Backdoor Mitigation by Distance-Driven Detoxification

Supplementary Material

6. Experiment details

For our experiments, we have adopted all baselines and configurations from the BackdoorBench [21]. Below, we provide a detailed overview of the experimental setup.

6.1. Attack details

In this section, we outline the configurations of each attack method utilized in our experiments and discussions:

- BadNets [4]: This pioneering work in backdoor learning involves inserting a small, fixed-pattern patch into images to manipulate certain pixels. Our implementation adheres to the default settings outlined in BackdoorBench.

- Blended backdoor attack (Blended) [3]: Utilizing an alpha-blending technique, this method integrates fixed patterns into images. In our experiments, we employ an alpha value of 0.2, which is the default in BackdoorBench. It is worth noting that this relatively high alpha value introduces noticeable changes to clean images, making the Blended Attack particularly challenging for defense mechanisms. The results for FT-SAM on the Blended Attack differ from those reported in [25], where an alpha value of 0.1 was used.

- Warping-based poisoned networks (WaNet) [12]: This attack employs a warping function during training to distort clean samples and create poisoned ones. We follow the default settings specified in BackdoorBench.

- Low-frequency attack (LF) [23]: By filtering out high-frequency artifacts from a Universal Adversarial Perturbation (UAP), this method applies a smooth trigger to images. Our implementation aligns with the default settings in BackdoorBench.

- Input-aware dynamic backdoor attack (Input-aware)[11]: This training-controllable attack learns a trigger generator that produces sample-specific triggers during the model's training phase. We adhere to the default settings in BackdoorBench.

- Sinusoidal signal backdoor attack (SIG) [1]: This clean-label attack uses a sinusoidal signal as a trigger to subtly alter images associated with the target label. Our experiments follow the default settings in BackdoorBench.

- Sample-specific backdoor attack (SSBA) [6]: Employing an auto-encoder, this method fuses a trigger into clean samples to generate poisoned data. We use the default settings from BackdoorBench.

To ensure a fair comparison, most of the attack checkpoints were obtained from BackdoorBench Model Zoo. For adaptive attacks, attacks with a poisoning ratio exceeding 10%, and attacks targeting ALL-to-ALL labels, we generated the necessary checkpoints using the default configurations provided in BackdoorBench, as these specific checkpoints are not available on the official website.

6.2. Defense details

In this section, we outline the configurations of each defense mechanism utilized in our experiments and discussions:

- FT (vanilla fine-tuning): This straightforward approach involves refining the compromised model using a reserved dataset. For consistency, we adhere to the vanilla configuration outlined in BackdoorBench.

- ANP [22]: Based on the work of Wu and Wang [22], this technique eliminates neurons that exhibit high sensitivity to changes in weights. Within the framework of BackdoorBench, ANP's pruning threshold can be determined either through grid search on the test data or by applying a predefined constant value. To ensure an fair comparison with other methods, we opt for a fixed threshold, setting it to 0.4—this adjustment yields superior results compared to the initially suggested threshold of 0.2. The remaining parameters align with BackdoorBench's default setup.

- FP [8]: As introduced by Liu et al. [8], this method selectively removes neurons based on their activation levels before proceeding with further refinement to maintain the model's accuracy on clean data. Our implementation follows the default guidelines provided by BackdoorBench.

- NC [19]: Wang et al. [19]'s method aims to identify potential triggers by optimizing them; upon detection of a backdoored model, it counteracts the backdoor through unlearning these optimized triggers. If the model is deemed clean, NC will not alter it. We use the default configurations in BackdoorBench.

- NAD [7]: This strategy, proposed by Li et al. [7], employs attention distillation to neutralize backdoors. We implement this method following the default settings specified in BackdoorBench.

- i-BAU [24]: Zeng et al. [24]'s approach leverages adversarial training with Universal Adversarial Perturbations (UAP) and hyper-gradients to combat backdoors. Our experimental setup uses the default conditions established in

BackdoorBench.

- FT-SAM [25]: Zhu et al. [25]'s method enhances the vanilla fine-tuning process by incorporating sharpness-aware minimization to bolster backdoor defenses. We apply this technique under the default settings prescribed by BackdoorBench.

- SAU [20]: Wei et al. [20]'s novel method utilizes Projected Gradient Descent (PGD) to create shared adversarial examples, which are subsequently unlearned to mitigate the impact of backdoors. Our experiments conform to the default settings detailed in BackdoorBench.

- D3 (Ours): The proposed method mitigates backdoor by steering the model parameters away from the initial backdoored model. Specifically, we focus on the weights of the linear layers, a common element across various model architectures. For experiments in Section 4, we configure the regularization parameter λ to 10 and the threshold ϵ to 0.1. The distance between the current and initial state of the selected weights is measured using the Frobenius norm of the difference, and we impose a constraint to maintain the Frobenius norm of the weights consistent with their initial values, ensuring $\|\boldsymbol{\theta}_s\|_F = \|\boldsymbol{\theta}_{init,s}\|_F$.
  Note: For the parameters λ and ϵ, a trade-off between accuracy and defense performance is observed. In simpler tasks, such as classification with a small dataset and a large model, using a smaller λ and a larger ϵ can enhance defense performance. Conversely, for more complex tasks, like classification with a complex dataset and a small model, a larger λ and a smaller ϵ is preferred to improve accuracy. For practical applications, we recommend setting ϵ within the range of $[0.1, 0.5]$, which corresponds to a prediction confidence interval of approximately 60% to 90%. For λ, we suggest values between 1 and 10 for simple tasks (e.g., GTSRB and CIFAR10) with robust models, and between 10 and 100 for complex tasks (e.g., ImageNet) with less powerful models. Further research into parameter selection could yield even better results, and this remains an area for future exploration.

7. Analysis and discussion

In this section, we provide more analysis and discussion of the proposed method, including the algorithm details, its effectiveness and more ablation studies.

7.1. D3 Algorithm

Optimization algorithm. Based on the aforementioned analysis, here we provide the details of the proposed method, summarized in the following algorithm 1.

Algorithm 1 Distance-Driven Detoxification

```
Input: Reserved dataset $\mathcal{D}_{cl}$, initial model weights $\boldsymbol{\theta}_{init}$, max iteration number $T$, selected weights $\boldsymbol{\theta}_s$, distance measure $d$, threshold $\epsilon$, multiplier $\lambda$, and the projection operator $\mathcal{P}$.
for $t = 0, ..., T-1$ do
  for Each mini-batch in $\mathcal{D}_{cl}$ do
    ▷ Loss computation
    Compute the weights distance
        $\mathcal{L}_{dis} = d(\boldsymbol{\theta}_s, \boldsymbol{\theta}_{init,s}).$
    Compute the classification loss $\mathcal{L}_{cls}$.
    Compute the overall batch loss
        $\mathcal{L}_{batch} = -\mathcal{L}_{dis} + \lambda \times max\left\{0, \mathcal{L}_{cls} - \epsilon\right\}.$
    ▷ Weights update
    Take gradient $\frac{\nabla \mathcal{L}_{batch}}{\nabla \boldsymbol{\theta}}$ and update $\boldsymbol{\theta}$ using SGD [2].
    Project the select weights $\boldsymbol{\theta}_s = \mathcal{P}(\boldsymbol{\theta}_s)$.
  end for
end for
return $\boldsymbol{\theta}$
```

Training cost comparison. Compared to vanilla fine-tuning, D3 incorporates the computation of the weight distance, which incurs only a minimal increase in computational cost. To underscore the efficiency of D3, we have benchmarked its execution time against other baselines. The experiments are performed with CIFAR-10, utilizing PreAct-ResNet18. All tests were executed on a server with an RTX 3090 GPU and an AMD EPYC 7543 32-Core Processor, ensuring a consistent number of training epochs across all methods for a fair evaluation. As illustrated in Figure 1, D3 exhibits faster execution times compared to most other defense strategies. This evidence reinforces the claim that D3 is not only effective but also highly efficient in combating backdoor attacks.

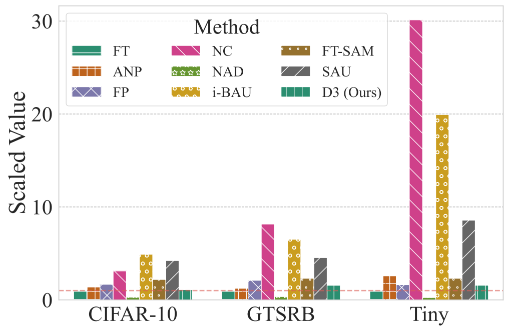

Figure 1. Comparative analysis of different methods across various Datasets. The values are normalized with respect to the training time of vanilla full-tuning.

7.2. More ablation study

Investigation of selected layers As outlined in Section 3, the proposed method encounters an overfitting issue, which compromises its generation capabilities and reduces its accuracy. To address this problem, it is essential to concentrate on a selective subset of model weights. Consequently, we explore the impact of various weight selections. Specifically, we perform experiments using the CIFAR-10 dataset with a PreAct-ResNet18 architecture under a 10% poisoning ratio.

Table 1. Selected layers in PreAct-ResNet18.

Index	Type	Shape
1	Conv2d	3x3x3x64
3	Conv2d	64x3x3x64
5	Conv2d	64x3x3x64
7	Conv2d	128x3x3x128
9	Conv2d	128x3x3x128
11	Conv2d	256x3x3x256
13	Conv2d	256x3x3x256
15	Conv2d	512x3x3x512
17	Conv2d	512x3x3x512
18	Linear	512x10

The PreAct-ResNet18 model consists of 18 layers, from which we have chosen 10 distinct layers, as detailed in Table 1, ranging from the initial to the final layer. We then assess the effectiveness of our approach by conducting separate experiments for each selected layer. The results are illustrated in Figure 2, demonstrating that our method can effectively counteract backdoor attacks regardless of the layer chosen. Notably, the deeper layers (the last three) contribute to maintaining a higher level of accuracy, likely because these layers have developed more refined feature representations. Conversely, the middle layers tend to struggle with achieving high ACC, possibly because they play a critical role in extracting clean features, which is vital for sustaining ACC.

In summary, selecting deeper layers is generally advisable. It is worth noting that our current analysis focuses solely on layer-wise weight selection. Future research could expand this investigation to include neuron-wise selection and combinations of multiple layers.

Investigation of distance metrics An essential aspect of the proposed method is the choice of distance metric. In this study, we primarily utilize a norm-based distance metric, defined as $d(\boldsymbol{\theta}_s, \boldsymbol{\theta}_{init,s}) = \|\boldsymbol{\theta}_s - \boldsymbol{\theta}_{init,s}\|$, where $\|\cdot\|$ represents a specific norm. This formulation enables us to design a corresponding projection operator to constrain the norm of the selected weights such that $\|\boldsymbol{\theta}_s\| = \|\boldsymbol{\theta}_{init,s}\|$.

To investigate the influence of different norm metrics, we conducted a series of experiments using the CIFAR-10 dataset with a PreAct-ResNet18 model architecture, under a 10% poisoning rate. Three distinct norm metrics were evaluated: the L_1 norm, the Frobenius norm (equivalent to the L_2 norm for two-dimensional matrices in linear weights), and the Nuclear norm. The performance of our method was assessed through separate experiments for each norm, with the results compiled in Table 2. The findings indicate that our approach effectively mitigates backdoor attacks across all tested norms. Note that as L_1 norm usually results in a significant larger value than other norms, it usually leads to a lower ACC than other norms. So, we recommend the Frobenius norm or the Nuclear norm.

Table 2. Results (%) on CIFAR-10 with PreAct-ResNet18 and poisoning ratio 10.0% with different norms.

Attack $\rightarrow$	BadNets [4]		Blended [3]		WaNet [12]	
Norm $\downarrow$	ACC	ASR	ACC	ASR	ACC	ASR
L_1	89.86	0.46	91.81	4.44	92.73	0.11
Frobenius/L_2	90.77	0.74	92.29	0.22	93.31	0.04
Nuclear	90.46	0.67	92.04	0.86	93.32	0.03
Attack $\rightarrow$	SSBA [6]		LF [23]		Input-aware [11]	
Norm $\downarrow$	ACC	ASR	ACC	ASR	ACC	ASR
L_1	91.18	1.59	91.65	2.10	92.43	0.11
Frobenius/L_2	91.85	0.81	92.37	1.31	92.96	0.06
Nuclear	91.82	1.17	92.35	1.16	92.89	0.07

7.3. Connection and distinction to fine-tuning based methods

In this section, we present a detailed comparison between our method and some existing fine-tuning based method, including those that are currently unpublished or in preprint form. This comparative analysis aims to highlight both the similarities and the unique contributions of our work.

Connection and distinction to FT-SAM [25] : The work by Zhu et al. [25] posits that vanilla fine-tuning (FT) lacks the capacity to escape from backdoor solutions, leading to insufficient mitigation of backdoor effects (Section 3.2 in [25]). This hypothesis aligns with our findings. However, their validation is limited to demonstrating that the weight differences are minimal and that the backdoor remains active (high ASR).

In contrast, our approach offers a deeper and more rigorous examination. We construct a continuous path between the backdoor model and the fine-tuned model, analyzing how the backdoor loss evolves along this path. This analysis reveals that increasing the distance between the models can effectively mitigate the backdoor effect. Furthermore, we introduce a novel optimization problem designed to explicitly move the model away from the backdoor solution, providing a more robust and systematic approach to backdoor mitigation.

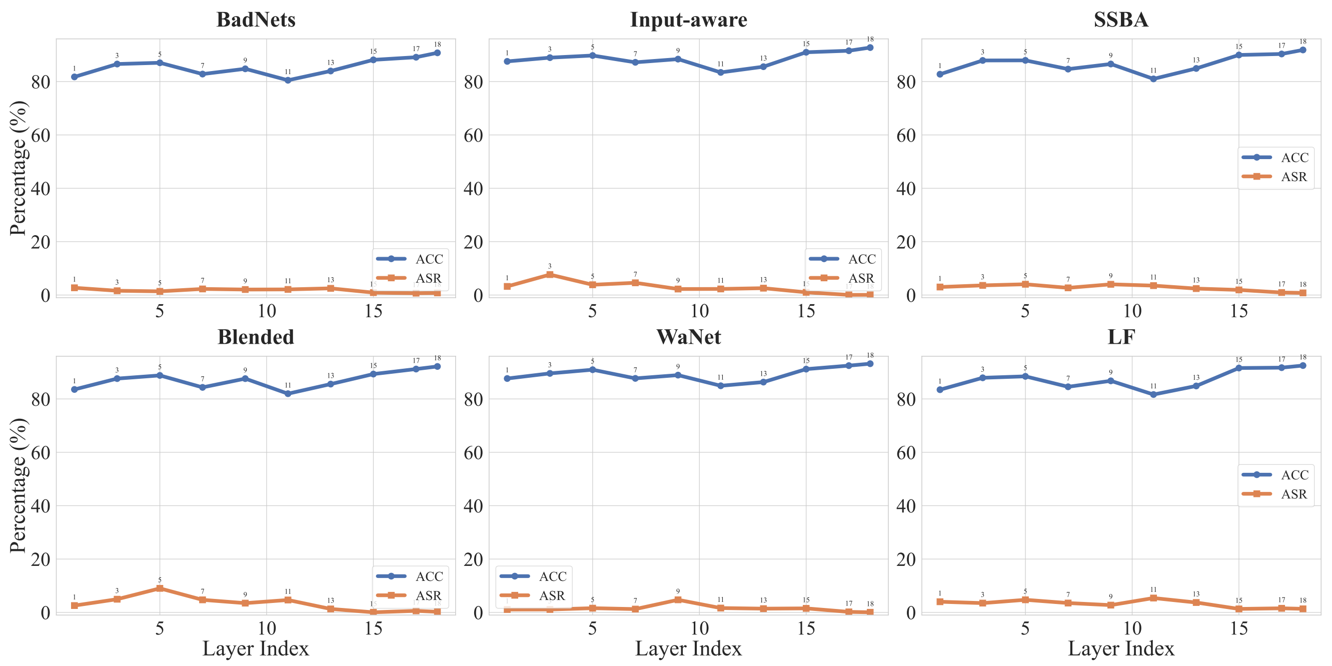

Figure 2. Visualization of experiments with different choices of layers in PreAct-ResNet18, with CIFAR-10 dataset and poisoning ratio 10%.

Connection and distinction to FST [10] : Min et al. [10] introduce a feature-shifting technique (FST) for backdoor mitigation. They analyze the features used in the final classification layer and propose shifting these features to mitigate backdoor effects. By approximating the features using the weights of a linear classifier, they formulate a regularized problem with the goal to decrease the feature similarity. Despite some similarities in the formulation, our method diverges from FST in several key aspects: **1) Perspective and Motivation:** Our method is grounded in the loss landscape perspective and addresses the inherent limitations of vanilla fine-tuning. In contrast, FST focuses primarily on feature manipulation. **2) Generality and Flexibility:** Our approach offers a more general framework, whereas FST can be seen as a specific instance of our method. For example, setting threshold ϵ to zero, selecting linear weight and using the negative weight product as a distance metric in our framework essentially reduces it to FST. However, as aforementioned analysis, our method can be applied with various selection of weights, distance metrics and thresholds ϵ. By highlighting these distinctions, we underscore the broader applicability and enhanced effectiveness of our method in mitigating backdoor attacks.

Connection and distinction to Super-FT [17] : Sha et al. [17] propose a method for backdoor mitigation by dynamically adjusting the learning rate during fine-tuning. Their approach involves using a high learning rate initially to "forget" the backdoor and then a low learning rate to maintain model utility. While this method is conceptually distinct from ours, it can be interpreted within the framework of loss landscape analysis. A high learning rate facilitates escaping local minima, thus helping the model to move away from the backdoor region, indirectly reducing the backdoor loss.

However, our method takes a more direct and explicit approach by constructing an optimization problem that explicitly increases the distance from the backdoor solution. This ensures a more controlled and predictable mitigation of backdoor effects. It is worth noting that since [17] is still a working paper and lacks an official implementation, we have not conducted a direct comparison with their method.

Connection and distinction to NFT [5] : Karim et al. [5] proposed Neural mask Fine-Tuning (NFT), a method that mitigates backdoor attacks by fine-tuning neural masks instead of directly modifying model weights. NFT leverages data augmentation techniques like MixUp to relax the trigger synthesis process, thereby reorganizing neuron activations to counteract backdoor triggers. While NFT does not explicitly adjust weights through conventional fine-tuning, it indirectly modifies the effective weights via mask optimization. Specifically, by optimizing masks on augmented data, NFT implicitly steers the masked weights away from the backdoored initial configuration. This contrasts with our approach, D3, which directly manipulates weights within a constrained subspace to explicitly counteract backdoor-induced shifts.

Summary of our unique contribution. Here, we would like to emphasize our contributions from three key perspectives: **1) Identify the Gap Between Ideal Backdoor Purification and Vanilla Fine-Tuning.** We identify a fundamental discrepancy between the ideal objective of backdoor purification and the vanilla fine-tuning loss, supported by our empirical observations in Figure 2. **2) Explicit Solution to Escape the Backdoor Region.** While some prior works, such as FT-SAM, FST, Super-FT and NFT can also move the model away from initial weights, they rely on implicit mechanisms such as SAM, Feature Shift, MixUp, Dynamic Learning Rate or Neural Masking, to indirectly achieve this objective. In contrast, by addressing the challenges in Section 3.3, we offer explicit and principled method for escaping the backdoor region. **3) Superior Performance in Challenging Scenarios.** The explicit nature of our method enables our method to provide more robust defense performance, particularly against adaptive attacks designed to counter fine-tuning-based defenses.

7.4. Grad-CAM Visualization

To gain deeper insights into the effectiveness of our proposed method, we utilized Grad-CAM [16] to visualize the attention mechanisms of both the backdoored and detoxified models. For optimal visualization, we specifically focused on the BadNets attack scenario, employing the CIFAR-10 dataset and a PreAct-ResNet18 architecture, with a 10% poisoning rate. This is because the BadNets attack uses a human-friendly trigger, making it ideal for visual analysis.

The results, illustrated in Figure 3, reveal that the backdoored model exhibits a pronounced focus on the trigger area when processing poisoned samples. In contrast, the detoxified model redirects its attention towards the semantic features of the images, effectively neutralizing the backdoor influence. Furthermore, the detoxified model demonstrates nearly identical attention patterns to the original, unpoisoned model when handling clean samples, suggesting that it maintains a high level of accuracy for these inputs.

8. Additional experiments

8.1. Experiments on more datasets and models

In this section, we extend the initial findings presented in Section 4 through a rigorous and comprehensive assessment of our proposed method. This evaluation spans multiple datasets and model architectures, thereby substantiating the versatility and robustness of our approach. We specifically examine its effectiveness with the PreAct-ResNet18, VGG19-BN, and ViT-B-16 models, employing the CIFAR-10, GTSRB, and Tiny ImageNet datasets. Note that for the more complex tasks involving Tiny ImageNet and ViT-B-16, where the challenge lies in balancing model complexity and task difficulty, we set different λ and ϵ, while reserving 10% of the training data. Recognizing the computational demands of the ViT-B-16 architecture, we further adjusted the learning rate to 0.001, ensuring that the optimization process remains efficient and effective, thus enabling the identification of more optimal solutions.

Here, we also provides the results of some ViT-compatible baselines and summarized the results in Table 4, showing the superior performance of D3.

Table 3. Results on Tiny ImageNet with ViT-B-16.

Attack →	BadNets [4]		Blended [3]		WaNet [12]	
Defense ↓	ACC	ASR	ACC	ASR	ACC	ASR
No Defense	73.96	99.79	75.28	99.93	62.68	99.61
D3 (**Ours**)	71.64	0.29	73.13	4.48	60.92	0.98
Attack →	**Input-aware [11]**		**LF [23]**		**SSBA [6]**	
Defense ↓	ACC	ASR	ACC	ASR	ACC	ASR
No Defense	63.86	99.9	59.72	3.81	76.37	99.27
D3 (**Ours**)	61.11	0.61	57.03	0.21	73.68	0.01

Table 4. Defense on Tiny-ImageNet+ViT.

Attack	Defense	FT	FT-SAM	SAU	D3
BadNet	ACC/ASR	72.54/5.89	73.31/0.89	68.85/0.00	71.64/0.29
Blended	ACC/ASR	73.89/18.56	74.85/12.35	67.56/6.52	73.13/4.48

The results of these meticulous evaluations are encapsulated in Tables 5, 6 and 3. These tables reveal a nuanced trade-off: while our method may introduce a slight reduction in accuracy on clean samples—particularly pronounced with larger, more complex models—this is a necessary concession for achieving enhanced robustness against adversarial attacks. This observation underscores the ongoing challenge of securing large-scale models.

Moreover, our results unequivocally demonstrate that the proposed method not only mitigates backdoor attacks effectively across a wide range of datasets and models but also achieves performance that is on par with, or surpasses, current state-of-the-art defense mechanisms. Consequently, these findings position our method as a highly promising and viable solution for enhancing the security and reliability of deep learning systems.

8.2. Baselines with various poisoning ratios.

We present part of results for the most competing baselines with different poisoning ratios. The results are summarized in Table 7, showing that the proposed method can consistently outperform baselines for various poison ratios.

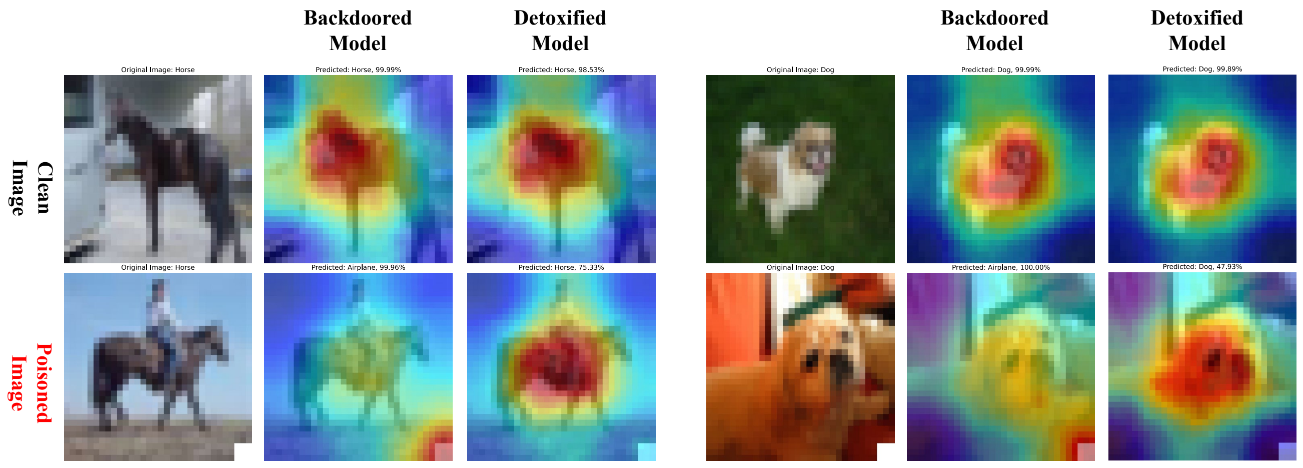

Figure 3. Visualization of Grad-CAM for Backdoored and Detoxified Models. The first row displays clean images, while the second row presents poisoned images. Columns two and five illustrate the results from the backdoored model, whereas columns three and six showcase the output of the detoxified model.

Table 5. Results on GTSRB with PreAct-ResNet18 and poisoning ratio 10.0%.

Defense →	No Defense			FT			ANP [22]			FP [8]			NC [19]		
Attack ↓	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [4]	97.24	59.25	N/A	**98.73**	5.09	77.08	96.89	0.06	79.42	98.21	0.09	79.58	97.48	0.01	79.62
Blended [3]	98.58	99.99	N/A	98.57	100.0	50.0	**98.75**	99.82	50.09	98.38	100.0	49.9	97.76	8.03	95.57
WaNet [12]	97.74	94.25	N/A	98.54	0.29	96.98	98.00	0.00	97.12	97.62	88.07	53.03	98.25	0.00	97.12
LF [23]	97.93	99.57	N/A	<u>98.01</u>	79.98	59.8	97.80	81.38	59.03	97.59	99.7	49.83	97.97	1.34	<u>99.11</u>
Input-aware [11]	97.26	92.74	N/A	98.40	29.81	81.46	<u>99.14</u>	<u>0.00</u>	<u>96.37</u>	98.08	2.32	95.21	98.55	0.01	96.36
SSBA [6]	97.98	99.56	N/A	97.92	99.1	50.2	97.86	98.73	50.36	97.75	99.46	49.94	97.72	<u>0.29</u>	**99.50**
Average	97.79	90.89	N/A	98.36	52.38	69.25	98.07	46.66	72.06	97.94	64.94	62.92	97.95	1.61	94.55
Defense →	**NAD [7]**			**i-BAU [24]**			**FT-SAM [25]**			**SAU [20]**			**D3 (Ours)**		
Attack ↓	**ACC**	**ASR**	**DER**	**ACC**	**ASR**	**DER**	**ACC**	**ASR**	**DER**	**ACC**	**ASR**	**DER**	**ACC**	**ASR**	**DER**
BadNets [4]	<u>98.69</u>	0.63	79.31	96.47	0.02	79.23	98.65	0.56	79.34	97.92	<u>0.00</u>	<u>79.63</u>	97.36	**0.00**	**79.63**
Blended [3]	<u>98.61</u>	100.0	50.0	92.35	86.35	53.71	98.28	68.74	65.48	98.15	**0.37**	**99.59**	97.24	<u>0.54</u>	<u>99.06</u>
WaNet [12]	<u>98.61</u>	0.56	96.84	96.72	0.0	96.62	**98.67**	0.01	<u>97.12</u>	98.38	<u>0.00</u>	**97.12**	97.62	**0.00**	97.07
LF [23]	**98.14**	51.83	73.87	95.78	16.15	90.64	97.67	<u>0.64</u>	**99.34**	94.35	0.65	97.67	96.41	**0.00**	99.03
Input-aware [11]	98.27	40.65	76.04	97.09	0.52	96.03	**99.55**	0.13	96.30	98.75	**0.00**	**96.37**	97.93	0.01	96.36
SSBA [6]	<u>97.95</u>	99.39	50.07	96.14	1.88	97.92	**98.38**	40.12	79.72	97.27	0.36	99.25	97.25	**0.06**	<u>99.39</u>
Average	<u>98.38</u>	48.84	71.02	95.76	17.49	85.69	**98.54**	18.37	86.22	97.47	<u>0.23</u>	<u>94.94</u>	97.30	**0.10**	**95.09**

Table 7. Baselines with different poison ratios on BadNets.

Defense	ANP	FT-SAM	SAU	Ours
Poison Ratio	ACC/ASR	ACC/ASR	ACC/ASR	ACC/ASR
1%	92.93/9.61	92.46/1.57	90.85/1.28	92.18/0.68
30%	88.39/27.48	89.77/1.81	88.25/1.57	88.97/0.99
50%	84.88/24.33	87.84/3.17	84.85/3.33	86.90/1.51

8.3. Extend to other tasks/modalities

As suggested by reviewers, D3 is not restricted to vision/classification tasks. While our focus is vision, we conducted a preliminary test on HiddenKiller [13], a backdoor attack for NLP. D3 effectively suppressed ASR on SST-2 (88.93% → 1.35%) and AGNews (96.68% → 1.23%), with $\leq 1\%$ ACC drop, suggesting promising cross-modal potential to explore in future work.

8.4. Experiments on more attacks

Previous experiments have conclusively demonstrated that our proposed D3 method not only matches but consistently outperforms state-of-the-art post-training defense techniques in terms of robustness. Given this established superiority, the focus here shifts to further validating the broad applicability of D3 across additional attack vectors. Specifically, we extend our evaluation to include two recently proposed attacks, namely Adap-Blended [15] and Refool [9], imple-

Table 6. Results on CIFAR-10 with VGG19-BN and poisoning ratio 10.0%.

Defense →	No Defense			FT			ANP [22]			FP [8]			NC [19]		
Attack ↓	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [4]	90.42	94.43	N/A	88.19	27.59	82.31	**90.50**	88.43	53.00	88.71	10.61	91.06	89.21	11.31	90.96
Blended [3]	91.91	99.5	N/A	90.08	86.82	55.42	**91.66**	97.43	50.91	90.28	89.53	54.17	90.07	83.33	57.16
WaNet [12]	84.58	96.49	N/A	91.35	5.72	95.39	89.80	**0.90**	**97.80**	90.84	1.94	97.28	91.20	6.88	94.81
LF [23]	83.28	13.83	N/A	87.67	1.82	56.01	89.22	1.41	56.21	88.64	1.34	56.24	86.64	1.36	56.24
Input-aware [11]	88.66	94.58	N/A	91.56	13.08	90.75	89.83	**2.79**	**95.89**	91.38	19.81	87.38	89.70	97.02	50.0
SIG [1]	83.48	98.87	N/A	88.01	4.28	97.29	81.93	**0.97**	**98.18**	88.22	11.07	93.9	83.48	98.87	50.0
SSBA [6]	90.85	95.11	N/A	89.26	70.22	61.65	**90.90**	93.18	50.97	89.45	63.14	65.28	90.85	95.11	50.0
Average	87.6	84.69	N/A	89.45	29.93	76.97	89.12	40.73	71.85	89.65	28.21	77.9	88.74	56.27	64.17
Defense →	NAD [7]			i-BAU [24]			FT-SAM [25]			SAU [20]			D3 (**Ours**)		
Attack ↓	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [4]	86.48	5.47	92.51	86.01	2.28	93.87	90.34	2.21	**96.07**	88.2	**0.49**	95.86	87.33	1.16	95.09
Blended [3]	88.6	83.86	56.17	87.58	69.9	62.64	91.35	20.28	89.33	87.71	**2.34**	**96.48**	89.72	6.25	95.53
WaNet [12]	90.73	10.33	93.08	89.76	1.61	97.44	**91.83**	1.18	97.66	89.35	5.30	95.60	90.91	1.45	97.52
LF [23]	83.72	**1.14**	**56.34**	87.68	1.47	56.18	**89.85**	1.21	56.31	82.08	1.23	55.70	81.3	1.82	55.02
Input-aware [11]	91.0	14.11	90.23	88.29	69.56	62.33	**92.05**	5.10	94.74	88.61	2.97	95.78	90.4	3.10	95.74
SIG [1]	86.07	7.39	95.74	83.41	5.37	96.71	**89.70**	3.82	97.52	85.57	3.58	97.64	87.88	2.96	97.96
SSBA [6]	88.33	56.64	67.97	87.56	22.26	84.78	90.27	34.14	80.19	86.49	6.66	92.05	88.5	**4.10**	**94.33**
Average	87.85	25.56	78.86	87.18	24.63	79.14	**90.77**	9.71	87.40	86.86	3.22	89.87	88.01	**2.98**	**90.17**

mented under their respective default configurations from the original papers. As shown in Table 8, D3 achieves consistent and robust defense performance against these advanced attacks, with all metrics remaining within the expected effective range. These results reaffirm the method's capability to maintain strong adversarial robustness without compromising classification accuracy.

Table 8. Experiments on more attacks.

Attack	Adap-Blended		Refool	
Defense	No Defense	Ours	No Defense	Ours
Poison Ratio	ACC/ASR	ACC/ASR	ACC/ASR	ACC/ASR
1%	92.32/67.11	91.26/1.24	92.80/46.17	91.39/1.96
5%	92.01/92.38	91.24/0.48	91.75/84.76	90.83/1.32
10%	91.40/95.18	90.09/0.83	91.08/93.55	90.89/1.67

In addition, we also consider tow recent attack SRA [14] and TaCT [18], and the results are shown below in Table 9.

Table 9. Experiments on recent attacks and suggested attacks.

Defense	Metric	SRA [14]	TaCT [18]
No Defense	ACC/ASR	89.28/92.58	91.54/94.35
D3	ACC/ASR	89.02/1.23	90.95/0.93

8.5. Experiments on more defenses

Here, we include additional evaluations on more defense. Results in Table 10 shows that D3 can achieve SOTA defense methods.

Table 10. Experiments on defenses, report ACC/ASR.

	Detection		In-training		Post-training		
Attack	Spectral	AC	ABL	DBD	OTBR	RNP	D3
BadNets	90.62/74.22	88.89/68.14	84.39/0.00	89.65/1.27	89.78/0.68	87.56/1.65	90.77/0.74
Blended	91.56/88.52	89.62/24.36	87.73/0.77	89.91/89.98	90.65/9.78	86.89/6.79	92.29/0.22
LF	91.72/89.55	90.85/36.58	85.79/2.21	82.98/49.32	90.48/63.89	87.27/9.33	92.37/1.31

8.6. Experiments on untargeted attack

Here, we conduct evaluations on untargeted BadNet and Blended attacks with 5%/10% poisoning. ASR is measured as the misclassification rate among originally correctly predicted samples. D3 reduces ASR to below 2% in all cases (Table 11), validating its generality.

Table 11. Experiments on Untargeted Attacks.

Attack →		BadNet		Blended	
Pratio ↓	Defense →	No defense	D3	No defense	D3
5%	ACC/ASR	91.97/95.32	90.55/0.56	92.65/99.25	91.98/1.15
10%	ACC/ASR	90.23/97.61	89.65/1.06	92.44/99.27	91.85/1.86

8.7. Scalability across models/image sizes.

Here, we evaluate D3 on ImageNette (a subset of ImageNet with large size 320×320) and a small model VGG-11. D3 achieves low ASR ($\leq 1.5\%$) with minimal clean accuracy loss (Table 13,14), showing robustness across model scales and input sizes.

Table 12. Results for ALL-to-ALL attacks on CIFAR-10 with PreAct-ResNet18 and poisoning ratio 10.0%.

Defense →	No Defense			FT			ANP [22]			FP [8]			NC [19]		
Attack ↓	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [4]	91.89	74.42	N/A	91.56	1.17	86.46	**92.27**	2.15	86.14	91.98	1.31	86.56	89.84	1.19	85.59
Blended [3]	93.67	86.69	N/A	**93.25**	81.46	52.4	91.82	2.59	91.12	92.38	13.25	86.07	91.35	31.65	76.36
Input-aware [11]	91.23	85.66	N/A	93.18	75.26	55.2	90.68	1.17	91.97	92.77	1.48	92.09	92.82	14.26	85.7
WaNet [12]	89.91	78.58	N/A	93.15	0.88	88.85	86.30	**0.63**	87.17	92.96	0.94	88.82	89.91	78.58	50.0
Average	91.68	81.34	N/A	92.78	39.69	70.73	90.27	1.64	89.1	92.52	4.24	88.38	90.98	31.42	74.41

Defense →	NAD [7]			i-BAU [24]			FT-SAM [25]			SAU [20]			D3 (**Ours**)		
Attack ↓	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER	ACC	ASR	DER
BadNets [4]	90.73	1.61	85.82	89.39	1.29	85.32	91.87	1.03	**86.68**	90.95	1.30	86.09	91.43	**0.68**	86.64
Blended [3]	92.46	65.95	59.76	90.46	8.83	87.32	92.66	**2.09**	**91.80**	91.32	3.19	90.58	92.56	2.30	91.64
Input-aware [11]	92.68	82.22	51.72	90.98	22.47	81.47	**93.38**	9.05	88.30	91.51	0.90	92.38	92.66	**0.83**	**92.42**
WaNet [12]	93.01	0.98	88.8	91.71	1.63	88.48	**93.51**	0.80	88.89	91.25	1.02	88.78	92.87	0.74	**88.92**
Average	92.22	37.69	71.53	90.64	8.56	85.65	**92.86**	3.24	88.92	91.26	1.60	89.46	92.38	**1.14**	**89.90**

8.8. Experiments on Multi-targets attacks

In previous experiments, we assumes that attacker only use one target label. However, in real applications, attackers may have multiple target labels. To evaluate our method against such cases, we conduct experiments with ALL-to-ALLattacks on CIFAR-10 with PreAct-ResNet18 and poisoning ratio 10%. Specifically, the target labels for the sample with original labels y are set to $y_t = (y+1) \mod K$ where $\mod$ is short for "modulus". The experiment results are summarized in Table 12. From Table 12, we can find that D3 achieves the top-2 defending performance in all attacks and the lowest average ASR. At the same time, D3 also achieves the best average DER, which further demonstrates its effectiveness in defending against backdoor attacks with multiple targets.

Experiments on multi-trigger and multi-target. Here, we also conducted experiments under combined attacks (e.g., BadNet + Blended), and D3 still reduces ASR to $\leq 2\%$ (Table 15).

Table 13. Experiments on ImageNette+PreAct-ResNet18.

Defense	No defense	FT-SAM	SAU	D3
Attack	ACC/ASR	ACC/ASR	ACC/ASR	ACC/ASR
BadNets	87.57/99.41	86.85/0.51	84.25/1.85	86.82/0.48
Blended	87.44/98.95	86.24/10.28	83.75/6.89	86.68/1.57

Table 14. Experiments on CIFAR10+VGG11.

Defense	No defense	D3	No defense	D3
Pratio	ACC/ASR	ACC/ASR	ACC/ASR	ACC/ASR
5%	88.35/89.97	87.89/0.65	89.52/97.52	88.98/1.32
10%	88.17/93.55	87.42/0.76	89.29/99.23	88.15/1.12

Table 15. Experiments on multi-targets and multi-triggers attack.

Attack		No defense	D3
Attack-1	Attack-2	ACC/ASR-1/ASR-2	ACC/ASR-1/ASR-2
Blended	BadNet	90.22/99.27/95.03	89.78/1.68/0.62
LF	BadNet	90.16/98.08/95.15	89.54/1.30/0.52
WaNet	BadNet	89.44/90.21/95.40	88.98/0.42/0.57

References

[1] 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. 1, 7

[2] Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Conference on Neural Information Processing Systems*, 2007. 2

[3] 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. 1, 3, 5, 6, 7, 8

[4] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, pages 47230–47244, 2019. 1, 3, 5, 6, 7, 8

[5] Nazmul Karim, Abdullah Al Arafat, Umar Khalid, Zhishan Guo, and Nazanin Rahnavard. Augmented neural fine-tuning for efficient backdoor purification. In *European Conference on Computer Vision*, pages 401–418. Springer, 2024. 4

[6] 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. 1, 3, 5, 6, 7

[7] 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. 1, 6, 7, 8

[8] 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. 1, 6, 7, 8

[9] Yunfei Liu, Xingjun Ma, James Bailey, and Feng Lu. Reflection backdoor: A natural backdoor attack on deep neural networks. In *Computer vision–ECCV 2020: 16th European conference, Glasgow, UK, August 23–28, 2020, proceedings, part X 16*, pages 182–199. Springer, 2020. 6

[10] 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. 4

[11] Tuan Anh Nguyen and Anh Tran. Input-aware dynamic backdoor attack. In *Conference on Neural Information Processing Systems*, 2020. 1, 3, 5, 6, 7, 8

[12] Tuan Anh Nguyen and Anh Tuan Tran. Wanet - imperceptible warping-based backdoor attack. In *International Conference on Learning Representations*, 2021. 1, 3, 5, 6, 7, 8

[13] Fanchao Qi, Mukai Li, Yangyi Chen, Zhengyan Zhang, Zhiyuan Liu, Yasheng Wang, and Maosong Sun. Hidden killer: Invisible textual backdoor attacks with syntactic trigger. *arXiv preprint arXiv:2105.12400*, 2021. 6

[14] Xiangyu Qi, Tinghao Xie, Ruizhe Pan, Jifeng Zhu, Yong Yang, and Kai Bu. Towards practical deployment-stage backdoor attack on deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13347–13357, 2022. 7

[15] 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. 6

[16] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *International Conference on Computer Vision*, 2017. 5

[17] Zeyang Sha, Xinlei He, Pascal Berrang, Mathias Humbert, and Yang Zhang. Fine-tuning is all you need to mitigate backdoor attacks. *arXiv preprint arXiv:2212.09067*, 2022. 4

[18] Di Tang, XiaoFeng Wang, Haixu Tang, and Kehuan Zhang. Demon in the variant: Statistical analysis of {DNNs} for robust backdoor contamination detection. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 1541–1558, 2021. 7

[19] 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. 1, 6, 7, 8

[20] 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. 2, 6, 7, 8

[21] 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

[22] Dongxian Wu and Yisen Wang. Adversarial neuron pruning purifies backdoored deep models. In *Conference on Neural Information Processing Systems*, 2021. 1, 6, 7, 8

[23] 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. 1, 3, 5, 6, 7

[24] 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. 1, 6, 7, 8

[25] 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. 1, 2, 3, 6, 7, 8