A Additional Experimental Settings

VGG Network. In this work, we follow [60] to adopt a small VGG network with the following structure. Each convolution layer is followed by a BatchNorm layer and the activation function is ELU with $\alpha = 1.0$. When using our AWM method, we need to substitute the Conv2d with MaskedConv2d.

	Input ($32 \times 32 \times 3$)
Block 1	Conv2d (MaskedConv2d) 3×3, 32 Conv2d (MaskedConv2d) 3×3, 32 Max Pooling 2×2, 32 Dropout (0.3)
Block 2	Conv2d (MaskedConv2d) 3×3, 64 Conv2d (MaskedConv2d) 3×3, 64 Max Pooling 2×2, 64 Dropout (0.4)
Block 3	Conv2d (MaskedConv2d) 3×3, 128 Conv2d (MaskedConv2d) 3×3, 128 Max Pooling 2×2, 128 Dropout (0.4)
	FC(2048) Softmax(Num of Classes)

Table 6: Structure of the small VGG network

Implementation. We adopt PyTorch [39] as the deep learning framework for implementations. In implementation, the outer optimization is conducted with Adam with a learning rate of 0.01 (decay to 0.001 after 50 epochs) and the inner optimization is conducted with SGD with a learning rate of 10. We use the default hyper-parameter setting as $\alpha = 0.9, \beta = 0.1, \gamma = 10^{-8}, \tau = 1000$ for both CIFAR-10 and GTSRB datasets. The batch size for training is summarized in Table 7.

Available Data Size n	One-Shot	100	200	500	5000
Batch Size b	16	32	32	128	128

Table 7: Summary of the Batch Size Settings

Attack Setting. In the single-target attack setting, we set Class 8 as the target for BadNets, Class 2 as the target for Trojan-SQ and Trojan-WM, and Class 0 as the target for l_0-inv, l_2-inv, BLEND, and WaNet. In the multi-target attack setting, we use the pattern of Trojan-SQ and relabel each sample from Class n to $n + 1$.

B Results on GTSRB

Table 8 presents the defense results on the GTSRB dataset. GTSRB dataset has 39209 training data and 12630 test data of 43 classes. Specifically, among the entire GTSRB training data, 1960 images are backdoored. We test with varying size of available data samples ranging from 5000 to 43 (one-shot) for each defense. The remaining samples are used to evaluate the defense result.

The left column depicts five single-target attack methods and one multi-target attack method. The first row represents two different adopted network structure. We present the ACC and ASR under each backdoor removal setting in the table, all attacks are capable of achieving an ASR close to 99% and an ACC around 98% with no defenses.

The performance of the baselines are comparable with AWM when there are sufficient available training data ($n = 5000$): most methods effectively remove the backdoors. Similar to CIFAR-10, IBAU suffers from the biggest performance drop (higher ASR or fail to remove the backdoor). As the number of samples of Class 0 is smaller than Class 2 in GTSRB, it is much easier to remove the

backdoor of l_0-inv and l_2-inv and achieve a very low ASR. However, in other attack settings, we can still observe that ANP is negatively affected by insufficient data. On the left part of Table 8, we can observe that ANP performs worse on the small VGG network, which backup our analysis in the paper. AWM shows state-of-the-art backdoor removal performances overall in this table.

As there are more classes in GTSRB than CIFAR-10, more instances are available in the one-shot setting, we do not use any data augmentation. Our AWM successfully removes all those backdoors while other baselines failed in removing the existing backdoor triggers for certain cases.

Table 8: Backdoor removal performance comparison with various available data sizes on GTSRB dataset with VGG and Resnet-18. Numbers represent percentages. **Bold** numbers indicate the best ACC after backdoor removal and blue color indicates successful backdoor removal.

Attack	Available Data Size n	VGG							Resnet-18						
		Origin	ANP		IBAU		AWM(Ours)		Origin	ANP		IBAU		AWM(Ours)	
			ACC	ASR	ACC	ASR	ACC	ASR		ACC	ASR	ACC	ASR	ACC	ASR
BadNets	5000	ACC	98.06	5.17	**99.06**	0.37	97.32	4.35	ACC	99.02	3.56	99.23	3.47	**99.33**	3.53
	500	98.11	97.35	6.35	97.02	0.32	**98.90**	4.31	98.58	98.47	3.40	**98.65**	3.91	96.50	3.25
	100	ASR	**96.41**	6.84	92.41	59.74	94.58	6.58	ASR	**97.57**	3.40	94.76	3.45	**97.19**	3.78
	One-shot	98.37	95.78	16.53	90.84	83.57	**95.48**	4.54	98.98	**96.79**	2.96	79.98	7.53	**96.91**	3.67
Trojan-SQ	5000	ACC	97.90	7.11	**99.17**	6.66	99.03	6.03	ACC	98.29	11.09	**99.21**	5.83	**99.45**	5.16
	500	98.18	97.49	11.49	96.82	5.92	**98.17**	7.05	98.83	98.55	8.64	**98.81**	5.77	96.58	6.18
	100	ASR	96.94	32.50	84.09	88.57	**95.89**	6.30	ASR	97.23	8.03	96.74	98.39	**97.34**	5.90
	One-shot	99.55	97.21	37.09	83.76	91.02	**93.96**	6.25	99.74	97.62	14.21	69.62	97.80	**96.51**	7.05
Trojan-WM	5000	ACC	98.03	7.20	99.02	0.53	**99.25**	5.70	ACC	98.39	3.66	**99.11**	6.41	**99.15**	4.79
	500	97.90	97.35	6.98	97.80	4.35	**98.52**	5.93	98.75	98.39	9.73	98.49	72.12	**96.57**	5.08
	100	ASR	97.13	19.65	90.37	15.84	**94.38**	5.06	ASR	97.89	9.87	96.38	94.94	**96.86**	7.89
	One-shot	99.82	97.45	25.49	88.65	30.52	**93.74**	5.99	99.65	97.71	46.52	87.27	93.41	**96.15**	6.74
L_0-inv	5000	ACC	98.07	0.48	**99.27**	0.46	98.73	0.35	ACC	98.85	0.64	**99.26**	0.83	**99.45**	0.42
	500	98.35	**98.24**	0.49	96.80	1.36	97.57	1.25	98.64	**98.70**	0.45	98.32	0.52	96.49	0.29
	100	ASR	**97.72**	0.38	84.24	7.14	94.05	0.96	ASR	**97.63**	0.58	96.68	38.09	93.73	0.22
	One-shot	100.0	**97.51**	0.43	80.71	10.63	94.56	0.73	100.0	97.56	0.48	83.66	58.11	93.25	0.32
L_2-inv	5000	ACC	97.79	6.74	**99.13**	0.54	**98.98**	1.81	ACC	98.65	1.27	98.87	0.43	**99.46**	0.46
	500	98.31	97.74	6.53	94.83	0.56	**97.88**	1.59	98.51	**98.72**	1.61	**98.57**	0.43	**98.86**	0.45
	100	ASR	**97.21**	0.46	88.45	7.03	96.36	6.17	ASR	**97.95**	6.26	91.22	0.00	**97.63**	0.44
	One-shot	99.80	**97.21**	0.74	87.42	6.89	96.09	2.37	99.93	97.35	6.67	88.00	42.53	**96.78**	0.61
all-to-all	5000	ACC	97.34	3.53	**98.95**	0.74	**98.68**	1.47	ACC	98.81	2.49	**99.30**	0.18	**99.18**	0.13
	500	98.15	95.70	3.03	96.45	10.74	**98.02**	4.45	98.59	**98.50**	2.32	97.79	4.78	96.22	0.65
	100	ASR	94.34	13.27	90.69	41.61	**95.15**	5.38	ASR	97.19	9.48	94.80	79.63	**89.04**	3.16
	One-shot	93.17	95.76	24.99	75.24	31.72	**93.02**	7.18	96.88	97.61	17.71	88.91	72.42	**87.33**	4.51

Convergence. Figure 4 demonstrates the one-shot training records of ACC and ASR in each epoch of AWM over the 5 single-target attacks. Note that we take ten steps of outer optimization after every 10 steps of inner optimization in each epoch. The backdoors are removed very quickly in most cases. Since we only have extremely insufficient clean data, it causes a little accuracy degradation after a long time of training. We report the averaged ACC and ASR after 100 epochs(1000 iterations) over 5 runs in previous table.

C Additional Studies of AWM

Masking Selected Layers. For the all-to-all attack on GTSRB, we visualize the distribution of mask values in different layers and try to achieve similar performance with masking fewer layers. Figure 5 shows the percentage of mask values falling into each interval. Most values fall in the smallest and the largest intervals, indicating the effect of sparsity constraint.

In the following experiment, we optimize the mask on each layer on CIFAR-10 with VGG. As shown in the Table 6, there are six convolution layers in this small VGG network. We name VGG-i as the VGG with only mask on the i-th convolution layer. We summarize the backdoor removal result with 100 instances in Table 9. Masking shallow convolution layers, such as VGG-1, 2, and 3, is much easier to remove the backdoor comparing with masking deep layers. Another phenomenon is that masking the first convolution layer causes the largest loss on accuracy. This inspires that it might be possible to save the number of masks by limiting them in shallow layers.

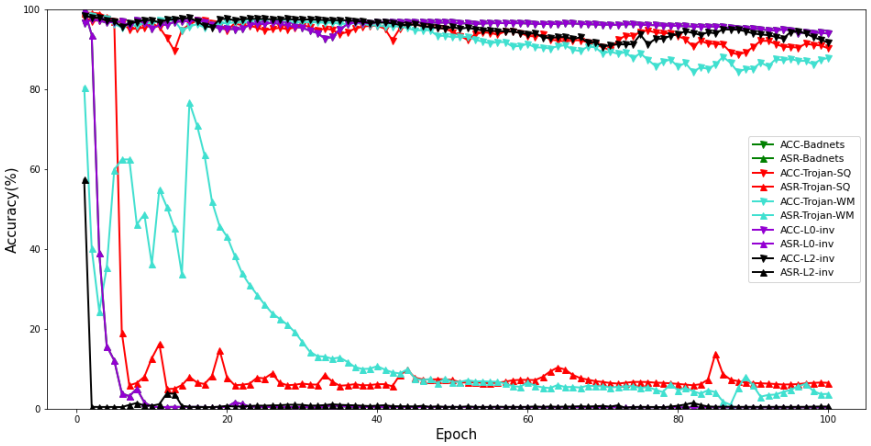

Figure 4: Training records of GTSRB (one-shot).

Figure 5: Distribution (%) of weight mask values of VGG.

Network	Number of Kernels	ACC	ASR
VGG	448	95.15	5.38
VGG-1	32	93.75	3.34
VGG-2	32	96.27	7.25
VGG-3	64	96.54	9.60
VGG-4	64	96.98	23.71
VGG-5	128	97.65	47.83
VGG-6	128	97.37	60.19

Table 9: Performance with Different Layers of Mask

We also visualizes the distribution of weight mask values in Figure 6. Note that this heatmap summarizes the six experiments and each row corresponds to the mask values in the specific layer. Figure 6 is similar to Figure 5: Overall, it shows that optimizing masks on a certain layer is dependent to other layers to some degree. Therefore, it is reasonable to consider separating or selectively optimizing masks on some layers, which we leave as future work.

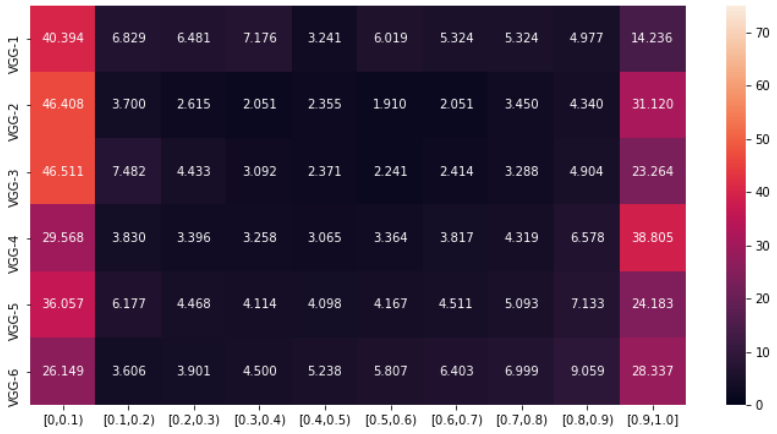

Figure 6: Distribution (%) of weight mask values via masking different positions.

Defend Against Multiple Triggers. It is interesting to explore how to break a backdoor defense method with knowledge of how it works. Some principles in developing backdoor attacks, such as making triggers invisible or natural, are data-driven and do not direct a specific backdoor removal optimization procedure. The corresponding adaptive attack can potentially be complex on the device since the current backdoor removal techniques, including our AWM, have already involved complicated bi-level optimization.

As an attempt, we design a simple method targeting our backdoor trigger reconstruction mechanism: we only estimate one universal trigger Δ in every iteration. Now suppose the attacker knows our design and decides to inject multiple backdoor patterns into the model. They would hope our design could only remove one of them and thus fail on the other triggers.

The following table summarizes the result of defending against multiple backdoor triggers on CIFAR-10. We first train a poisoned model using three triggers: BadNets, Trojan-WM, and L_2-inv. The overall poison rate is set as 5%. Then we apply our AWM to remove the backdoors. ASR (all) represents the attack success rate as long as any one of the three triggers fooled the model. ASR1, ASR2, and ASR3 represent the attack success rate of each of the three triggers. Finally, ACC is the test accuracy on the clean test set. The results show that adding multiple triggers still cannot penetrate our AWM defense even with limited resources.

ShuffleNet	ASR(all)	ASR1	ASR2	ASR3	ACC
Original	99.83	95.80	99.52	99.15	84.86
AWM(500 images)	10.38	9.54	8.19	13.16	77.02
AWM(one-shot)	8.47	13.54	10.34	16.21	71.83

Table 10: Defend against multiple backdoors.

We conjecture that this is because although multiple triggers are involved, our algorithm will still try to identify the most likely triggers in each iteration. Thus when the first (the most prominent) 'trigger' is removed, the algorithm will automatically target the next likely 'trigger'.

Such backdoor removal strategies seem to be hard to penetrate. Successful attacks may need to leverage tri-level optimization problems, which are notoriously hard to solve, or aim to make the removal strategy impractical by lowering its natural accuracy, which currently has no concrete solutions. We also leave this problem as one of our future work directions.

D Objectives in Ablation Study

We list the formal objective functions for the five modifications of AWM compared with in Section 5.3 of our paper.

0) Full AWM

$$\min_{\mathbf{m}\in[0,1]^d} \mathbb{E}_{(\mathbf{x},y)\sim D}\, \alpha\mathcal{L}(f(\mathbf{x};\mathbf{m}\odot\boldsymbol{\theta}),y) + \beta \max_{\|\boldsymbol{\Delta}\|_1\leqslant\tau} [\mathcal{L}(f(\mathbf{x}+\boldsymbol{\Delta};\mathbf{m}\odot\boldsymbol{\theta}),y)] + \gamma\|\mathbf{m}\|_1, \quad \text{(D.1)}$$

1) *No Clip*: AWM with no $\boldsymbol{\Delta}$ clipping:

$$\min_{\mathbf{m}\in[0,1]^d} \mathbb{E}_{(\mathbf{x},y)\sim D}\, \alpha\mathcal{L}(f(\mathbf{x};\mathbf{m}\odot\boldsymbol{\theta}),y) + \beta \max [\mathcal{L}(f(\mathbf{x}+\boldsymbol{\Delta};\mathbf{m}\odot\boldsymbol{\theta}),y)] + \gamma\|\mathbf{m}\|_1, \quad \text{(D.2)}$$

where β is set to be $1-\alpha$.

2) *No Shrink*: AWM with no L_1 regularization on $\mathbf{m}$;

$$\min_{\mathbf{m}\in[0,1]^d} \mathbb{E}_{(\mathbf{x},y)\sim D}\, \alpha\mathcal{L}(f(\mathbf{x};\mathbf{m}\odot\boldsymbol{\theta}),y) + \beta \max_{\|\boldsymbol{\Delta}\|_1\leqslant\tau} [\mathcal{L}(f(\mathbf{x}+\boldsymbol{\Delta};\mathbf{m}\odot\boldsymbol{\theta}),y)]\,. \quad \text{(D.3)}$$

3) *NC-NS*: AWM with no $\boldsymbol{\Delta}$ clipping and $\mathbf{m}$ regularization;

$$\min_{\mathbf{m}\in[0,1]^d} \mathbb{E}_{(\mathbf{x},y)\sim D}\, \alpha\mathcal{L}(f(\mathbf{x};\mathbf{m}\odot\boldsymbol{\theta}),y) + \beta \max [\mathcal{L}(f(\mathbf{x}+\boldsymbol{\Delta};\mathbf{m}\odot\boldsymbol{\theta}),y)]\,. \quad \text{(D.4)}$$

4) L_2 *Reg*: AWM with $\boldsymbol{\Delta}$'s L_2 regularization;

$$\min_{\mathbf{m}\in[0,1]^d} \mathbb{E}_{(\mathbf{x},y)\sim D}\, \alpha\mathcal{L}(f(\mathbf{x};\mathbf{m}\odot\boldsymbol{\theta}),y) + \beta \max_{\|\boldsymbol{\Delta}\|_2\leqslant\tau} [\mathcal{L}(f(\mathbf{x}+\boldsymbol{\Delta};\mathbf{m}\odot\boldsymbol{\theta}),y)] + \gamma\|\mathbf{m}\|_1. \quad \text{(D.5)}$$

5) L_2 *Reg NC*: AWM with $\boldsymbol{\Delta}$'s L_2 regularization and no clipping;

$$\min_{\mathbf{m}\in[0,1]^d} \mathbb{E}_{(\mathbf{x},y)\sim D}\, \alpha\mathcal{L}(f(\mathbf{x};\mathbf{m}\odot\boldsymbol{\theta}),y) + \beta \max [\mathcal{L}(f(\mathbf{x}+\boldsymbol{\Delta};\mathbf{m}\odot\boldsymbol{\theta}),y) + \|\boldsymbol{\Delta}\|_2] + \gamma\|\mathbf{m}\|_1. \quad \text{(D.6)}$$