A MORE IMPLEMENTATION DETAILS

The backdoor triggers used in our experiments are shown in Figure 6.

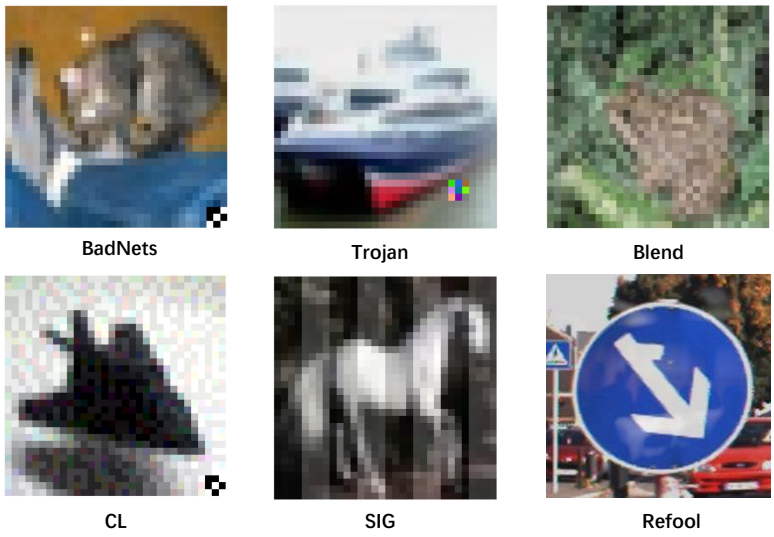


Figure 6: Examples of backdoored CIFAR-10 images by the 6 attacks.

Table 3: A configuration summary for the 6 backdoor attacks: datasets, models, and triggers.

Backdoork	BadNets	Trojan	Blend	Clean-Label	Signal	Refool
Dataset	CIFAR-10	CIFAR-10	CIFAR-10	CIFAR-10	CIFAR-10	GTSRB
Model	WideResNet	WideResNet	WideResNet	WideResNet	WideResNet	WideResNet
Inject Rate	0.1	0.05	0.1	0.8	0.1	0.8
Trigger Type	Grid	Square	Random Noise	Grid + PGD Noise	Sinusoidal Signal	Reflection
Trigger Size	3×3	3×3	Full Image	3×3	Full Image	Full Image
Target Label	0	0	0	0	0	0
ASR	100.00%	100.00%	99.97%	99.21%	99.91%	95.16%
ACC	85.65%	81.24%	84.95%	82.43%	84.36%	82.38%

Detailed implementation on 6 state-of-the-art backdoor attacks:

- BadNets: The trigger is a 3×3 checkerboard (pixel values are 128 or 255) at the bottom right corner of images. We labeled the backdoor examples with a chosen target label and achieved an attack success rate of 100% with an injection rate of 10%.
- Trojan attack. We follow the method proposed in the paper to reverse engineer a 3×3 square trigger from the last fully-connected layer of the network. In order to reduce the impact on clean accuracy, we poisoned only 5% of training data with the reverse-engineered Trojan trigger. We achieved an attack success rate of 100% with an injection rate of 5%.
- Blend attack. We used the random patterns reported in the original paper. We achieved an attack success rate of 99.97% with an injection rate of 10% and a blend ratio of $\alpha = 0.2$.
- Clean-label attack (CL). We followed the same settings as reported in the paper. Specifically, we used Projected Gradient Descent (PGD) to generate adversarial perturbations bounded to L_1 maximum perturbation $\epsilon = 16$. The trigger is a 3×3 grid at the bottom right corner of images. We achieved an attack success rate of 99.21% with an injection rate of 80%.

- Sinusoidal signal attack (SIG). We generate the backdoor trigger following the horizontal sinusoidal function defined in their paper with $\Delta = 20$ and $f = 6$. We achieved an attack success rate of 99.91% with an injected rate of 10%.

- Reflection attack (Refool). The implementation is based on the open-source code[2]. We achieved an attack success rate of 95.16% with an injection rate of 80%.

More Details on Defense Baselines We adopted the same settings used in NAD for the standard finetuning approach and finetuned the model until convergence. We replicated Fine-pruning[3] via PyTorch and pruned the last convolutional layer of the model as suggested in the original paper Liu et al. (2018a). For a fair comparison, the pruning ratio was set to a value such that the ACC of the pruned network matched the ACC of our NAD approach. We used the open-source code[4] for mode connectivity repair (MCR) and set the endpoint model t = 0 and t = 1 with the same backdoored WRN-16-1. We trained the connection path for 100 epochs and evaluated the defense performance of the model on the path. Other settings of the code remain unchanged.

B COMPARISON WITH DATA AUGMENTATION TECHNIQUES

Cutout (DeVries & Taylor, 2017) and Mixup (Zhang et al., 2018) are popular data augmentation methods for CNNs. Cutout masks out random sections of input images during training and Mixup randomly morphs the training images. We evaluate in this section the independent effectiveness of Mixup and Cutout in erasing backdoor triggers. For Cutout[5], we set the number of patches to be cut out of each image to 1 and each patch is a 3×3 square. For Mixup[6], we set α to be the default value of 1, indicating that we sample the weight uniformly between zero and one. Other settings for attacks and defenses are identical to the settings specified in Section 4.1. The results (see Table 4) can be a supplement of Table 1. We conclude that data augmentation techniques have mitigating effects on backdoors only when the transformation images are similar to the trigger patterns. They are hence not general against a wide range of backdoor attacks.

Table 4: Comparison with Mixup and Cutout on erasing backdoor triggers.

Backdoor	Before		Mixup		Cutout		NAD (Ours)	
Attack	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
BadNets	100%	85.65%	68.22%	80.27%	28.17%	**82.73%**	**3.81%**	81.85%
Trojan	100%	81.24%	96.20%	71.83%	50.22%	**80.13%**	**19.63%**	79.16%
Blend	99.97%	84.95%	99.11%	80.51%	15.30%	**81.78%**	**3.04%**	81.68%
CL	99.21%	82.43%	93.77%	77.13%	73.33%	**81.34%**	**9.18%**	80.34%
SIG	99.91%	84.36%	52.11%	79.94%	99.95%	**82.77%**	**2.52%**	81.95%
Refool	95.16%	82.38%	8.76%	77.84%	91.86%	80.06%	**3.18%**	**80.73%**
Average	99.04%	83.50%	69.69%	77.92%	59.80%	**81.46%**	**7.22%**	80.83%
Deviation	-	-	↓ 29.35%	↓ 5.58%	↓ 39.24%	↓ **2.03%**	↓ **91.82%**	↓ 2.66%

C MORE RESULTS OF MODE CONNECTIVITY REPAIR (MCR)

We use the open-source code of MCR and compare its performance to our NAD method. The experiments are conducted on CIFAR-10 dataset using 5% clean finetune data. We first run MCR with the two endpoint models t = 0 and t = 1 which use the same backdoored WRN-16-1 model. Figure 7 shows the convergence rate of MCR and our NAD against BadNets attack. We then run an additional experiment for MCR using two different endpoint models: t = 0 and t = 1 use the backdoored WRN-16-1 and the finetuned backdoored WRN-16-1 respectively. This result is reported in Table 5. We find that using different endpoint models can not further improve the performance of MCR.

[2]https://github.com/DreamtaleCore/Refool

[3]https://github.com/kangliucn/Fine-pruning-defense

[4]https://github.com/IBM/model-sanitization/tree/master/backdoor/backdoor-cifar

[5]https://github.com/uoguelph-mlrg/Cutout

[6]https://github.com/leehomyc/mixup-pytorch

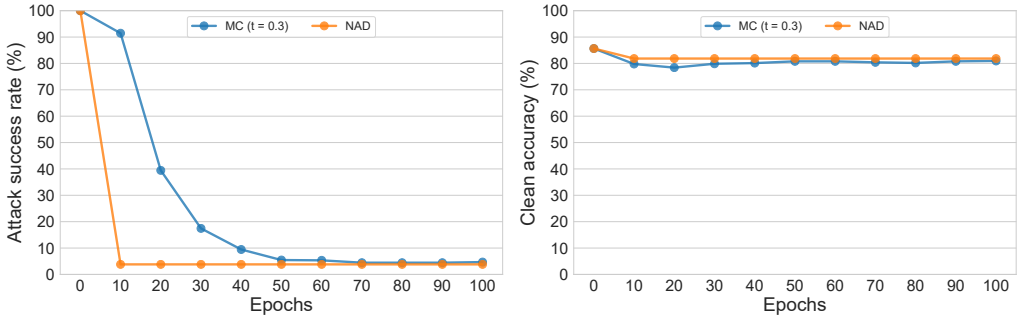


Figure 7: Convergence rate comparison between MCR and NAD against BadNets attack with 5% clean training data. We show the best result of MC at the connection point t = 0.3. Note that it takes longer for MCR to converge yet its ASR is still higher than that of the NAD's.

Table 5: Performance of MCR with different endpoint models on CIFAR-10 dataset. **B** denotes the backdoored WRN-16-1 and **F-B** denotes the backdoored WRN-16-1 after Fine-tuning. MCR-(B,B) denotes the default setting where the two endpoint models are both B, while MCR-(B,F-B) denotes the MCR using two different endpoint models B and F-B. The best results are **boldfaced**.

Backdoor	Before		MCR-(B,B)		MCR-(B,F-B)		NAD (Ours)	
Attack	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
BadNets	100%	85.65%	**4.65%**	80.94%	6.00%	80.56%	4.77%	**81.17%**
Trojan	100%	81.24%	41.25%	78.76%	53.31%	78.31%	**19.63%**	**79.16%**
Blend	99.97%	84.95%	64.33%	80.34%	70.65%	80.51%	**4.04%**	**81.68%**
CL	99.21%	82.43%	32.95%	79.04%	42.66%	80.31%	**9.18%**	**80.34%**
SIG	99.91%	84.36%	**1.62%**	80.94%	7.32%	81.12%	2.52%	**81.95%**
Refool	95.15%	82.38%	8.76%	78.84%	10.95%	79.03%	**3.18%**	**80.73%**

D EXPERIMENTAL RESULTS OF TRIGGER RECOVERING TECHNIQUE

Qiao et al. (2019) proposed MESA that recovers the trigger distribution via generative modeling and then removes the backdoor by model retraining. We implemented this work based on their open-source code[7]. Note that we report the best averaging results of defense performance and we changed nothing in the code besides setting the proportion of available training data to 5%. We present the results in Table 5.

Table 6: Comparison between NAD and retraining-based approaches that use both the original trigger (Org-T) and the MESA-recovered trigger (Rec-T). While all methods are able to reduce the ASR of BadNets to a similar level, NAD is able to reduce the ASR of CL by 16 more percent in comparison to the model retrained with the original trigger and by 22 more percent in comparison to the model retrained with the MESA-generated trigger.

Backdoor	Before		Retrain w/ rec-T		Retrain w/ org-T		NAD (Ours)	
Attack	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
BadNets	100%	85.65%	4.96%	81.23%	**3.91%**	**82.14%**	4.77%	81.17%
CL	99.21%	82.43%	31.23%	79.12%	25.23%	79.57%	**9.18%**	**80.34%**

E EXPERIMENTAL RESULTS OF HYPER-PARAMETER β

We only give a rough estimate of β for all the backdoor attacks in Figure 8 and it certainly provides better results by a more granular level of tuning.

[7]https://github.com/superrrpotato/Defending-Neural-Backdoors-via-Generative-Distribution-Modeling

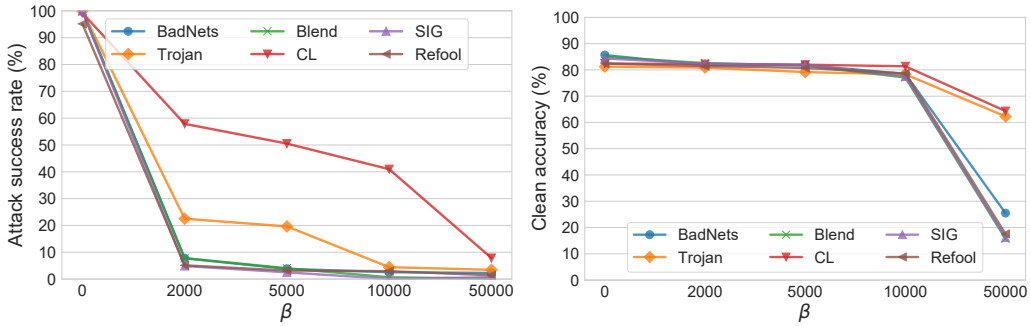


Figure 8: Parameter analysis: performance of our NAD approach under different β.

F EXPERIMENTAL RESULTS OF DIFFERENT ATTENTION FUNCTIONS

We compare the performance of NAD under scenarios where 4 different attention functions, $\mathcal{A}_{mean}$, $\mathcal{A}^2_{mean}$, $\mathcal{A}_{sum}$, and $\mathcal{A}^2_{sum}$ are employed. We use the BadNets attack as our benchmark attack. Again, we evaluate the performance of NAD using two metrics, the ASR and the ACC, and the results are summarized in Table 7.

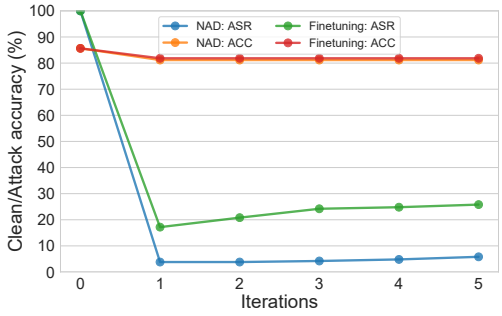


Figure 9: Iterative NAD and Finetuning against BadNets.

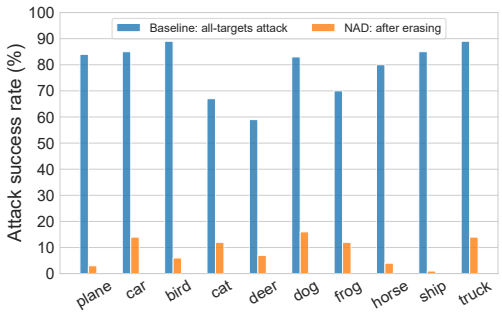


Figure 10: Erasing all-target BadNets attack.

Table 7: Performance of NAD using different attention functions against BadNets on CIFAR-10 with 5% clean data. ASR: attack success rate; ACC: clean accuracy. The best results are in **boldfaced**.

Attention Function	$\mathcal{A}_{mean}$		$\mathcal{A}^2_{mean}$		$\mathcal{A}_{sum}$		$\mathcal{A}^2_{sum}$	
	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
Baseline	100%	85.86%	100%	85.86%	100%	85.86%	100%	85.86%
Epoch 1	11.81%	66.72%	1.98%	47.52%	9.38%	68.81%	1.36%	47.25%
Epoch 2	16.34%	79.92%	8.86%	78.14%	10.82%	79.34%	9.81%	77.91%
Epoch 3	13.86%	81.83%	4.50%	81.00%	7.67%	81.69%	5.12%	81.12%
Epoch 4	14.16%	81.90%	5.96%	80.67%	8.39%	81.38%	5.80%	80.83%
Epoch 5	12.28%	81.50%	4.60%	81.30%	6.89%	81.46%	**4.21%**	**81.55%**

G EXPERIMENTAL RESULTS OF ITERATIVE NAD

We evaluate whether NAD can be further improved with multiple iterations of distillation. In this experiment, we adopted the same configuration and set the iteration times to 5. Taking the BadNets attack as an example. The results in Figure 9 show that the attack rate has not been further reduced, and has even slightly increased by 2% in some cases. We hypothesize that the attentions of the backdoored neurons have been correctly aligned with the attentions of the benign neurons after a single-iteration of erasing. Whereas multiple iterations of distillation will make NAD refocus on the trigger pattern. Therefore, we believe that one-iteration of distillation is sufficient to guarantee

the best result. Note that iterative Finetuning does not lead to further improvement over one-time finetuning neither.

H ERASING ALL-TARGET BACKDOOR ATTACKS

Unlike a single-target attack where the goal is to misclassify all backdoored images as one pre-specified target class, an all-target attack aims to misclassify every source class label as different ones (in our case, misclassify the original label i as $(i + 1)\ \%\ 10$). In this experiment, we adopted the same settings (i.e. single target attacks on BadNets) to conduct all-target attacks on the WRN-16-1 network. We found that an all-target attack is a tougher task than a single-target attack. It is harder to attain a satisfactory ASR with an all-target attack. The results in Figure 10 show that NAD is able to reduce the ASR across all poison-classes (from 79% to 9.7%) effectively with only 5% of clean training data.

I FEATURE MAPS V.S. ATTENTION MAPS

A natural question to ask is: *why attention maps instead of feature maps?* This can be traced back to the field of knowledge distillation. Directly aligning the feature maps could lead to an information loss on the sample density in the space, and this could lead to a decrement in the distillation performance (Zagoruyko & Komodakis, 2017; Huang & Wang, 2017; Lopez et al., 2019). In the context of backdoor erasing, aligning the feature maps is not a good option because the backdoor neurons are only weakly, if not at all, activated by clean samples (Gu et al., 2019). In contrast, attention maps contain integrated information (see Equation 1) of both backdoored and benign neurons' feature maps, even when the neurons are not fired. (see Table 8). Figure 11 visualizes activation maps of a backdoored image on BadNets, Finetuned BadNets with 5% of clean training data, and BadNets erased by NAD with 5% of clean training data. The attention maps aggregated across the channels using 5 different attention functions are shown in Figure 12.

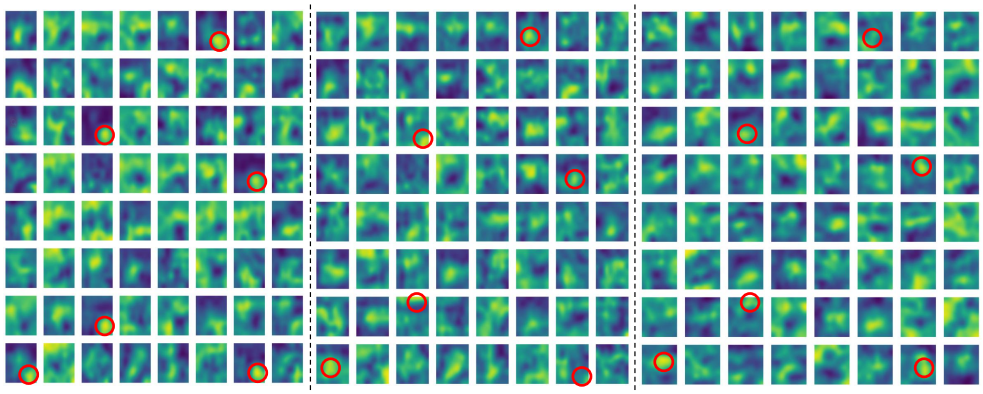


Figure 11: The activation map of one backdoored image at Group 3 of WRN-16-1 for (a) BadNets, (b) Finetuned BadNets with 5% of clean training data, and (c) BadNets erased by our NAD with 5% of clean training data. Each small patch is a channel (64 channels in total). The small red circles highlight the regions that are fired by the trigger pattern at different channels of the activation map.

Table 8: NAD using attention map versus activation map against BadNets.

		Before	Activation Map	Attention Map
CIFAR-10	ASR	100%	98.44%	**3.81**%
(WRN-16-1)	ACC	85.65%	82.66	81.85%

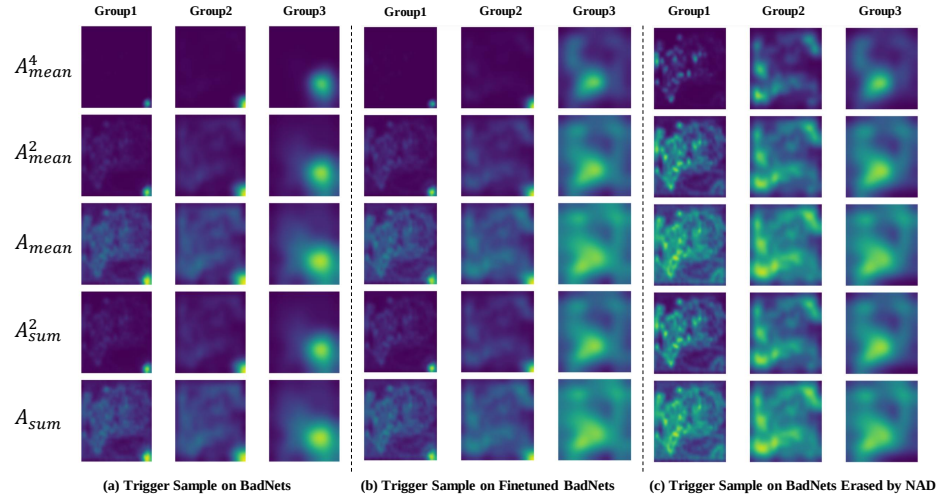


Figure 12: The attention maps derived by 5 different attention functions are shown for (a) BadNet, (b) Finetuned BadNet by 5% clean training data, and (c) BadNets erased by our NAD.

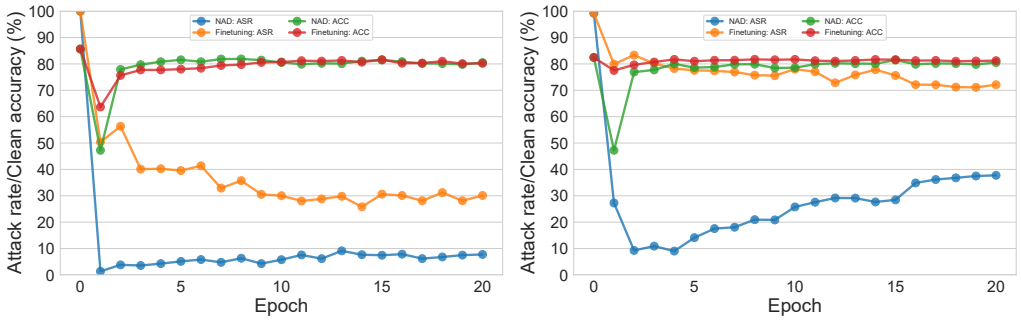


Figure 13: The learning curves (test ASR and ACC) of the NAD student network and a Finetuning network on CIFAR-10 against BadNets (left) and CL (right). In NAD, the student network tends to overfit to the teacher network, unless an early stopping is applied based on the validation ACC.

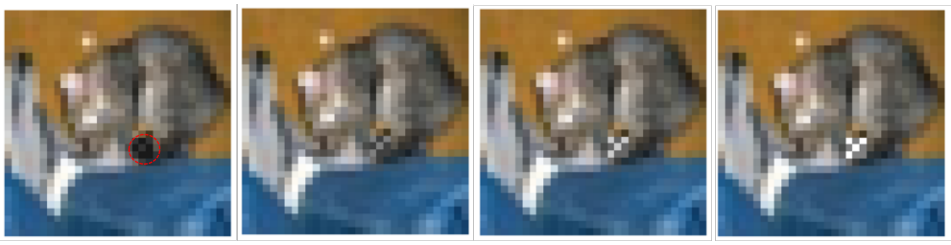
J OVERFITTING IN NAD

Here, we run NAD for a sufficiently long time (e.g. 20 epochs) to test if the student will eventually overfit to the finetuned teacher network. This experiment is conducted on CIFAR-10 against BadNets and CL attacks. We also run the Finetuning defense as a comparison. Note that, the teacher network of NAD is only finetuned for 10 epochs, following the settings in Section 4.1. As shown in Figure 13 (Appendix J), the student network of NAD can indeed overfit to the partially purified teacher network. However, this can be effectively addressed by a simple early stopping strategy: stop the finetuning when there are no significant improvements on the validation accuracy within a few epochs (e.g. at epoch 5). As shown by the green curves, the clean accuracy of NAD first drops, then quickly recovers and stabilizes at a high level within a few epochs. This also highlights the efficiency of our NAD defense as only a few epochs of finetuning is sufficient to erase the backdoor trigger.

K EFFECTIVENESS AGAINST ADAPTIVE ATTACKS

A backdoor adversary may attempt to construct a more stealthy backdoor trigger that does not cause obviouse shift of the attention. To simulate this scenario, we design an adaptive version of BadNets on CIFAR-10 that attaches the trigger pattern at the center region of the image. Such an adaptive attack will only shift the attention close to the center region and has a weaker activation response. Since most of the CIFAR-10 objects are located at the center of the clean images, this adaptive attack may make the attention distillation much less effective. Figure 14 illustrates a few examples of backdoored images for this type of attack. We use a scaling parameter $\alpha \in [0, 1]$ to adjust the pixel value of a black-white square trigger pattern (all images are normalized into the range of $[0, 1]$).

For example, for $\alpha = 0.2$, we scale pixel values of the trigger pattern p to $p \times \alpha$. The results of our NAD against this adaptive attack are reported in Table 9. Our NAD method can still effectively erase the adaptive attack while maintaining high accuracy on clean data. Interestingly, Finetuning can only effectively erase the weaker trigger, and not as effective as NAD (especially in the case of $\alpha = 1.0$ attack). We conjecture this is because the center regions are more hard overwritten by the clean images used for finetuning. We leave the exploration of more advanced adaptive attacks for future work.



(a) Square Trigger α = 0.2 (b) Square Trigger α = 0.5 (c) Square Trigger α = 0.8 (d) Square Trigger α = 1.0

Figure 14: The triggers used by an adaptive BadNets attack against our NAD under different α (a scaling factor of the original black-white square). The trigger patterns are all placed at the center of the image.

Table 9: Performance of our NAD (β = 2,0000 for α = 1.0 and β = 1,0000 for other α) against an adaptive BadNets attack. ASR: attack success rate; ACC: clean accuracy. The best results are **boldfaced**.

Square	Before		Finetuning		NAD (Ours)	
Trigger	ASR	ACC	ASR	ACC	ASR	ACC
$\alpha = 0.2$	99.85%	82.11%	7.51%	79.26%	**4.92%**	**80.32%**
$\alpha = 0.5$	99.87%	83.04%	7.65%	77.84%	**3.98%**	**78.91%**
$\alpha = 0.8$	99.97%	82.85%	12.65%	79.91%	**4.08%**	**80.38%**
$\alpha = 1.0$	100%	83.23%	90.77%	79.56%	**5.83%**	**80.41%**