bppattack (BPP) [42] is also a training-controllable attack that employs image quantization and dithering as the Trojan trigger. We use the default setting in BackdoorBench.
- Trojaning attack on neural networks (TrojanNN) [27] inverses the neural network to generate a general trojan trigger. We use the default setting in BackdoorBench.

Adaptation to data poisoning attack. In our paper, we explore scenarios where attacks can only utilize data poisoning techniques. To facilitate a more comprehensive comparison of our method, we modify attacks originally designed for training-controllable scenarios, removing the training component to adapt them to a data poisoning setting.

A.2 Defense details

Here, we summarize the details of each defense method used:

- AC [3] is a detection method that detects the poisoned sample using the abnormal clustering for poisoned samples. By removing the detected samples, AC can effectively defend against backdoor attack. We use the default setting in BackdoorBench.
- Spectral [39] is a detection method that detects the poisoned sample using the abnormal Spectral Signature for poisoned samples. By removing the detected samples, AC can effectively defend against backdoor attack. We use the default setting in BackdoorBench.
- ABL [20] utilizes the early-learning phenomenon of poisoned samples to detect poisoned samples and then unlearns them to mitigate the backdoor effect. We use the default setting in BackdoorBench.
- DBD [15] divides the training process into three stages and uses self-supervised techniques to detect the poisoned sample and learn a clean model. We use the default setting in BackdoorBench.
- NAB [26] first employs an advanced detection method to filter the poisoned samples. Then, the detected samples are relabeled by employing other techniques and planted with non-adversarial triggers to suppress the backdoor. In this work, we use the detection method from ABL and the self-supervised method from DBD to relabel the samples. For other settings, We use the default setting in BackdoorBench.

- FT finetunes the model on a small, clean, reserved dataset to mitigate the backdoor effect. We use the default setting in BackdoorBench.
- FP [24] is a pruning-based method that prunes neurons according to their activations and then fine-tunes the model to keep clean accuracy. We use the default setting in BackdoorBench.
- NC [40] first optimizes a possible trigger to detect backdoored models. If detected as backdoored, unlearn the optimized trigger. If detected as clean, the model is returned unchanged.
- NAD [21] uses Attention Distillation to mitigate backdoors. We use the default setting in BackdoorBench.
- i-BAU [50] uses adversarial training with UAP and hyper-gradient to mitigate the backdoor. We use the default setting in BackdoorBench.
- PDB (**Ours**) defends backdoor attack by injecting defensive backdoor. We set the reserved dataset size to 10% of the training dataset. The chosen parameters are $\lambda_1 = 1$ and $\lambda_2 = 1$. To enhance the defensive backdoor, each defensive poisoned sample is sampled five times in an epoch, and we set $\tau(\boldsymbol{x}) = \boldsymbol{x} + 0.1 \cdot \boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(0, 1)$. The defensive backdoor utilizes a target mapping function $h(y) = (y + 1) \mod K$, along with a 7×7 patch trigger with pixel value 2 as illustrated in Figure 2.

Adaptation to ViT-B-16. For all experiments on CIFAR-10 and GTSRB, we train the model 100 epochs with batch size 256 for fair comparison. For Tiny ImageNet with ViT-B-16, we consider a fine-tuning task as recommended by BackdoorBench. Specifically, we train each model 10 epochs with batch size 128 and initialize the model with pre-trained weights.

B Additional experiment results

This section provides additional experiment results to supplement the observations claimed in Section 4.

B.1 Main experiments on GTSRB with PreAct-ResNet18

Table 7, 8, and 9 summarize the results of various defense methods against backdoor attacks on the GTSRB dataset using the PreAct-ResNet18 model architecture. These methods were evaluated at different poisoning ratios (1.0%, 5.0%, and 10.0%). Notably, the results demonstrate that PDB effectively mitigates backdoor attacks, consistently achieving top-2 defense performance across all cases.

Table 7: Results on GTSRB with PreAct-ResNet18 and poisoning ratio 5.0%.

Defense →	No Defense		AC [3]			Spectral [39]			ABL [20]			DBD [15]			NAB [26]			PDB (**Ours**)		
Attack ↓	ACC	ASR	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [12]	97.3	57.95	**97.55**	84.67	50.0	96.36	92.51	49.53	95.8	0.00	78.22	78.76	0.00	69.70	84.06	92.32	43.38	96.75	**0.00**	**78.70**
Blended [5]	98.84	99.93	97.48	99.73	49.42	**97.58**	99.83	49.41	44.18	**0.00**	72.63	83.08	100.00	42.12	87.94	99.34	44.84	97.00	0.03	**99.03**
SSBA [22]	98.11	99.34	97.16	98.3	50.05	97.31	97.96	50.29	80.2	**0.01**	90.71	81.57	99.96	41.73	94.14	13.57	90.90	**97.61**	0.02	**99.41**
WaNet [30]	97.42	92.85	95.91	54.1	68.62	96.85	62.12	65.08	1.45	99.40	2.02	84.09	0.01	89.76	84.96	9.53	85.43	**96.92**	**0.00**	**96.17**
BPP [42]	98.21	98.97	**97.64**	83.67	57.37	97.36	87.3	55.42	12.77	100.00	7.28	82.52	0.00	91.64	89.39	0.00	95.08	97.46	**0.00**	**99.11**
Trojan [27]	98.55	100.0	97.00	100.0	49.22	**97.94**	100.0	49.7	83.42	0.00	92.43	73.17	0.01	87.30	89.32	12.10	89.33	97.73	**0.00**	**99.59**
Average	98.07	91.51	97.12	86.74	54.11	97.23	89.95	53.24	52.97	33.23	57.22	80.53	33.33	70.38	88.3	37.81	74.83	**97.24**	**0.01**	**95.33**

Table 8: Results on GTSRB with PreAct-ResNet18 and poisoning ratio 10.0%.

Defense →	No Defense		AC [3]			Spectral [39]			ABL [20]			DBD [15]			NAB [26]			PDB (**Ours**)		
Attack ↓	ACC	ASR	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [12]	97.24	59.25	95.53	95.94	49.14	96.83	94.22	49.79	93.82	0.00	77.92	82.75	0.00	72.38	95.34	0.03	78.66	**97.05**	**0.00**	**79.53**
Blended [5]	98.58	99.99	98.61	100.0	50.0	**98.76**	100.0	50.0	32.99	**0.00**	67.2	81.73	99.97	41.58	95.91	2.87	97.23	96.98	0.02	**99.18**
SSBA [22]	97.98	99.56	96.60	99.47	49.36	**97.57**	99.55	49.8	63.21	0.58	82.1	91.11	99.94	46.56	96.82	0.85	98.77	96.42	**0.00**	**99.00**
WaNet [30]	97.74	94.25	96.36	72.65	60.11	96.94	74.82	59.32	21.35	84.09	16.88	84.03	0.00	90.27	82.88	75.26	52.07	**97.36**	**0.00**	**96.94**
BPP [42]	97.43	99.9	97.43	88.69	55.6	**97.73**	93.32	53.29	10.51	99.94	6.54	86.47	100.00	44.52	81.92	32.01	76.19	97.40	**0.00**	**99.93**
Trojan [27]	98.57	100.0	96.76	100.0	49.1	97.73	100.0	49.58	78.19	0.00	89.81	87.09	100.00	44.26	**97.75**	0.06	**99.56**	96.90	**0.00**	99.17
Average	97.92	92.16	96.88	92.79	52.22	**97.59**	93.65	51.96	50.01	30.77	56.74	85.53	66.65	56.60	91.77	18.51	83.75	97.02	**0.00**	**95.63**

Table 9: Results on GTSRB with PreAct-ResNet18 and poisoning ratio 1.0%.

Defense →	No Defense		AC [3]			Spectral [39]			ABL [20]			DBD [15]			NAB [26]			PDB (**Ours**)		
Attack ↓	ACC	ASR	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [12]	98.35	50.26	**98.42**	81.73	50.00	97.32	86.26	49.49	0.48	100.00	1.06	84.29	0.00	68.10	87.16	3.51	67.78	96.62	**0.00**	**74.27**
Blended [5]	98.8	95.67	97.10	94.34	49.82	97.70	94.02	50.28	28.22	**2.27**	61.41	83.4	99.97	42.30	86.6	97.14	43.9	**98.20**	6.13	**94.47**
SSBA [22]	98.75	94.54	96.83	86.5	53.06	97.28	87.65	52.71	4.12	68.09	15.91	88.6	**0.00**	92.20	88.08	95.41	44.67	**98.06**	0.49	**96.68**
WaNet [30]	97.08	62.24	96.48	4.14	78.75	**97.35**	28.02	67.11	29.75	33.21	30.86	87.15	0.00	76.16	85.29	2.7	73.87	97.30	**0.00**	**81.12**
BPP [42]	98.26	62.21	97.11	40.91	60.08	**98.12**	64.61	49.93	6.41	20.48	24.94	78.41	0.00	71.18	87.55	55.82	47.85	96.52	**0.00**	**80.24**
Trojan [27]	98.17	100.0	97.38	100.0	49.60	**97.72**	100.0	49.77	21.69	0.00	61.76	81.74	0.00	91.78	84.39	99.82	43.2	96.83	**0.00**	**99.33**
Average	98.23	77.49	97.22	67.94	56.89	**97.58**	76.76	53.22	15.11	37.34	32.66	83.93	16.66	73.62	86.51	59.07	53.54	97.26	**1.10**	**87.69**

B.2 Main experiments on CIFAR-10 with PreAct-ResNet18

Table 10 and 11 summarize the results of different defense methods against backdoor attacks on the CIFAR-10 dataset using the PreAct-ResNet18 model architecture. These methods were evaluated at different poisoning ratios (1.0% and 10.0%). Notably, they achieved the top-2 lowest ASR in 12 out of 14 cases.

Table 10: Results on CIFAR-10 with PreAct-ResNet18 and poisoning ratio 10.0%.

Defense →	No Defense		AC [3]			Spectral [39]			ABL [20]			DBD [15]			NAB [26]			PDB (**Ours**)		
Attack ↓	ACC	ASR	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [12]	91.32	95.03	88.80	86.23	53.14	89.98	92.41	50.64	83.32	**0.00**	93.52	89.65	1.28	96.04	75.51	0.08	89.57	**90.25**	0.40	**96.78**
Blended [5]	93.47	99.92	88.52	99.72	47.63	90.35	99.84	48.48	77.3	0.73	91.51	69.91	99.98	38.22	86.25	**0.12**	96.29	**91.21**	0.92	**98.37**
SIG [2]	84.48	98.27	82.41	94.61	50.79	83.01	92.27	52.26	57.8	0.00	85.79	60.67	100.0	38.10	83.07	17.54	89.66	**91.10**	**0.00**	**99.13**
SSBA [22]	92.88	97.86	90.00	96.23	49.37	89.63	90.5	52.05	80.79	**0.00**	92.88	63.5	99.51	35.31	88.77	1.84	95.95	**90.95**	0.19	**97.87**
WaNet [30]	91.25	89.73	91.93	96.8	50.0	**91.94**	90.17	50.0	83.19	**0.00**	90.84	80.9	6.61	86.39	80.01	0.88	88.81	90.92	0.29	**94.56**
BPP [42]	90.69	99.78	89.29	99.73	49.32	**92.34**	99.72	50.03	78.55	13.77	86.94	68.65	100.0	38.98	75.66	**0.21**	92.27	90.47	1.11	**99.22**
Trojan [27]	93.42	100.0	89.75	99.97	48.18	89.70	100.0	48.14	11.07	100.00	8.82	66.23	100.0	36.40	86.17	1.49	95.63	**91.24**	**0.59**	**98.62**
Average	91.07	97.23	88.67	96.19	49.78	89.56	94.99	50.23	67.43	16.36	78.61	71.36	72.48	52.78	82.21	3.17	92.60	**90.88**	**0.50**	**97.79**

Table 11: Results on CIFAR-10 with PreAct-ResNet18 and poisoning ratio 1.0%.

Defense →	No Defense		AC [3]			Spectral [39]			ABL [20]			DBD [15]			NAB [26]			PDB (**Ours**)		
Attack ↓	ACC	ASR	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [12]	93.14	74.73	88.88	27.18	71.65	**92.62**	74.2	50.01	72.81	53.20	50.6	78.09	2.99	78.35	90.85	1.43	85.50	91.59	**0.31**	**86.44**
Blended [5]	93.76	94.88	89.77	86.09	52.40	**93.27**	93.32	50.53	66.26	**0.17**	83.61	70.18	8.04	81.63	86.65	3.23	92.27	91.77	1.20	**95.84**
SIG [2]	93.82	83.4	90.0	74.81	52.38	**93.09**	85.91	49.63	64.33	**0.00**	76.96	75.01	67.82	48.38	88.73	2.40	87.96	90.21	1.19	**89.30**
SSBA [22]	93.43	73.44	89.61	27.92	70.85	**92.71**	54.79	58.97	83.11	56.33	53.4	78.52	1.13	78.70	85.11	64.51	50.31	91.56	**1.02**	**85.28**
WaNet [30]	90.65	12.63	89.53	4.87	53.32	**92.79**	8.81	51.91	65.43	53.43	37.39	79.74	4.60	48.56	85.91	21.92	47.63	91.45	**0.82**	**55.91**
BPP [42]	90.81	87.23	89.13	16.74	84.40	**91.94**	21.02	83.11	55.61	11.78	70.13	86.3	6.91	87.91	91.06	45.62	70.81	90.73	**1.38**	**92.89**
Trojan [27]	93.58	99.97	89.72	99.53	48.29	**92.83**	99.61	49.8	24.73	100.00	15.58	74.49	99.99	40.45	88.52	5.67	94.62	90.01	**1.90**	**97.25**
Average	92.74	75.18	89.52	48.16	61.90	**92.75**	62.52	56.28	61.75	39.27	55.38	77.48	27.36	66.28	88.12	20.68	75.58	91.05	**1.12**	**86.13**

B.3 Main experiments on CIFAR-10 with VGG19-BN

Table 12, 13, and 14 summarize the results of different defense methods against backdoor attacks on the CIFAR-10 dataset using the VGG19-BN model architecture. These methods were evaluated at different poisoning ratios (1.0%, 5%, and 10.0%). Impressively, PDB achieves the top-2 lowest ASR in 20 out of 21 cases.

Table 12: Results on CIFAR-10 with VGG19-BN and poisoning ratio 5.0%.

Defense →	No Defense		AC [3]			Spectral [39]			ABL [20]			DBD [15]			NAB [26]			PDB (**Ours**)		
Attack ↓	ACC	ASR	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [12]	91.19	93.92	85.75	88.49	50.0	9.97	99.99	9.39	**90.35**	21.18	85.95	61.15	2.61	80.63	75.28	2.79	87.61	87.43	**1.17**	**94.50**
Blended [5]	92.24	99.43	**87.18**	97.82	48.28	10.0	100.0	8.88	75.92	**0.12**	91.50	52.72	99.99	30.24	76.36	6.52	88.52	86.14	1.09	**96.12**
SIG [2]	91.91	97.23	86.30	97.11	47.26	70.18	96.84	39.33	84.29	0.00	94.81	54.16	99.36	31.13	76.65	0.64	90.66	**88.77**	**0.00**	**97.05**
SSBA [22]	91.53	90.39	85.93	74.07	55.36	10.0	100.0	9.23	79.72	38.47	70.06	48.43	97.05	28.45	77.14	12.84	81.58	**87.34**	**4.87**	**90.67**
WaNet [30]	87.42	94.32	86.03	52.47	70.23	10.0	100.0	11.29	69.85	98.71	41.22	56.72	19.82	71.90	62.73	55.31	57.16	**87.87**	**2.15**	**96.09**
BPP [42]	89.3	98.3	86.50	72.54	61.48	10.0	100.0	10.35	39.62	**0.00**	74.31	59.82	17.05	75.89	48.10	84.27	36.42	**87.96**	0.07	**98.45**
Trojan [27]	92.26	99.99	86.74	99.97	47.25	10.71	97.26	10.59	76.25	0.00	91.99	47.8	100.00	27.77	**88.00**	2.68	96.53	87.61	**0.00**	**97.67**
Average	90.84	96.23	86.35	83.21	54.26	18.69	99.16	14.15	73.71	22.64	78.55	54.4	62.27	49.43	72.04	23.58	76.92	**87.59**	**1.33**	**95.79**

Table 13: Results on CIFAR-10 with VGG19-BN and poisoning ratio 1.0%.

Defense →	No Defense		AC [3]			Spectral [39]			ABL [20]			DBD [15]			NAB [26]			PDB (**Ours**)		
Attack ↓	ACC	ASR	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [12]	91.51	79.87	86.58	9.01	82.96	**89.80**	55.54	61.31	79.32	88.57	43.9	54.81	11.12	66.02	76.48	5.72	79.56	88.67	**1.34**	**87.84**
Blended [5]	91.83	94.19	87.17	85.10	52.21	**90.07**	93.07	49.68	85.11	95.13	46.64	55.09	**0.00**	78.73	67.57	10.40	79.76	87.46	4.38	**92.72**
SIG [2]	92.11	92.79	87.22	79.34	54.28	**89.10**	1.36	**94.21**	87.37	92.86	47.63	55.92	**0.00**	78.3	76.45	88.38	44.38	88.31	8.30	90.34
SSBA [22]	91.9	38.08	86.90	10.61	61.23	**89.82**	24.12	55.94	77.15	61.7	42.62	58.45	12.20	46.21	71.51	44.13	39.80	86.02	**2.24**	**64.98**
WaNet [30]	90.12	23.47	86.56	5.19	57.36	10.00	100.00	9.94	10.03	100.0	9.96	55.8	15.22	36.97	61.62	22.64	36.17	**88.63**	**1.52**	**60.23**
BPP [42]	89.2	47.71	86.43	4.64	70.15	10.00	100.00	10.40	84.03	11.11	65.72	56.81	7.67	53.83	75.22	26.57	53.58	**88.36**	**1.36**	**72.76**
Trojan [27]	91.97	99.92	87.57	99.68	47.92	**90.47**	28.72	84.85	80.36	99.98	44.2	55.68	99.87	31.88	68.77	**6.37**	85.18	87.55	7.70	**93.90**
Average	91.23	68.0	86.92	41.94	60.87	67.04	57.54	52.33	71.91	78.48	42.95	56.08	20.87	55.99	71.09	29.17	59.78	**87.86**	**3.84**	**80.40**

Table 14: Results on CIFAR-10 with VGG19-BN and poisoning ratio 10.0%.

Defense →	No Defense		AC [3]			Spectral [39]			ABL [20]			DBD [15]			NAB [26]			PDB (**Ours**)		
Attack ↓	ACC	ASR	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [12]	90.42	94.43	84.81	93.83	47.50	10.00	100.0	9.79	**89.50**	12.16	90.68	56.35	20.95	69.71	37.38	1.54	69.92	86.03	**1.47**	**94.29**
Blended [5]	91.91	99.5	86.21	99.17	47.32	10.00	100.0	9.04	85.18	4.60	94.08	46.97	99.97	27.53	35.24	3.01	69.91	**88.04**	**0.24**	**97.69**
SIG [2]	83.48	98.87	79.00	99.8	47.76	27.90	98.6	22.34	31.68	0.00	73.53	43.4	99.97	29.96	**80.64**	6.48	94.77	79.48	**0.00**	**97.43**
SSBA [22]	90.85	95.11	85.54	90.61	49.60	10.00	100.0	9.57	**86.27**	**0.16**	**95.19**	51.21	98.81	30.18	74.21	2.64	87.91	85.82	0.70	94.69
WaNet [30]	84.58	96.49	85.05	80.78	57.86	10.00	100.0	12.71	68.10	99.82	41.76	56.87	36.84	65.97	68.73	36.18	72.23	**86.99**	**2.41**	**97.04**
BPP [42]	89.32	99.79	86.62	93.22	51.93	**90.00**	99.16	50.32	72.09	100.00	41.38	54.07	99.96	32.38	52.92	89.00	37.19	87.79	**0.12**	**99.07**
Trojan [27]	91.57	100.0	87.07	99.97	47.77	10.00	100.0	9.22	81.70	0.00	95.06	46.6	100.0	27.51	80.45	2.21	93.33	**87.73**	**0.00**	**98.08**
Average	88.88	97.74	84.90	93.91	49.96	23.99	99.68	17.57	73.50	30.96	75.96	50.78	79.5	40.46	61.37	20.15	75.04	**85.98**	**0.71**	**96.90**

B.4 Main experiments on Tiny ImageNet with ViT-B-16

To demonstrate the effectiveness and scalability of PDB, we evaluate our proposed method against backdoor attacks on the Tiny ImageNet dataset using the ViT-B-6 model architecture. We consider different poisoning ratios (1.0%, 5%, and 10.0%). The results, presented in Table 15, highlight our method's ability to effectively mitigate backdoor attacks for a large dataset and a large model.

Table 15: Results on Tiny ImageNet with ViT-B-16.

Poisoning ratio →	10%				5%				1%			
Defense →	No Defense		PDB (**Ours**)		No Defensel		PDB (**Ours**)		No Defense		PDB (**Ours**)	
Attack ↓	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
BadNets [12]	73.96	99.79	72.88	0.01	76.15	99.72	73.71	0.00	76.98	97.45	74.87	0.01
Blended [5]	75.28	99.93	73.85	0.00	76.00	99.83	72.70	0.00	77.39	98.03	74.24	0.00
SSBA [22]	76.37	99.28	74.27	0.00	75.30	99.86	72.65	0.00	76.05	88.96	74.62	0.00
WaNet [30]	62.68	99.61	74.46	0.00	60.90	99.73	72.82	0.01	63.40	95.87	74.49	0.05
BPP [42]	61.81	99.82	72.83	0.00	63.08	99.68	73.30	0.00	62.98	94.21	73.84	0.00
Trojan [27]	75.16	99.86	72.72	0.00	74.98	99.76	73.00	0.00	77.15	99.07	75.01	0.00
Average	70.88	99.72	73.50	0.00	71.07	98.86	73.03	0.00	72.33	95.03	74.51	0.01

B.5 Experiments on invisible backdoor attack and low-poisoning ratio attack

As aforementioned, the proposed method, PDB, does not rely on specific assumptions about the type of attack, making it effective for defending against both invisible backdoor attacks and attacks with low poisoning ratios. To demonstrate this effectiveness, we conducted experiments using low poisoning ratios (0.5% and 0.1%) for both *Visible* and *Invisible* attacks. The results are summarized in Table 16, from which we can find that PDB can consistently mitigate backdoor attacks.

Table 16: Results on PreAct-ResNet18 and CIFAR10 for invisible and low poisoning ratio attacks

	Trigger Type →	Visible	Visible	Invisible	Invisible	Invisible	Invisible	Invisible	Invisible	Invisible	Invisible
Attack →		BadNet	BadNet	Blended	Blended	Sig	Sig	SSBA	SSBA	WaNet	WaNet
Poisoning ratio ↓	Defense ↓	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
0.10%	No Defense	93.61	1.23	93.80	56.11	93.73	41.27	93.89	1.62	92.18	0.78
0.10%	PDB	91.55	0.87	91.91	0.36	91.59	0.39	91.72	0.42	91.87	0.89
0.50%	No Defense	93.76	50.06	93.68	93.30	93.80	82.43	93.41	35.67	91.27	1.12
0.50%	PDB	91.62	0.60	91.66	0.31	91.72	0.12	91.65	0.54	91.72	0.92

C Discussion and additional analysis

C.1 Influences of reserved dataset

We explore the impact of the reserved dataset on defense performance, considering both dataset size and source:

- **Dataset size.** We investigate the influence of reserved dataset size using the CIFAR-10 dataset, a 5% poisoning ratio, and the PreAct-ResNet architecture. Figure 5 summarizes the results, demonstrating that our proposed method effectively mitigates malicious backdoor effects across a wide range of reserved dataset sizes. Notably, increasing the reserved dataset significantly improves the accuracy of our method.

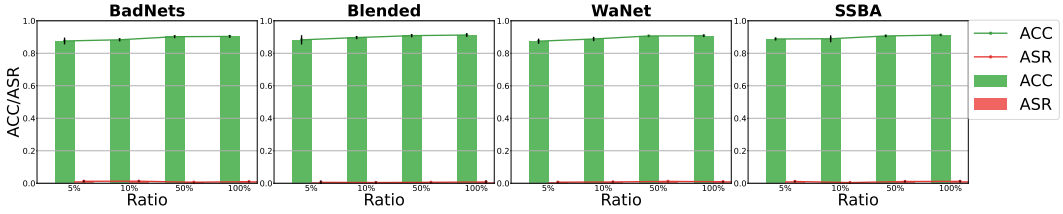

Figure 5: Defense results with different sizes of the reserved dataset. The ratio represents the ratio of the reserved dataset compared with the whole training dataset. Note that the Defense ASR is below 1% and may be barely visible.

- **Defense with generated dataset.** As generative models have evolved, incorporating the generated dataset as an additional source for backdoor defense has become practical and reasonable. To explore the source of this reserved dataset, we assess our method using the generated data CIFAR-5m from Nakkiran et al. [29]. with DDPM. Our findings, summarized in Table 17, demonstrate that the advanced generative model can indeed supply a sufficient dataset for applying our defense strategy.

Table 17: Defense results using generated dataset on CIFAR-10 and PreAct-ResNet18 (%).

Attack →	BadNets [12]		Blended [5]		SIG [2]		SSBA [22]		Average	
Defense ↓	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
No Defense	92.64	88.74	93.67	99.61	93.64	97.09	93.27	94.91	93.31	95.09
PDB (**Ours**)	91.08	0.38	91.36	0.70	91.79	0.06	91.58	0.46	91.45	0.40
PDB (**Ours**)+Generate Dataset	90.17	0.48	90.90	0.97	91.33	0.00	91.49	0.56	91.24	0.78

C.2 Influences of the trigger and target

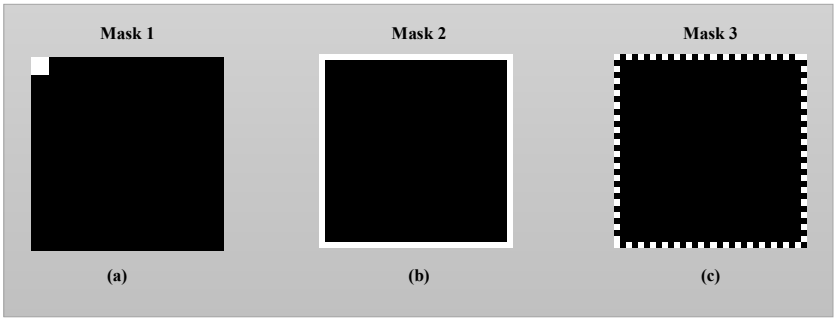

Figure 6: The masks of patched triggers.

In this section, we explore the impact of trigger design and target assignment strategy. Specifically, we evaluate different trigger configurations using a patch trigger with three distinct masks (illustrated in Figure 6). Additionally, we consider two target assignment strategies, *i.e.*, $h_1(y) = (y+1) \mod K$ and $h_2(\phi(\boldsymbol{x})) = -\phi(\boldsymbol{x})$, where y is the hard label and $\phi(\boldsymbol{x})$ is the logits for the model output. Our experiments, detailed in Table 18, demonstrate the effectiveness of our method across various defensive backdoor designs.

Table 18: Results on PDB with different configurations.

Configuration →			Config 1 (a+h_1)		Config 2 (b+h_1)		Config 3 (c+h_1)		Config 4 (a+h_2)	
Defense →	No Defense		PDB (**Ours**)		PDB (**Ours**)		PDB (**Ours**)		PDB (**Ours**)	
Attack ↓	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
BadNets [12]	92.64	88.74	**91.08**	**0.38**	90.09	0.40	90.89	0.76	89.93	0.81
Blended [5]	93.67	99.61	91.36	0.70	89.20	0.04	90.45	0.03	**91.96**	**0.00**
SIG [2]	93.64	97.09	91.79	0.06	91.17	0.10	**92.13**	0.38	91.51	**0.00**
SSBA [22]	93.27	94.91	91.58	0.46	91.03	0.21	89.58	0.37	**91.85**	**0.09**
Average	93.31	95.09	**91.45**	0.40	90.37	0.19	90.76	0.38	91.31	**0.23**

Special case study: Same target label for malicious backdoor and defensive backdoor. While our trigger remains inaccessible to the attacker, the target assignment strategy could potentially be stolen or coincidentally used by the attacker. In this scenario, we conduct experiments where both the attacker and the defender employ the same target assignment strategy but different triggers. Our results demonstrate that our method remains effective in such cases. Specifically, assuming both the attacker and defender choose $h(y) = (y+1) \mod K$, we summarize the results in Table 19, highlighting our method's resilience even when the attacker uses the same target label as the defender.

Table 19: Results on attacks with the same target label as defensive backdoor.

Poisoning Ratio →	5%				10%			
Defense →	No Defense		PDB (**Ours**)		No Defense		PDB (**Ours**)	
Attack ↓	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
BadNet	92.54	65.85	91.37	1.01	91.89	74.42	91.26	1.00
Blended [5]	93.71	83.68	92.28	1.64	93.74	86.85	92.03	1.99
BPP [42]	90.92	86.42	91.89	1.45	91.49	87.71	91.95	1.57
SIG [2]	93.73	88.02	92.02	1.59	93.40	90.61	91.91	1.29
Average	92.73	80.99	91.89	1.42	92.63	84.90	91.79	1.46

C.3 Influences of augmentation

In this section, we explore the impact of augmentation on enhancing defensive backdoors using the CIFAR-10 dataset, a 5% poisoning ratio, and the PreAct-ResNet architecture. We specifically investigate Gaussian noise augmentation by setting $\tau(\boldsymbol{x}) = \boldsymbol{x} + \alpha * \boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(0, 1)$ determines the augmentation intensity. The results are summarized in Figure 7. Notably, even without any augmentation ($\alpha = 0$), PDB exhibits significant efficacy, likely due to the robustness of the defense mechanisms and our controlled training injection process. Furthermore, as augmentation strength increases, the ASR decreases, albeit at the cost of reduced ACC, indicating a tradeoff between augmentation intensity and model performance.

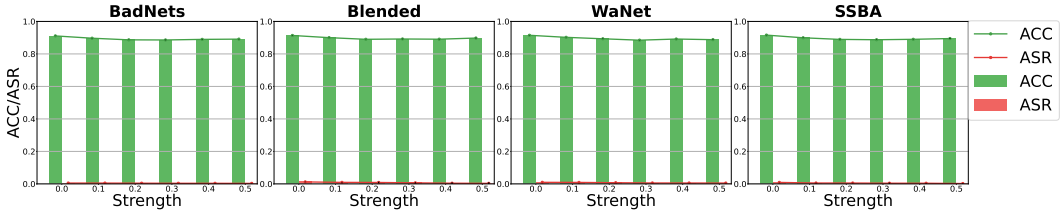

Figure 7: Defense results with different strength of augmentation.

C.4 Comparison with post-training methods

Given a reserved dataset, the defender may follow a 'poisoned first and mitigation later' manner to train the backdoored model first and employ post-training method to mitigate the backdoor effect. Therefore, to thoroughly evaluate our method, we compare it with SOTA post-training approaches that aim to remove backdoor effects after model training. To ensure fairness, we adopt the common practice of reserving 5% of the training data for post-training evaluation. Our experiments are conducted on the CIFAR-10 dataset using PreAct-ResNet18 with a 5% poisoning ratio. The summarized results in Table 20 demonstrate that our method consistently achieves superior performance in defending against backdoor attacks, highlighting its promising effectiveness.

Table 20: Results on CIFAR-10 with PreAct-ResNet18 and poisoning ratio 5.0%.

Defense →	No Defense		FT			FP [24]			NC [40]			NAD [21]			i-BAU [50]			PDB (**Ours**)		
Attack ↓	ACC	ASR	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [12]	92.64	88.74	91.22	4.49	91.42	**92.47**	17.47	85.55	89.89	1.17	**92.41**	91.03	4.73	91.20	89.21	0.81	92.25	89.34	**0.66**	92.39
Blended [5]	93.67	99.61	93.25	98.78	50.21	93.46	98.87	50.27	**93.66**	99.61	50.00	93.10	99.06	49.99	85.92	36.33	77.76	90.24	**0.87**	**97.66**
SIG [2]	93.64	97.09	92.84	96.42	49.93	93.27	99.79	49.81	**93.65**	97.09	50.00	92.49	96.98	49.48	86.12	3.50	93.03	90.26	**0.01**	**96.85**
SSBA [22]	93.27	94.91	**93.08**	87.46	53.63	93.05	87.14	53.77	91.04	0.80	**95.94**	92.49	88.63	52.75	89.43	2.30	94.39	90.02	**0.38**	95.64
WaNet [30]	91.76	85.5	**93.47**	31.32	77.09	92.14	26.10	79.7	91.76	85.50	50.00	93.31	50.40	67.55	91.13	6.11	89.38	89.96	**0.90**	**91.40**
BPP [42]	91.47	99.34	**93.37**	3.46	**97.94**	85.98	**3.11**	95.37	91.47	99.34	50.00	93.09	3.53	97.91	91.35	5.72	96.75	90.9	3.17	97.80
Trojan [27]	93.79	99.99	**92.90**	44.22	77.44	83.89	12.11	88.99	92.59	95.06	51.87	92.68	4.24	97.32	89.42	7.49	94.06	89.64	**0.19**	**97.82**
Average	92.89	95.03	**92.88**	52.31	71.09	90.61	49.23	71.92	92.01	68.37	62.89	92.60	49.65	72.31	88.94	8.90	91.09	90.05	**0.88**	**95.65**

C.5 Detailed comparison to NAB

As previously mentioned, the work most relevant to our research is NAB [26], which presents a promising and impressive approach for enhancing backdoor defense methods. In this context, we conduct a detailed analysis to highlight the distinctions between our method, PDB, and NAB.

PDB does not rely on poisoned sample detection. Our method diverges from the conventional "detection-mitigation" pipeline by operating independently of any poison detection methods. In contrast, NAB relies on identifying poisoned samples to deploy its defensive trigger.

PDB does not depend on suspicious sample relabeling. Unlike NAB, our method does not require accurate sample relabeling. In NAB, detected samples are equipped with a defensive trigger and subsequently relabeled. Unfortunately, if a few clean samples are mistakenly labeled as "suspicious," the defensive trigger may function as a malicious one if further relabeling is incorrect. Note that accurately relabeling a sample is getting harder as the dataset gets larger. Therefore, we observe such scenarios appear more frequently in large datasets during our experiments. For instance, when applying the BadNets attack with a 5% poisoning ratio on Tiny ImageNet using ViT-B-16, NAB's model accuracy drops from 73.60% (without a defensive trigger) to 28.88% (with the trigger) due to low relabel correctness and poison detection rates.

C.6 Resistance to ALL2ALL attack

In this section, we compare PDB with other baselines in ALL2ALL attacks on CIFAR-10 using PreAct-ResNet18. The poisoning ratio is set to 5% and 10%. Specifically, the target labels for samples with original labels y are adjusted to $(y+2) \mod K$ (different from the defensive target). The experimental results are summarized in Table 21 and Table 22. Notably, PDB achieves the best defending performance, demonstrating superior effectiveness in countering backdoor attacks with multiple targets.

Table 21: ALL2ALL attack results on CIFAR-10 with PreAct-ResNet18 and poisoning ratio 5.0%.

Defense →	No Defense		AC [3]			Spectral [39]			ABL [20]			DBD [15]			NAB [26]			PDB (**Ours**)		
Attack ↓	ACC	ASR	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [12]	92.5	61.33	90.1	53.7	52.61	**92.33**	57.73	51.72	52.46	59.96	30.66	87.1	4.52	75.70	80.51	62.74	44.0	90.68	**2.72**	**78.40**
Blended [5]	93.51	83.87	91.36	78.56	51.58	**93.72**	84.66	50.0	68.04	35.62	61.39	75.24	26.62	69.49	90.34	79.09	50.8	91.87	**3.95**	**89.14**
SIG [2]	93.52	88.15	91.49	83.07	51.52	**94.02**	88.77	50.0	67.2	59.67	51.08	76.19	20.26	75.28	82.65	83.19	47.04	91.73	**3.13**	**91.62**
BPP [42]	90.92	86.42	91.26	83.67	51.37	**93.72**	87.52	50.0	33.44	19.69	54.62	77.48	2.24	85.37	86.32	83.33	49.24	91.89	**1.45**	**92.48**
Average	92.61	79.94	91.05	74.75	51.77	**93.45**	79.67	50.43	55.28	43.74	49.44	79.0	13.41	76.46	84.96	77.09	47.77	91.54	**2.81**	**87.91**

Table 22: ALL2ALL attack results on CIFAR-10 with PreAct-ResNet18 and poisoning ratio 10.0%.

Defense →	No Defense		AC [3]			Spectral [39]			ABL [20]			DBD [15]			NAB [26]			PDB (**Ours**)		
Attack ↓	ACC	ASR	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [12]	91.82	72.2	88.89	65.73	51.77	**90.54**	72.11	49.4	26.34	27.9	39.41	85.52	5.25	80.32	83.47	60.34	51.76	90.53	**2.49**	**84.21**
Blended [5]	93.54	87.04	91.01	82.12	51.19	**93.83**	87.09	50.0	74.11	56.56	55.53	80.08	22.48	75.55	88.68	79.7	51.24	91.89	**3.02**	**91.18**
SIG [2]	93.64	90.65	91.18	87.14	50.52	**93.58**	90.68	49.97	86.47	82.69	50.4	68.23	42.20	61.52	82.05	83.85	47.6	92.08	**2.68**	**93.20**
BPP [42]	91.49	87.71	91.66	85.97	50.87	**93.99**	89.41	50.0	75.58	74.15	48.82	81.9	2.68	87.72	84.44	81.52	49.57	91.95	**1.57**	**93.07**
Average	92.62	84.4	90.68	80.24	51.09	**92.98**	84.82	49.84	65.62	60.32	48.54	78.93	18.15	76.28	84.66	76.35	50.04	91.61	**2.44**	**90.42**

C.7 Design of defensive trigger and the satisfaction to Principle 4

Here, we summarize the aforementioned experiments and provide a comprehensive discussion about *how to meet Principle 4 (Resistance against other backdoors)* from the following perspectives:

- **Design of defensive trigger:**
  1. **Defensive trigger size:** Trigger size directly contributes to the strength of the defensive backdoor. In Table 23, we evaluate PDB with a square defensive trigger with sizes ranging

from 1x1 to 9x9. From Table 23, we can find that *a larger trigger leads to a stronger defensive backdoor*, resulting in a higher ACC and a lower ASR. However, *as the trigger size increases, it may interfere with the visual content of the image*, leading to a slight decrease in ACC. Notably, as the square trigger has strong visibility, a trigger with size 1x1 can still alleviate the malicious backdoor to some extent.

Table 23: Results on PreAct-ResNet18 with Poisoning Ratio 5% and different defensive trigger size

Attack →	BadNet		Blended		Sig		SSBA		WaNet	
Defensive Trigger size ↓	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
1x1	48.75	6.88	52.34	5.06	53.15	5.22	52.03	6.20	58.04	4.26
2x2	74.39	3.37	81.38	2.71	81.13	2.32	77.49	3.87	76.49	3.98
3x3	86.08	0.26	85.60	0.67	86.94	0.07	87.01	0.49	85.70	0.97
4x4	89.46	0.28	90.07	0.56	90.38	0.07	89.60	0.44	89.98	0.92
5x5	91.51	0.33	92.22	0.31	92.35	0.06	92.14	0.64	92.05	0.97
6x6	90.78	0.32	91.93	0.49	92.04	0.04	91.82	0.41	91.52	0.91
7x7	91.08	0.38	91.36	0.70	91.79	0.06	91.58	0.46	91.47	0.92
8x8	90.48	0.33	91.56	0.40	91.59	0.02	91.41	0.39	91.44	0.86
9x9	90.21	0.32	91.24	0.39	90.79	0.03	90.92	0.32	90.87	0.56

2. **Defensive trigger position:** As aforementioned, the position of the defensive trigger is essential for Principle 3, i.e., minimal impact on model performance. Table 24 shows that triggers placed in different positions (corner, random, and center) achieve a similar effect in defending against backdoor attacks. However, *placing a trigger at the center of an image significantly degrades accuracy, as the trigger masks the core patterns of the image.*

Table 24: Results on PreAct-ResNet18 with Poisoning Ratio 5% and different positions

Attack →	BadNet		Blended		Sig		SSBA		WaNet	
Defensive Trigger Position ↓	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
Corner	91.08	0.38	91.36	0.7	91.79	0.06	91.58	0.46	91.47	0.92
Random	88.79	0.69	90.10	0.81	90.12	0.16	89.39	0.66	89.49	0.97
Center	87.35	0.63	87.82	0.44	88.19	0.06	87.93	0.89	87.70	0.93

3. **Pixel value:** For a square trigger, pixel value is also an important parameter for PDB. In Table 25, we evaluate PDB with a square defensive trigger with different pixel values, from which we can find that *PDB can achieve high effectiveness across different pixel values.*

Table 25: Results on PreAct-ResNet18 with Poisoning Ratio 5% and different pixel values

Attack →	BadNet		Blended		Sig		SSBA		WaNet	
Pixel ↓	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
1.50	90.69	0.57	91.40	0.56	91.74	0.09	91.54	0.60	91.44	0.83
2.00	91.08	0.38	91.36	0.7	91.79	0.06	91.58	0.46	91.47	0.92
2.50	90.99	0.48	91.39	0.50	91.77	0.04	91.48	0.80	91.54	0.54
-0.50	90.94	0.48	91.78	0.91	91.56	0.01	91.64	0.61	91.69	0.60
-1.00	90.84	0.23	92.31	0.07	91.81	0.00	91.85	1.07	91.83	0.90
-1.50	90.86	0.29	91.62	1.00	91.88	0.00	91.43	0.66	91.86	0.78

- **Backdoor enhancement strategy during training:**
  1. **Increasing sampling frequency:** Given a fixed number of defensive poisoned samples, the defensive backdoor can be further enhanced by increasing the sampling frequency of poisoned samples, forcing the model to pay more attention to defensive poisoned samples. Table 26 shows that *a larger sampling frequency leads to a stronger defensive backdoor*, resulting in a higher ACC and a lower ASR. Note that for the malicious attacker, the poisoning ratio is expected to be low to ensure the stealthiness of the attack.
  2. **Data augmentation:** From Fig 7, we can find that PDB, without any sample augmentation ($\alpha = 0$), exhibits significant efficacy with ASR lower than 2%. As augmentation

Table 26: Results on PreAct-ResNet18 with poisoning ratio 5% and different sampling frequencies

Attack →	BadNet		Blended		Sig		SSBA		WaNet	
Frequency ↓	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
1	91.01	0.69	91.19	1.48	91.38	4.62	91.19	1.24	91.13	1.07
3	91.06	0.57	91.27	1.39	91.73	0.10	91.28	0.72	91.44	0.97
5	91.08	0.38	91.36	0.70	91.79	0.06	91.58	0.46	91.47	0.92
7	91.34	0.27	91.56	0.59	91.98	0.04	91.89	0.43	91.84	0.27
9	92.15	0.20	92.19	0.50	92.27	0.02	92.30	0.31	92.48	0.16

strength increases, the ASR decreases, indicating *a stronger augmentation can help further enhance PDB's effectiveness*. However, a tradeoff between augmentation intensity and model performance is also observed.

In summary, a visible trigger with *larger trigger size*, *higher sampling frequency*, and *data augmentation* contribute to meeting Principle 4.

C.8 Factors that influence the accuracy of PDB

Here, we would like to discuss the factors that influence the accuracy of PDB:

- **Model capacity and data complexity:**
  1. **Model capacity:** Since PDB introduces additional task, i.e., injecting defensive backdoor, increasing the model capacity helps to increase the accuracy of PDB, as evidenced in Table 27.

Table 27: Results on different models

Model	ResNet-18		ResNet-34		ResNet-50	
Metric	ACC	ASR	ACC	ASR	ACC	ASR
No Defense	92.54	76.27	93.08	82.48	93.76	87.26
PDB	91.81	0.29	92.63	0.28	93.67	0.18

  2. **Dataset complexity:** By comparing defense results with different datasets, we can find that by decreasing the dataset complexity, the accuracy of PDB increases significantly.
- **Strength of defensive backdoor:**
  1. **Strength of augmentation:** From Figure 7, we can find that there exists a tradeoff between ACC and ASR. Therefore, the accuracy of PDB can be boosted by reducing the strength of augmentation.
  2. **Sampling frequency:** From Table 26, we can find that by increasing the sampling frequency of defensive poisoned samples, the accuracy of PDB can be boosted.
  3. **Trigger size:** Table 23 shows that a proper choice of trigger size can also help to increase the accuracy. Therefore, if a validation set is accessible, a proper trigger size can be chosen to increase accuracy.

In summary, due to the "home field advantage" of PDB, there are several ways to maintain a high accuracy even in the case of a low malicious poisoning ratio, such as increasing model capacity, simplifying the dataset, reducing the strength of augmentation to defensive poisoned samples, increasing the sampling frequency and choosing a proper defensive trigger size.

C.9 Novelty and comparison with backdoorIndicator

Here, we highlight the distinctions between our approach and BackdoorIndicator [19].

First, we would like to clarify the following differences between our work and backdoorIndicator [19]:

- **Threat model:** [19] targets decentralized training (FL) setting, where multiple clients train models locally and contribute updates to a central server. Our work considers a centralized training setting where only a central server is used.
- **Task:** [19] focuses on detecting malicious clients, whereas our method aims to train a secure model on a poisoned dataset without clients.
- **Motivation:** [19] is built on the motivation that planting subsequent backdoors with the same target label enhances previously planted backdoors, therefore, providing a way to detect the poisoned clients, while our method is based on the motivation that planting a concurrent reversible backdoor can help to mitigate the malicious backdoor.
- **Methodology:** [19] utilizes OOD samples for backdoor client detection while our method constructs a proactive defensive poison dataset, following well-designed principles.

Second, we would like to discuss the challenges in direct utilizing backdoorIndicator in our setting:

- BackdoorIndicator is designed to detect malicious clients within a federated learning (FL) context. This makes it challenging to apply BackdoorIndicator directly to our centralized environment since the task of identifying backdoored clients does not naturally fit into this setting (only a central server).
- For comparison between BackdoorIndicator and our method, we need to emulate an FL scenario by assigning each image to a separate client (ensuring existence of benign client), thereby creating 50,000 local models from the CIFAR-10 dataset to defend a single attack with PreAct-ResNet18. This would require an impractical amount of computational resources, estimated at over 30,000 hours (1,250 days) of training time and 30TB of storage space using a server with a single RTX 3090 GPU.

C.10 Comparison to FT-SAM

To address the comparison with FT-SAM [55], we have adapted their method to our experimental setting. It's worth noting that in [55], the authors employ the Blended attack with a blending ratio of 0.1 (Blended-0.1), whereas we use a blending ratio of 0.2 (Blended-0.2). For consistency and completeness, we have now included experiments using both blending ratios, and the results are shown below:

Table 28: Results on PreAct-ResNet18 with FT-SAM

	Attack →	BadNet	BadNet	Blended-0.2	Blended-0.2	Blended-0.1	Blended-0.1	Sig	Sig	SSBA	SSBA
Poisoning ratio ↓	Defense ↓	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
5%	FT-SAM	92.66	1.22	92.87	31.54	92.76	2.87	92.82	1.80	92.83	3.27
5%	PDB	91.08	0.38	91.36	0.70	91.85	0.22	91.79	0.06	91.58	0.46

From Table 28, we can find FT-SAM can achieve a higher accuracy as it aims to fine-tune a backdoored model while PDB aims to train a model from scratch. Consistent with [55], Table 28 shows that FT-SAM can mitigate backdoor attacks for most cases, except for the Blended-0.2. We observe that FT-SAM struggles to defend against blended attacks with higher blending ratios, such as 0.2. Notably, PDB achieves a significantly lower ASR across all cases, with an average ASR below 0.5%.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction of the paper accurately reflect the paper's contributions and scope. They clearly state that the paper addresses the challenge of training a clean model on a potentially poisoned dataset by proposing a novel defense mechanism.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the limitations of the work performed by the authors. In the "Limitations and future work" section, we acknowledge that their implementation of the reversible backdoor defense has several key limitations.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper appears to provide detailed information on the experimental setup based on a popular Benchmark Project, including the datasets used, the model architectures, the backdoor attack strategies, and the defense mechanisms compared.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Provided in the supplemental material.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Provided in the main text and the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Provided in the main text and the appendix. Due to space limit, the bar are mainly visualized in the plots.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Provided in the main text and the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Provided in the main text.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: No such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Properly credited.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: Do not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.