# A Appendix

## A.1 Details of Baseline Backdoor Attacks and Defenses

In this section, we provide the details of leveraged baseline backdoor attacks and defense methods.

**Backdoor Attacks.** We evaluate on 14 prominent backdoor attacks, following the original trigger patterns and poisoning strategies, with a fixed poisoning rate of 10%. Figure 7 provides visualizations of different backdoor triggers, where we displays images stamped with triggers in the first rows and the differences between poisoned images and their source versions in the second rows.

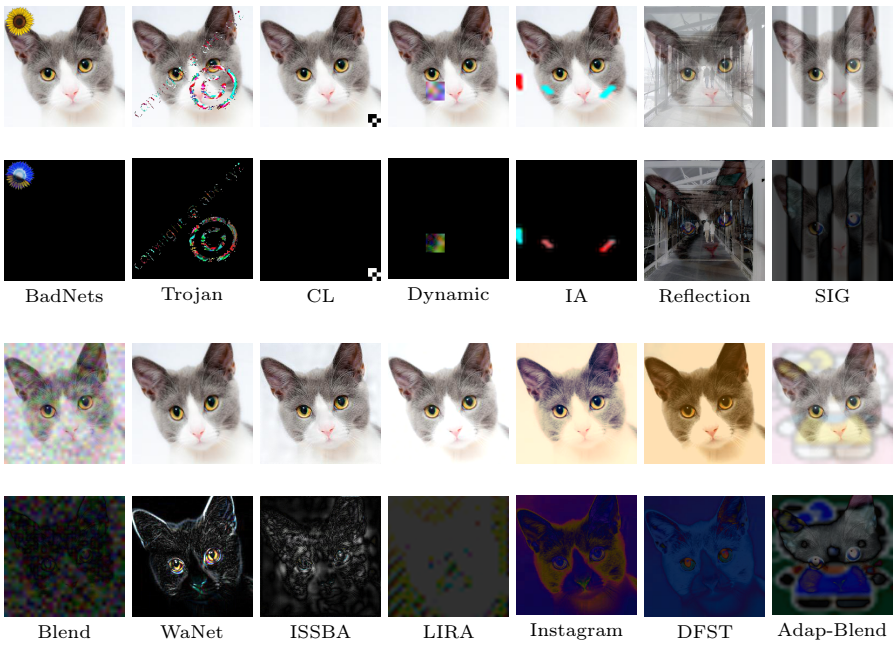


**Fig. 7:** Backdoor trigger examples

- *BadNets* [20] introduces backdoor attacks by incorporating a small percentage of poisoned samples into the training data using standard data-poisoning. During inference, images stamped with the trigger are misclassified to the specified target label.
- *Trojan* [40] injects the backdoor by manipulating selective internal neurons, ensuring the trigger activates these neurons with high values, causing targeted misclassification.
- *CL* [65] proposes a clean-label attack that poisons only samples of the target class during training. It introduces adversarial perturbations to target inputs, misclassifying them to other classes before applying data-poisoning.

- *Dynamic backdoor* [53] leverages a trigger generator to inject various triggers randomly into different inputs.
- *IA* [48] utilizes two trigger generation networks to create trigger masks and patterns based on the inputs, establishing a unique one-to-one mapping between the input and its trigger.
- *Reflection* [41] blends inputs with another image to produce a reflection effect.
- *SIG* [3] perturbs input images with strip effects. It also operates as a clean-label attack.
- *Blend* [6] applies small random perturbations and blends them with the input to create the trigger.
- *WaNet* [47] uses a complex wrapping function to induce a line-bending effect on inputs as the trigger.
- *ISSBA* [37] injects an invisible trigger using an image-to-image transforming network.
- *LIRA* [15] employs a network to inject sample-specific perturbations into inputs as the trigger.
- *Instagram* [39] uses Instagram filters to introduce the trigger.
- *DFST* [7] utilizes a CycleGAN to apply a sunshine effect on inputs as the trigger. It also incorporates a detoxification process to eliminate low-level trigger features, directing the model's focus to high-level features.
- *Adap-Blend* [51] leverages asymmetric and low-confidence training to reduce the latent distance between clean and poisoned samples, enhancing the stealthiness and robustness of the attack against existing defenses.

**Backdoor Mitigation Baselines.** We compare our technique UNIT with 7 state-of-the-art defenses given the same number of clean training data. We follow the original implementation to conduct experiments and tuning parameters to acquire best performance.

- *Standard Fine-Tuning (FT)* retrains the model using the given clean data. We perform fine-tuning for 20 epochs with an initial learning rate of $10^{-2}$ and reduce the learning rate by a factor of 10 every 4 epochs. Data augmentation techniques, including random cropping, rotation, and horizontal flipping, are applied to enhance model generalization.
- *Fine-Pruning (FP)* [38] first prunes dormant neurons with low activation values on clean inputs (potential backdoor neurons) and then applies standard fine-tuning to the model.
- *NAD* [35] distills the knowledge from the teacher model to the student model. The teacher model is derived from the backdoored model after standard fine-tuning. The backdoor effect is removed through the distillation to the student model, only based on clean representations.
- *ANP* [69] observes that backdoor neurons are sensitive to small perturbations in weight values. It prunes the most sensitive neurons based on this observation.
- *NC* [66] reverse-engineers backdoor triggers and applies adversarial training to neutralize the effectiveness of the generated triggers.

– *I-BAU* [74] introduces a min-max formulation to eliminate backdoors and leverages implicit hypergradients to optimize the balance between removal efficiency and effectiveness.
– *SEAM* [83] leverages the catastrophe forgetting assumption [31] by first retraining the model on clean samples with randomly assigned labels to forget both clean and backdoor behaviors. It then fine-tune the model on samples with the correct labels to restore the clean performance.

**Table 3:** Evaluation results on the latest backdoor attacks and defenses

| Attacks | No Defense | | CLP | | FST | | RNP | | FT-SAM | | Super-FT | | UNIT | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Acc. | ASR | Acc. | ASR | Acc. | ASR | Acc. | ASR | Acc. | ASR | Acc. | ASR | Acc. | ASR |
| BadNets | 94.82 | 100.0 | 92.37 | 1.18 | 92.50 | **0.00** | 92.18 | 0.64 | 91.75 | 0.86 | 91.58 | 0.90 | **92.48** | 0.78 |
| Instagram | 94.62 | 99.59 | 91.69 | 48.08 | **91.93** | 8.55 | 90.05 | 14.50 | 90.96 | 9.91 | 90.55 | 6.83 | 91.43 | **4.98** |
| Reflection | 93.29 | 99.59 | **92.06** | 36.16 | 91.46 | 7.58 | 89.59 | 84.30 | 90.58 | 53.96 | 91.02 | 48.07 | 91.44 | **6.63** |
| NARCISSUS | 92.67 | 95.48 | 87.98 | 84.64 | 86.86 | 41.49 | 89.17 | 79.96 | 89.12 | 43.61 | **89.31** | 59.48 | 88.74 | **37.97** |
| COMBAT | 94.00 | 88.74 | 90.07 | 50.93 | 91.66 | 73.79 | 90.99 | 72.57 | 89.22 | 62.42 | **90.61** | 59.31 | 90.13 | **48.58** |

### A.2 Additional Evaluation on the Latest Backdoor Attacks and Defense Mechanisms

We compare UNIT with five additional state-of-the-art baselines, CLP [80], FST [45], RNP [36], FT-SAM [82] and Super-FT [54]. CLP identifies and prunes backdoor neurons by examining the channel Lipschitzness to reduce the backdoor effect. It is based on the observation that backdoor neurons tend to have high channel Lipschitz values. FST actively deviates the weights of the classification layer (e.g., the last fully connected layer in ResNet-18) from the originally compromised weights. It then fine-tunes the feature extraction weights to calibrates the shifted classification weights, aiming to destroy the backdoor correlation. RNP identifies malicious neurons by unlearning using clean samples with randomly shuffled labels and then recovering using the ground-truth labels. Malicious neurons stand out as they are sensitive to this change and RNP prunes them accordingly. FT-SAM leverages sharpness-aware minimization to achieve better unlearning while Super-FT design a special learning rate scheduler to enhance the backdoor unlearning. We consider three typical backdoor attacks, BadNets [20], Instagram filter [39] and Reflection [41]. In addition, we include two latest backdoor attacks, NARCISSUS [75] and COMBAT [29], which are both robust clean-label attacks. The experiments are conducted using the ResNet-18 model and CIFAR-10 dataset, with 5% of the original training samples available for defense. Results, presented in Table 3, indicate that while all defense techniques are effective against conventional attacks like BadNets, they perform less effectively against more complex and recent attacks, particularly NARCISSUS and COMBAT. UNIT consistently surpasses other state-of-the-art methods in mitigating backdoor effects, at the comparable cost to clean accuracy. However, it is important to note that while

UNIT demonstrates superior performance, it still falls short against the latest clean-label attacks. This may be due to the extremely subtle distribution differences between poisoned and clean activations introduced by these attacks, making them difficult for UNIT to approximate. Our future work will focus on improving UNIT's effectiveness against these robust clean-label attacks.

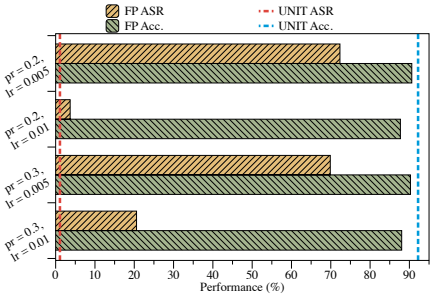


**Fig. 8:** Parameter sensitivity of FP

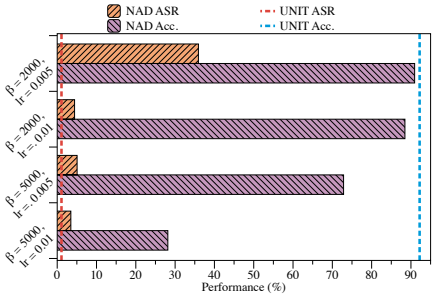


**Fig. 9:** Parameter sensitivity of NAD

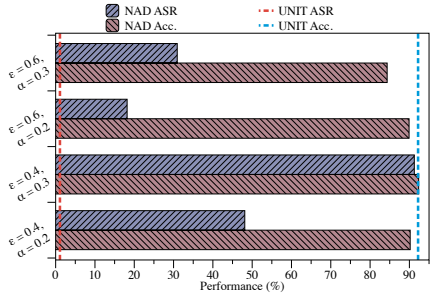


**Fig. 10:** Parameter sensitivity of ANP

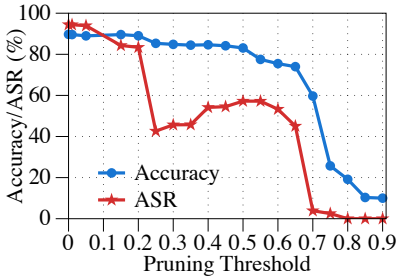


**Fig. 11:** Different pruning ratios

### A.3 Parameter Sensitivity Analysis

We take three typical baselines to illustrate that existing methods are sensitive to their parameters and require sophisticated parameter tuning to ensure the good performance. However, UNIT is parameter-efficient and outperforms the existing methods. We conduct experiments using CIFAR-10 dataset and ResNet-18 network. We inject CL [65] backdoor into the model and apply FP [38], NAD [35] and ANP [69] to mitigate the attack. For each defense, we take two key parameters and evaluate the performance for different parameter values. Specifically, we take the pruning ratio ($pr$) and learning rate ($lr$) for FP, distillation strength ($\beta$) and learning rate ($lr$) for NAD, and adversarial perturbation ($\epsilon$) and pruning coefficient ($\alpha$) for ANP. Results are presented in Figure 8, Figure 9 and Figure 10, where the y-axis denotes the parameter values while the x-axis presents the performance in percentage (accuracy or ASR). For each parameter setting,

we visualize the performance using two bars, i.e., resulting ASR (top bar) and Acc. (lower bar). The red and blue dashed lines represents the resulting ASR and accuracy after applying UNIT. Observe that the performance of existing methods are significantly sensitive to parameter tuning, with large fluctuation over slightly different parameters. On the contrary, UNIT is parameter-efficient and outperforms the three baselines, indicating its practical applicability.

In addition, we take the state-of-the-art pruning method RNP [36] to defend against the Reflection backdoor [41]. Figure 11 shows that as the pruning threshold increases (more neurons are pruned), both accuracy (blue curve) and ASR (red curve) degrade similarly. This indicates that some neurons handle both clean and backdoor tasks, and no matter how the parameters are tuned to control the pruning rate, it is fundamentally difficult to eliminate the backdoor effect without a non-trivial accuracy cost (Section 3). In contrast, UNIT achieves 91.44% accuracy and reduces ASR to 6.63% by precisely tightening the neural distribution.

### A.4 Adaptive Attacks

In this section, we discuss three adaptive attack scenarios in detail.

**Activation Suppression.** To tamper UNIT's effectiveness, an adversary may attempt to bridge the gap between benign and backdoor activation. Specifically, an adaptive loss is incorporated during training to suppress the backdoor activation:

$$Loss = \mathcal{L}(M(x), y) + \mathcal{L}(M(x \oplus T), y_T) + \alpha \cdot \sum_{l=1}^{L} ||F^l(x) - F^l(x \oplus T)||_2^2, \quad (6)$$

Here, $M$ denotes the model, $x$ and $y$ denote clean images and their labels, $x \oplus T$ denote their poisoned versions and $y_T$ is the attack target class. $F^l(x)$ represents the activation value of $x$ at the $l$-th layer as defined in Section 4.1. The adaptive loss term $||F^l(x) - F^l(x \oplus T)||_2^2$ uses Mean Squared Error (MSE) to reduce the difference between benign and poisoned activation. The parameter $\alpha$ controls the trade-off between the adaptive loss term and the normal cross-entropy loss $\mathcal{L}$. The experiment is conducted on CIFAR-10 and ResNet18 using BadNets as the backdoor attack. We evaluate five $\alpha$ values: 0, 0.1, 1, 10, 100. Table 4 presents the results, where the first column shows $\alpha$ values, the second and third columns present accuracy and ASR without defense, the fourth column shows the adaptive loss, and the last two columns present accuracy and ASR after applying UNIT. Observe that with the increase of $\alpha$, the adaptive loss decreases, signifying successful activation suppression. However, UNIT continues to effectively reduce the ASR to less than 1.07%. The reason is that despite the reduced backdoor activation, UNIT can still effectively clip the slightly higher malicious activation.

**Label-specific Backdoor.** Attackers may employ label-specific strategies to impact the efficacy of UNIT. In label-specific backdoor attacks, the backdoor exclusively influences samples belonging to the victim class. Samples stamped

**Table 4:** Evaluation against activation suppression

| $\alpha$ | No Defense | | | UNIT | |
|---|---|---|---|---|---|
| | Acc. | ASR | Adap-Loss | Acc. | ASR |
| 0 | 94.84% | 100.00% | 0.1108 | 93.05% | 1.07% |
| 0.1 | 94.25% | 100.00% | 0.0433 | 92.52% | 0.84% |
| 1 | 94.13% | 100.00% | 0.0137 | 92.42% | 0.73% |
| 10 | 93.77% | 100.00% | 0.0034 | 91.85% | 1.02% |
| 100 | 93.42% | 100.00% | 0.0015 | 91.35% | 0.90% |

with the trigger but not from the victim class will not be misclassified into the attack target class. Consequently, label-specific backdoor attacks rely on normal features and could potentially impact the difference between clean and malicious activation distributions. We conduct experiments on CIFAR-10 using ResNet18 and utilized BadNets as a representative attack. When poisoning the model, we not only introduce images of the victim class stamped with the trigger and labeled as the target class but also incorporate negative samples to achieve label-specificity. Negative samples consist of images from classes other than the victim class, stamped with the trigger and labeled as their source labels. These negative samples aid the model in learning the correlation between the backdoor trigger and the victim class. Table 5 presents the results, with the first column denoting the attack victim-target pair, the second and third columns representing clean accuracy and ASR without defense, the fourth column illustrating the effectiveness of label-specificity (ASR of images stamped with the trigger from non-victim classes), and the last two columns displaying the results after applying UNIT. Observe that negative samples facilitate label-specificity, reducing the ASR of non-victim classes from 100% to nearly 3%, while the ASR of the victim class remains high at approximately 98%. After applying UNIT, the clean accuracy remains high with only about a 2% degradation, while the ASR decreases to less than 4.40%, demonstrating the effectiveness of UNIT against label-specific attacks. The reason is that even if the backdoor relies on benign features, it still necessitates reasonably large activation values to be triggered. UNIT identifies and mitigates these large activation values, rendering it effective against label-specific attacks.

**Table 5:** Adaptive attack through leveraging label specificity.

| V-T Pair | No Defense | | Specificity | UNIT | |
|---|---|---|---|---|---|
| | Acc. | ASR | Non-victim ASR | Acc. | ASR |
| 0-9 | 94.62% | 98.60% | 100.00% → 2.88% | 92.31% | 1.50% |
| 8-3 | 94.83% | 98.60% | 100.00% → 2.51% | 92.64% | 0.90% |
| 2-4 | 94.71% | 97.90% | 100.00% → 3.20% | 92.74% | 4.40% |
| 6-5 | 94.89% | 98.40% | 100.00% → 3.04% | 92.60% | 0.00% |
| 7-1 | 94.79% | 98.20% | 100.00% → 2.06% | 92.97% | 0.00% |

**Trigger-specific Backdoor.** Attackers may exploit trigger-specificity to dynamically impact UNIT. Trigger-specificity entails that the backdoor is activated only when a specific pattern is presented in the image. In other words, a ground-truth

trigger stamped with some noise will not induce the backdoor effect. The potential impact on the effectiveness of UNIT arises from the model's potential use of benign neurons to extract high-level features of the trigger pattern. This could lead to a reduction in the difference between benign and backdoor activation distributions. We conducte experiments on CIFAR-10 using ResNet18 and employ BadNets as an example attack, utilizing a yellow flower as the backdoor trigger. In addition to poisoned samples stamped with the trigger and relabeled as the target class, we introduce negative samples to establish trigger-specificity. We added Gaussian noise to the trigger pattern, stamping the noisy pattern onto certain training images while keeping their ground-truth labels. This approach help the model learn the high-level trigger pattern instead of some low-level features. Table 5 presents the results, where the first column denotes the noise level added to the negative samples, the second and third columns represent clean accuracy and ASR without defense, the fourth column illustrates the effectiveness of trigger-specificity (ASR of images stamped with noisy triggers), and the last two columns display the results after applying UNIT. Observe that the noisy ASR in the fourth column is significantly reduced when the noise level is $0.05, 0.1, 0.5, 1.0$, indicating that negative samples effectively realize trigger-specificity. Notably, in the last two columns, UNIT still reduces the ASR from 100% to nearly 1%, while maintaining high clean accuracy. UNIT proves effective against trigger-specific attacks because even if the model learns the high-level backdoor trigger pattern, it cannot circumvent the separation between clean and malicious distributions. This discrepancy is leveraged by UNIT to mitigate the backdoor effect.

**Table 6:** Adaptive attack through leveraging trigger specificity.

| Level | No Defense | | Specificity | UNIT | |
|---|---|---|---|---|---|
| | Acc. | ASR | Noisy Trigger ASR | Acc. | ASR |
| 0.01 | 94.65% | 100.00% | 100.00% → 99.98% | 92.55% | 1.01% |
| 0.05 | 94.79% | 100.00% | 100.00% → 6.84% | 92.85% | 0.86% |
| 0.1 | 94.70% | 100.00% | 100.00% → 0.59% | 92.58% | 1.06% |
| 0.5 | 94.64% | 100.00% | 77.08% → 0.64% | 92.65% | 0.87% |
| 1 | 94.99% | 100.00% | 36.50% → 0.76% | 92.65% | 0.92% |

### A.5 Ablation Study

In this section, we perform a series of experiments to assess the performance of UNIT under different attack and defense settings. Additionally, we conduct an ablation study on the design choices and hyper-parameters of UNIT.

**Different Activation Functions** There are different types of activation functions used in deep neural networks. We evaluate UNIT on BadNets-poisoned models using different activation functions. CIFAR-10 and ResNet18 are used for the study. We replace the standard activation function (ReLU) with five commonly used functions, LeakyReLU [43], SELU [32], ELU [12], TanhShrink [49],

**Table 7:** Evaluation on different activation functions

| Activation | Original | | UNIT | |
|---|---|---|---|---|
| | Acc. | ASR | Accuracy | ASR |
| ReLU | 94.82% | 100.00% | 92.46% | 0.78% |
| LeakyReLU | 94.17% | 100.00% | 92.02% | 0.96% |
| SELU | 90.89% | 99.93% | 89.66% | 1.06% |
| ELU | 91.24% | 99.98% | 90.52% | 0.99% |
| TanhShrink | 89.80% | 100.00% | 89.21% | 1.13% |
| Softplus | 88.08% | 99.97% | 87.19% | 1.58% |
| Sigmoid | 80.43% | 99.80% | 77.62% | 5.31% |
| Tanh | 90.95% | 99.98% | 88.14% | 6.03% |

**Table 8:** Ablation study on different number given clean training samples

| Attacks | No Defense | | 10% | | 5% | | 1% | | 0.1% | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Acc. | ASR | Acc. | ASR | Acc. | ASR | Acc. | ASR | Acc. | ASR |
| BadNets | 94.82% | 100.00% | 92.81% | 1.09% | 92.48% | 0.78% | 90.60% | 1.59% | 88.01% | 2.63% |
| Trojan | 94.73% | 100.00% | 92.44% | 1.65% | 92.38% | 2.17% | 91.07% | 1.96% | 89.93% | 2.73% |
| CL | 94.58% | 98.46% | 92.36% | 0.93% | 92.21% | 1.09% | 89.98% | 1.81% | 90.90% | 8.09% |
| Dynamic | 95.08% | 100.00% | 92.34% | 1.28% | 92.77% | 1.54% | 88.48% | 2.01% | 90.29% | 2.14% |
| IA | 91.15% | 97.96% | 90.78% | 1.82% | 89.93% | 1.03% | 87.66% | 2.11% | 89.38% | 3.45% |
| Reflection | 93.29% | 99.33% | 91.64% | 2.74% | 91.44% | 6.63% | 88.82% | 7.47% | 83.02% | 0.46% |
| SIG | 94.97% | 99.80% | 92.45% | 0.41% | 92.48% | 1.74% | 89.30% | 0.46% | 86.88% | 1.18% |
| Blend | 94.62% | 100.00% | 91.75% | 1.55% | 91.99% | 1.18% | 88.77% | 2.06% | 90.86% | 0.61% |
| WaNet | 94.36% | 99.80% | 91.09% | 1.83% | 91.02% | 2.44% | 89.53% | 9.90% | 82.96% | 3.68% |
| ISSBA | 94.55% | 100.00% | 91.96% | 1.19% | 91.84% | 1.57% | 89.53% | 1.86% | 88.10% | 2.53% |
| LIRA | 95.11% | 100.00% | 92.65% | 0.82% | 92.29% | 0.58% | 89.49% | 1.93% | 82.76% | 1.17% |
| Instagram | 94.62% | 99.59% | 91.71% | 4.05% | 91.43% | 4.98% | 91.07% | 11.66% | 86.25% | 21.64% |
| DFST | 93.25% | 99.77% | 91.83% | 3.54% | 91.64% | 4.02% | 88.59% | 3.90% | 88.97% | 8.80% |
| Adap-Blend | 94.22% | 82.80% | 91.35% | 18.39% | 90.84% | 15.03% | 88.78% | 47.64% | 87.64% | 58.57% |
| Average | 94.26% | 98.39% | 91.94% | 2.95% | 91.77% | 3.20% | 89.41% | 5.88% | 87.57% | 8.41% |

Softplus [79], Sigmoid [22], and Tanh [17]. Table 7 shows the results. UNIT is effective in all the cases, reducing the ASR from near 100.00% to less than 1.6%. The impact on the clean accuracy is negligible (less than 2% degradation). This is attributed to the distinct separation between clean and poisoned activation distributions, a phenomenon that persists across various activation functions. Hence, UNIT is able to ensure robust performance irrespective of the activation function utilized.

**Different Numbers of Clean Training Data.** We study the impact of different numbers of given clean training samples on UNIT. Table 8 presents the results, comparing UNIT's performance when provided with 10%, 5% (default), 1%, and 0.1% of training data for defense. The experiments are conducted on CIFAR-10 and ResNet-18. Observe that in general, UNIT exhibits better performance (higher clean accuracy and lower ASR) when given more clean samples. This is expected as UNIT requires clean samples to approximate a tight distribution and eliminate malicious activation. More samples allowing for a more precise approximation of the real distribution. Obtaining clean samples is generally feasible, even from the internet, supporting the argument that UNIT is generally effective and suitable for real-world applications. Moreover, even with access to only 0.1% of samples (totally 50 images in CIFAR-10 dataset), UNIT still effectively reduces ASR to an average of 8.41%, albeit with a sacrifice of around 7% in accuracy.

**Table 9:** Evaluation on different low poisoning rates

| Attacks | No Defense | | FP | | NAD | | ANP | | UNIT | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Acc. | ASR | Acc. | ASR | Acc. | ASR | Acc. | ASR | Acc. | ASR |
| BadNets (1%) | 93.87 | 100.0 | 89.78 | 52.32 | 88.69 | 38.62 | 88.15 | 7.56 | **92.72** | **0.85** |
| BadNets (0.1%) | 94.05 | 100.0 | 89.66 | 99.84 | 89.72 | 49.05 | 86.45 | 37.27 | **90.22** | **1.44** |
| Blend (1%) | 93.92 | 99.92 | 89.43 | 29.69 | 89.64 | 0.07 | 88.16 | 0.10 | **92.96** | **0.00** |
| Blend (0.1%) | 93.88 | 99.89 | 88.87 | 41.56 | 88.64 | 0.20 | 87.51 | 0.48 | **92.14** | **0.11** |
| WaNet (1%) | 93.59 | 94.91 | 89.01 | 5.77 | **91.31** | 10.43 | 89.93 | **1.48** | 90.97 | 2.19 |
| WaNet (0.1%) | 93.80 | 93.75 | 89.79 | 9.89 | 90.22 | 21.39 | **90.26** | **1.44** | 90.08 | 4.07 |

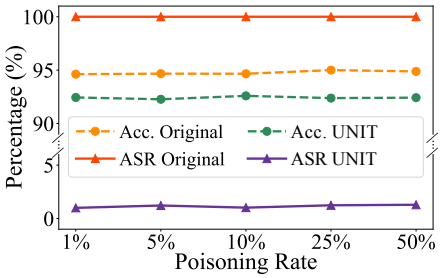


**Fig. 12:** Ablation study on different poisoning rates

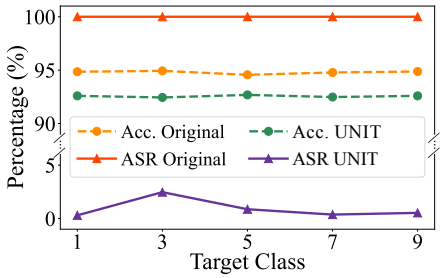


**Fig. 13:** Ablation study on different target labels

**Different Poisoning Rates.** We study the impact of various poisoning rates on defense performance. Utilizing CIFAR-10 and ResNet-18, we introduce BadNets triggers with poisoning rates of 1%, 5%, 10%, 25%, and 50%. Figure 12 illustrates the results, with the x-axis representing poisoning rates and the y-axis denoting performance. Notably, UNIT consistently reduces the ASR from 100.00% to approximately 1%, with minimal accuracy sacrifice (less than 2%). This highlights the robustness of UNIT across different data poisoning rates.

Additionally, we compared UNIT with three baselines on CIFAR-10 and ResNet-18 at extremely low poison rates (1% and 0.1%). Table 9 shows that while baseline performance degrades at lower poison rates, UNIT remains robust and outperforms them.

**Different Target Labels.** We investigate the influence of different target labels on defense performance of UNIT. We use CIFAR-10, ResNet-18 and BadNets trigger to conduct the experiments and employ labels 1, 3, 5, 7, and 9 as the targets. Figure 13 illustrates the results, where the x-axis represents the target classes, and the y-axis indicates performance. Remarkably, UNIT consistently reduces the ASR from 100.00% to 0.30%-2.47% without significantly impacting clean accuracy, underscoring the robustness of UNIT across various target labels.

**Time cost for different model scales.** We evaluate the time cost of UNIT for different model scales. Results are shonw in Figure 14. For larger models, e.g., ResNet-101, UNIT reduces ASR to below 2% with only a 3% accuracy drop. The

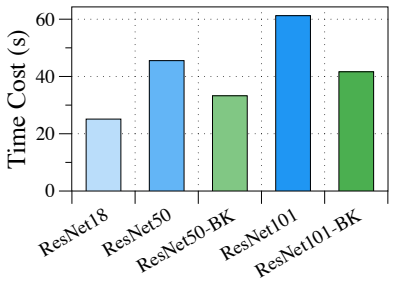


**Fig. 14:** Ablation study on different model scales

**Table 10:** Ablation study on different design choices

| Attacks | No Defense | | Current Setting | | Act. Rejection | | First Two Layers | | Last Two Layers | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Acc. | ASR | Acc. | ASR | Acc. | ASR | Acc. | ASR | Acc. | ASR |
| BadNets | 94.82% | 100.00% | 92.48% | 0.78% | 90.52% | 1.97% | 92.88% | 3.19% | 90.74% | 1.42% |
| Trojan | 94.73% | 100.00% | 92.38% | 2.17% | 90.93% | 4.67% | 91.88% | 5.29% | 90.95% | 2.48% |
| Reflection | 93.29% | 99.33% | 91.44% | 6.63% | 90.77% | 23.89% | 91.39% | 74.80% | 90.43% | 7.95% |
| Instagram | 94.62% | 99.59% | 91.43% | 4.98% | 91.10% | 12.03% | 91.93% | 78.40% | 91.34% | 7.17% |
| DFST | 93.25% | 99.77% | 91.64% | 4.02% | 91.39% | 8.62% | 92.14% | 76.58% | 90.99% | 34.76% |
| Adap-Blend | 94.22% | 82.80% | 90.84% | 15.03% | 90.82% | 39.93% | 91.84% | 78.69% | 90.39% | 57.76% |
| Average | 94.26% | 98.39% | 91.77% | 3.20% | 90.92% | 15.19% | 92.01% | 52.83% | 90.81% | 18.59% |

time cost increases with model size (blue bars), but even for ResNet-101, UNIT completes in about 1 minute. To enhance efficiency, we can optimize only on *key layers*, e.g., each residual block in ResNets. This adaptation reduces the time cost by 25% (green bars) while still being effective. This demonstrates UNIT's efficiency even for large-scale models.

**Comparison Between Clipping and Rejection.** We conduct a comparison of two approaches within UNIT for handling maliciously large activation values after the distribution approximation, i.e., clipping and rejection. Clipping reduces large values to the distribution boundary value, while rejection directly sets outlier values to zero. Our experiment is performed on CIFAR-10 and ResNet-18, with results presented in the first half of Table 10. The first column denotes different attacks, Columns 2-3 present the attack performance without defense, Columns 4-5 denote the defense performance of the current setting of UNIT (activation clipping), and Columns 6-7 show the performance using activation rejection. Notably, clipping generally provides superior performance, resulting in higher accuracy and lower ASR compared to rejection. The underlying reason is that rejection, similar to neuron pruning, is coarse-grained. Specifically, for neurons responsible for extracting both benign and backdoor features, rejection harms accuracy while not rejecting retains the backdoor effect. In contrast, clipping eliminates only the higher values while allowing the extraction of benign features.

**Table 11:** Ablation study of different customized accuracy degradation.

| Degradation | Accuracy | ASR |
|---|---|---|
| No Defense | 94.73% | 100.00% |
| 0.01% | 94.08% (-0.65%) | 4.01% |
| 0.1% | 93.89% (-0.84%) | 3.56% |
| 2% | 92.38% (-2.35%) | 2.17% |
| 5% | 88.36% (-6.37%) | 1.40% |
| 10% | 84.21% (-10.52%) | 1.12% |

**Table 12:** Ablation study of different optimization steps and learning rates.

| Config | No Defense | 10 & 0.01 | 10 & 0.001 | 50 & 0.01 | 50 & 0.001 |
|---|---|---|---|---|---|
| BA | 93.51% | 91.79% | 91.31% | 92.14% | 91.41% |
| ASR | 100.00% | 4.76% | 5.49% | 3.82% | 4.91% |

**Comparison of Operating on Different Layers.** We examine the impact of applying UNIT to different layers: (1) All layers (current setting), (2) Only the first two layers, and (3) Only the last two layers. The results are presented in the last half columns of Table 10. Observations indicate that applying UNIT to all layers, the current setting, generally yields the best performance compared to operating only on the first two or last two layers. Notably, for simple backdoors such as BadNets and Trojan, where most features are extracted in the first few layers, applying UNIT to the first two layers is sufficient to eliminate the backdoor effect while preserving clean accuracy. However, for complex backdoors like Reflection and Instagram, where backdoor features are extracted in later layers of the network, applying UNIT to the last few layers achieves better performance. Additionally, advanced attacks such as DFST and Adap-Blend, which tend to hide backdoor extraction across almost all layers, can only be effectively defended against by applying UNIT to all layers.

**Effect of Setting Different Accuracy Degradation.** We examine the impact of different accuracy degradation expectations for UNIT. In our experiment, we assess 5 accuracy degradation values (default is 2%). The evaluated attack model is trained on CIFAR-10 using ResNet18 and injected with the Trojan [40] backdoor. The results in Table 11 indicate that as the degradation increases, both accuracy and ASR decrease. This reveals a trade-off between sacrificed clean accuracy and remaining ASR. Users seeking high clean accuracy with some tolerance for backdoor may opt for a low accuracy degradation, and vice versa.

**Effect of Different Optimization Steps and Learning Rates.** We study the effect of different optimization steps ($S$) and learning rates ($\eta$). The evaluated attack model is trained on CIFAR-10 using ResNet18 and injected with the Blend [6] backdoor. The results are shown in Table 12, where "10 & 0.01" means $S = 10$ and $\eta = 0.01$. We observe that UNIT demonstrates consistently good performance across various reasonable parameter settings, showcasing its robustness and efficiency in parameter tuning.