A Motivation of Using Randomness

A.1 Motivation 1

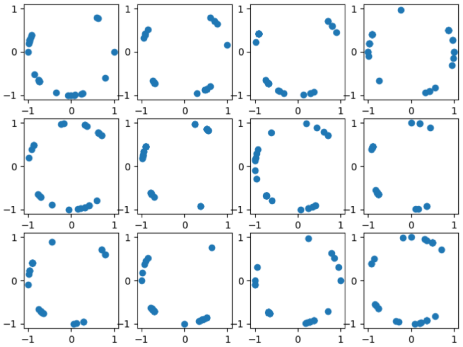


Figure 7: Distribution of adversarial examples for a group of model individuals of the same architecture. Each grid shows the relative positions of adversarial examples to the clean image (the origin) on two randomly selected input dimensions. Here, the distances of adversarial examples on the given x-y plane are normalized. Adversarial examples are generated by CW-PGD attack.

A.2 Motivation 2

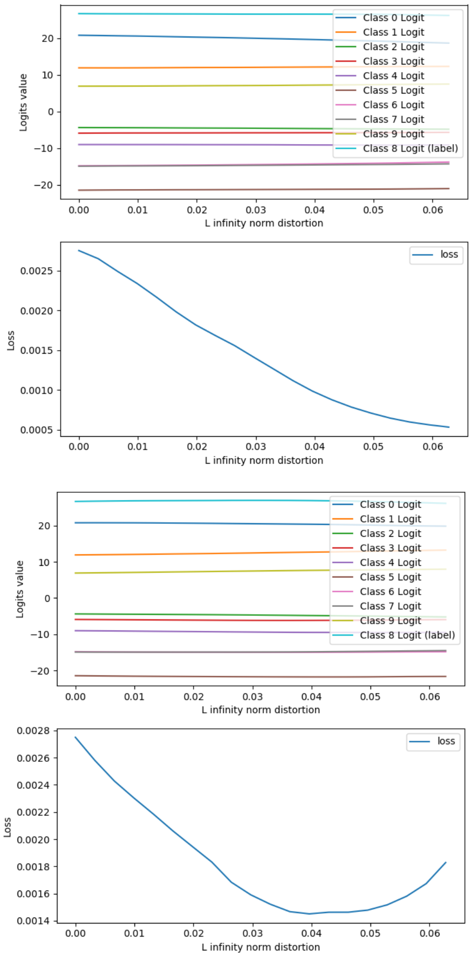


Figure 8: Examples of logits and loss changes in random directions.

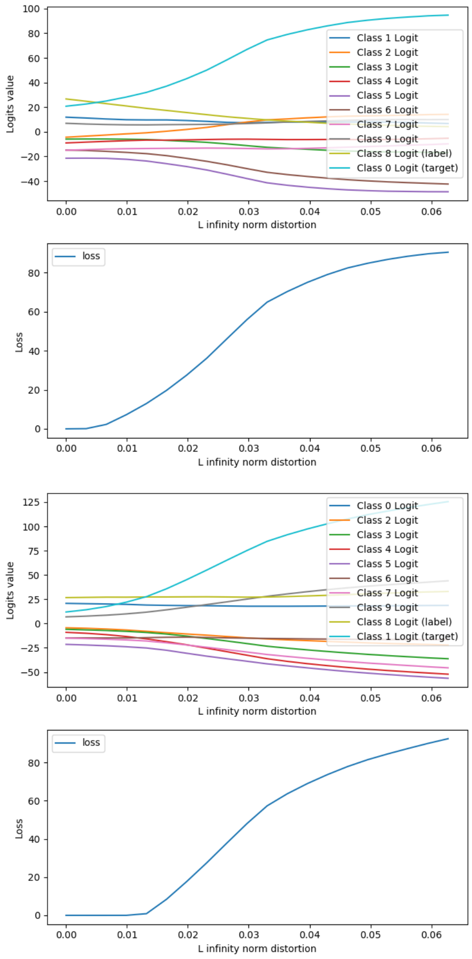


Figure 9: Examples of logits and loss changes in (targeted) adversarial directions. Adversarial directions are found by performing CW-PGD attack on the clean image (origin).

A.3 Input Gradient Deviation v.s. Mean Defense Rate

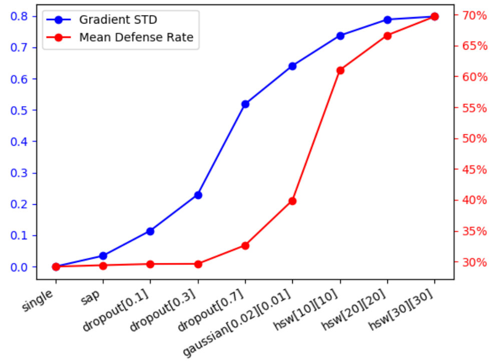


Figure 10: Input gradient variance is highly correlated with defense effectiveness for stochastic defenses.

B Training for HRS

Algorithm 1 Training HRS-protected Model

```
Require:
    HRS model architecture with $M$ switching blocks. $N^i$ denotes the number channels in block $i$ and $c_i^j$ denotes the $j$'s channel in block $i$.
for block $i \in [1, M]$ do
    for channel $j \in [1, N_i]$ do
        Construct a HRS Model with $M$ blocks:
        for block $k \in [1, M]$ do
            if $k < i$ then
                Construct $N_i$ channels with trained channels $\{c_k^l \mid l \in [1, N_k]\}$;
                Freeze channels in block $k$;
            else
                Construct 1 channel for block $k$ with randomly initialized weights;
                Set all channels in block $k$ to be trainable;
            end if
            Train all trainable channels to convergence;
            Save trained channel $c_i^j$;
        end for
    end for
end for
return A HRS Model with trained channels $\{c_i^j \mid i \in [1, M], j \in [1, N_i]\}$
```

C Experiment Details

C.1 Attack Details

All attacks are implemented in a white-box, targeted attack setting for a fair comparison. For reproducibility of the experiments, we summarize the hyper-parameters we used for each attack.

- **FGSM:** No hyper-parameter needs to specify. Different attack strengths are given by varying the step size ϵ.
- **CW:** We run CW attack with L_2 distortion metric. We run gradient descent for 100 iterations with step size of 0.1 and use 10 rounds binary search finding the optimal weight factor c. Different attack strengths are given by varying the confidence factor κ.
- **PGD**: We run gradient descent for 100 iterations with step size of 0.1. Different attack strengths are given by varying maximum allowed L_∞ perturbation ϵ.
- **CW-PGD**: The same setting as PGD.

C.2 Defense Implementation of Gaussian Noise

On MNIST, we use the recommended standard deviations which are 0.2 for the ”init-noise” (noise before the input layer) and 0.1 for the ”inner-noise” (noise before other conv layers). However, on CIFAR-10 we found this setting decreases test accuracy significantly (reducing to 60%), thus we use 10× smaller deviations (0.02 and 0.01 respectively). We also found that using Gaussian noise solely is not sufficient to prevent the model from over-fitting. As a solution we also use dropout during training in order to prevent over-fitting.

C.3 Pilot Research on EOT

We run a pilot test to determine the value of n for EOT attacks. We find using $n = 10$ is enough as the benefits of using a larger n saturates when $n > 10$. An example of using different n values for EOT is given in 11. Here we also plot Gaussian defense with a 3× larger noise deviation which is not in 6.2 as it drops the test accuracy to 76.83%. The purpose is to show that the defenses that seem to be less sensitive to EOT (as shown by the lines on the top of Figure 11) do not indicate they are truly resistant to EOT.

C.4 Model Architecture

Table 3: Base model architectures for MNIST and CIFAR-10 Datasets.

	Model for MNIST	Model for CIFAR-10
Conv layer	32 filters with size (3,3)	64 filters with size (3,3)
Conv layer	32 filters with size (3,3)	64 filters with size (3,3)
Pooling layer	pool size (2,2)	pool size (2,2)
Conv layer	64 filters with size (3,3)	128 filters with size (3,3)
Conv layer	64 filters with size (3,3)	128 filters with size (3,3)
Pooling layer	pool size (2,2)	pool size (2,2)
Fully connected	200 units	256 units
Fully connected	200 units	256 units
Output layer	10 units	10 units

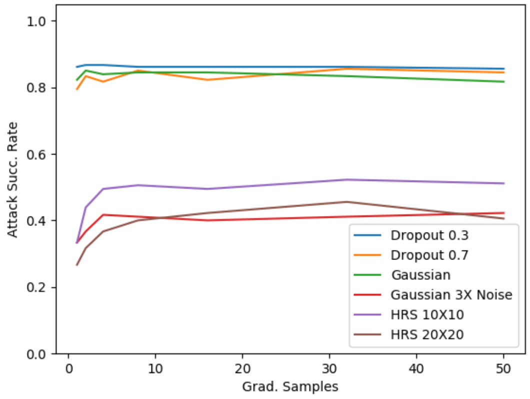


Figure 11: Pilot Research on EOT.

C.5 Implementation Details on Study of Defense Efficiency

Spots of HRS are due to different number of block channels ranging from 5×5 to 30×30. Spots of dropout are due to different training and testing dropout rate ranging from 0.1 to 0.9. Spots of Gaussian noise are due to different initial and inner Gaussian noise deviations ranging from (0.01, 0.005) to (0.11, 0.055) on CIFAR-10 and from (0.1, 0.05) to (0.325, 0.1625) respectively. Spots of Adversarial Training are due to different ϵ bounds from 0.5/255 to 4.5/255 of adversarial examples used in training.

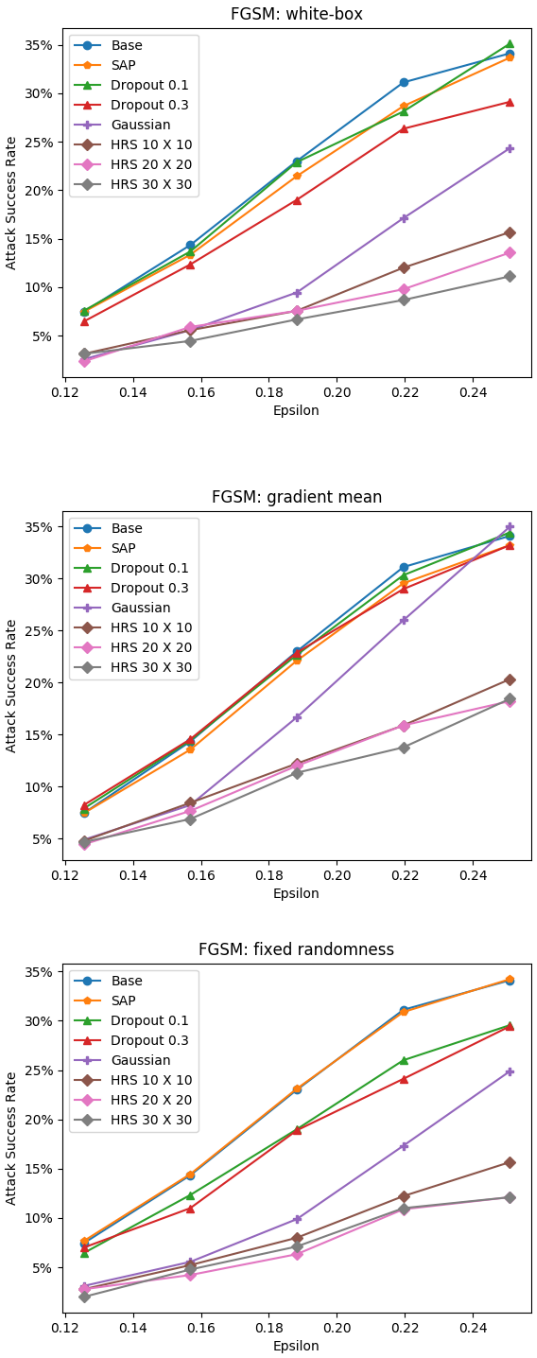
C.6 Test Accuracy

Table 4: Test accuracy of different defense methods.

Model	MNIST	Dev.(e-4)	CIFAR	Dev.(e-4)
Base	99.04%	/	79.17 %	/
SAP	99.02%	1.47	79.16 %	2.81
Dropout 0.1	98.98%	3.45	79.08%	7.99
Dropout 0.3	98.68%	6.06	78.65 %	16.67
Dropout 0.7	/	/	76.02 %	24.52
Gaussian	99.02%	5.82	78.04 %	7.25
HRS 10*10	98.95%	5.33	78.93%	26.81
HRS 20*20	98.91%	6.02	78.76%	20.32
HRS 30*30	98.85%	8.31	78.69%	23.25

D Experimental Results

D.1 MNIST

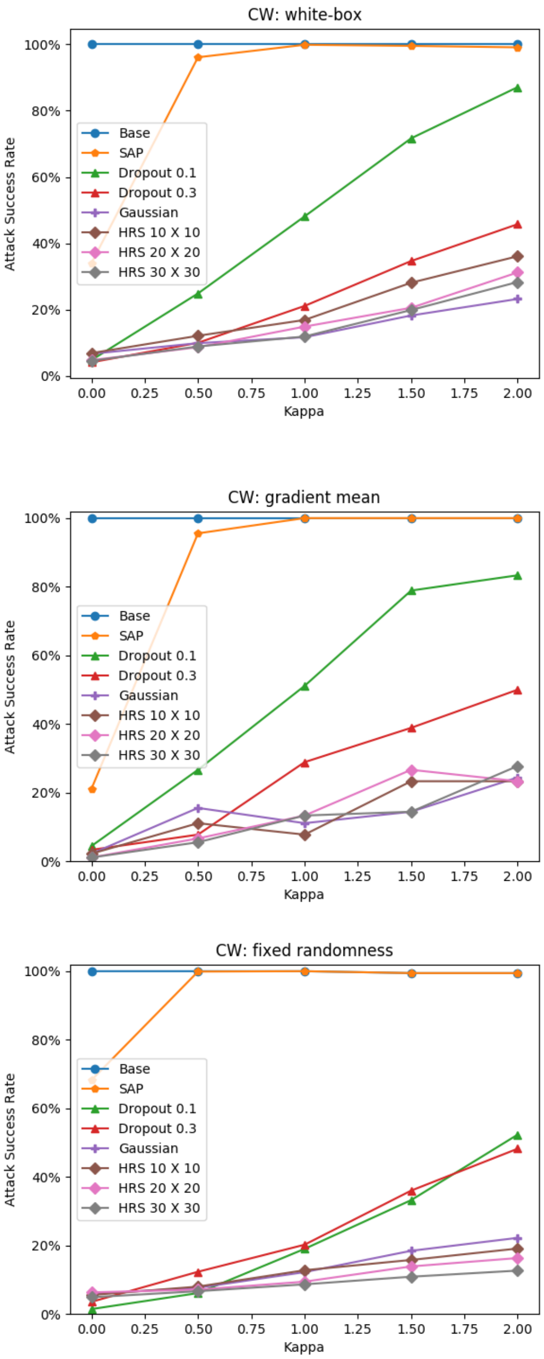


FGSM: white-box

FGSM: gradient mean

FGSM: fixed randomness

Figure 12: Attack success rate of FGSM on different defenses.



CW: white-box

CW: gradient mean

CW: fixed randomness

Figure 13: Attack success rate of CW on different defenses.

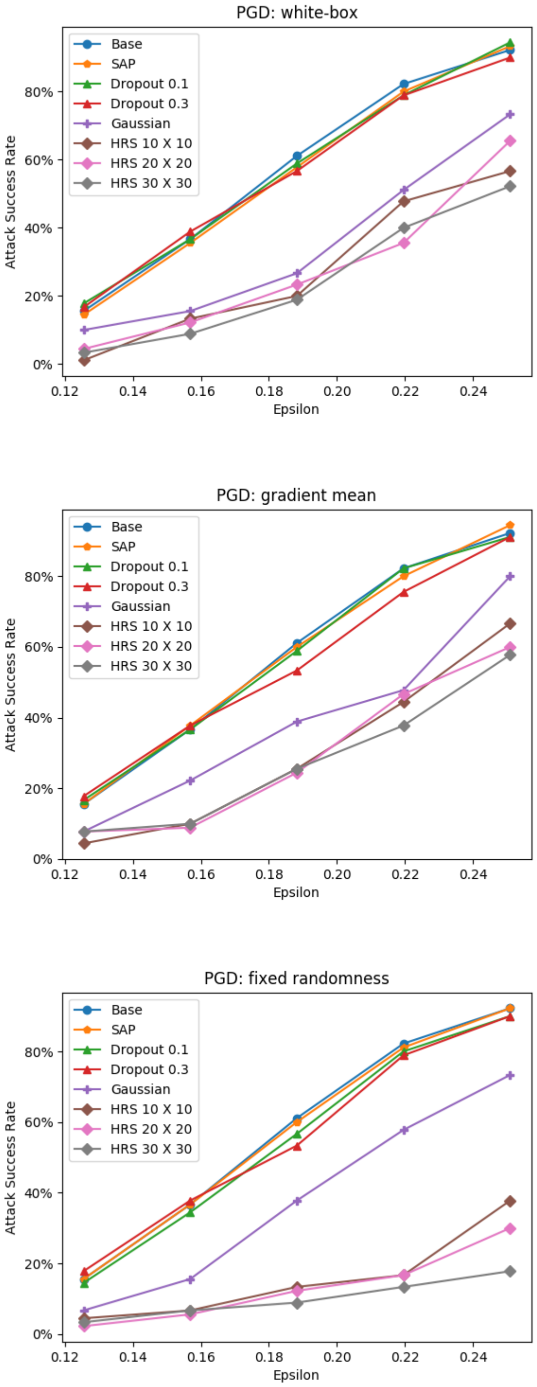


Figure 14: Attack success rate of PGD on different defenses.

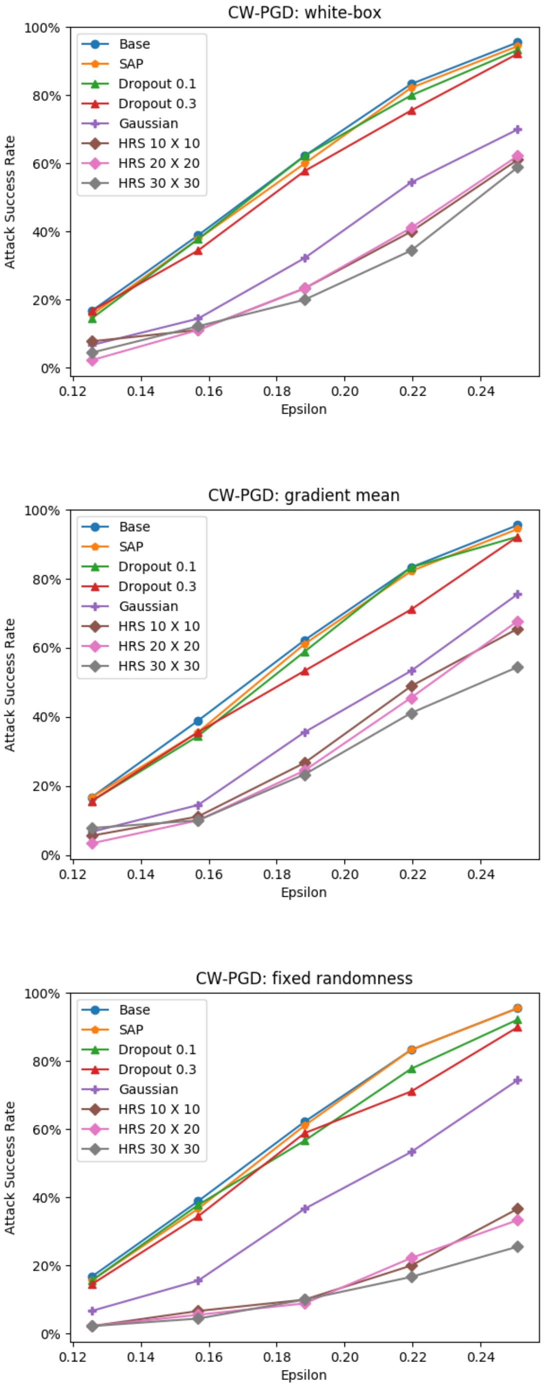


Figure 15: Attack success rate of CW-PGD on different defenses.

D.2 CIFAR-10

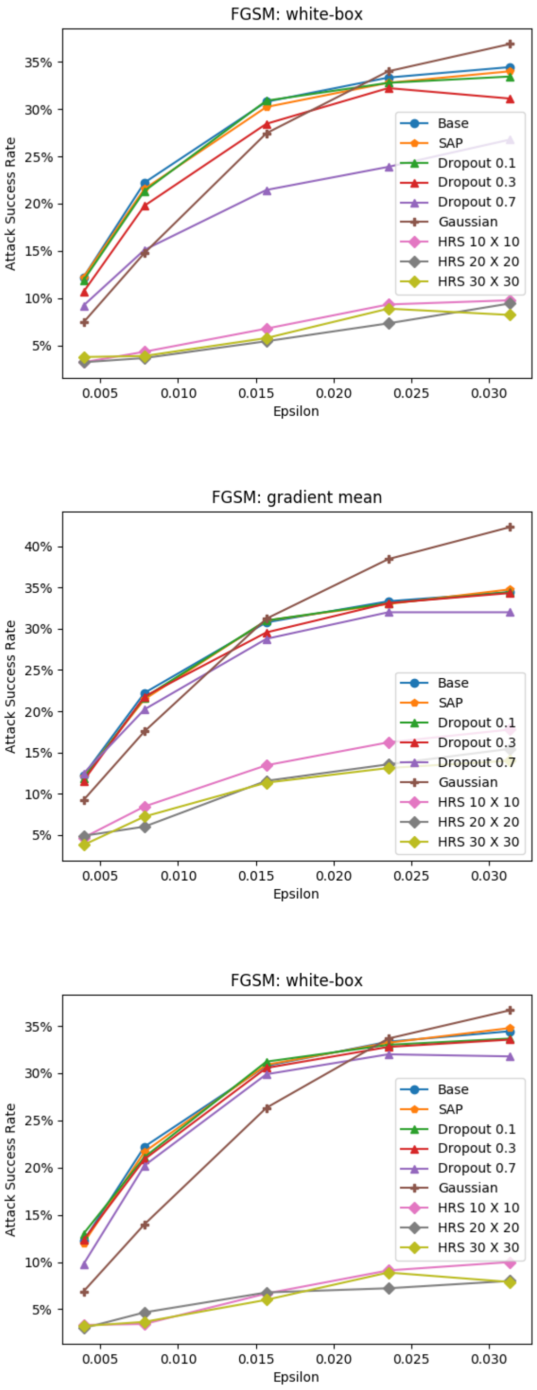


Figure 16: Attack success rate of FGSM on different defenses.

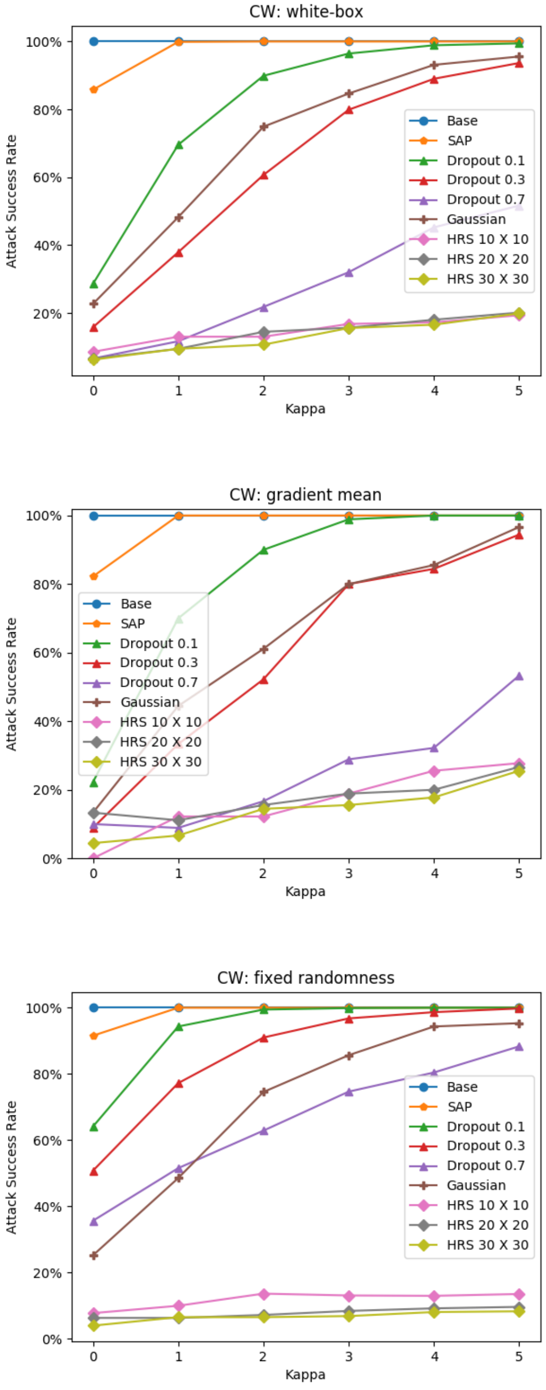


Figure 17: Attack success rate of CW on different defenses.

E Adversarial Reprogramming

The number of parameters in the input transformation is crucial factor to achieve high reprogramming accuracy. We applied the original input transformation in [Elsayed *et al.*, 2018] to our CIFAR to MNIST reprogramming task but find the reprogramming performance is poor due to the lack of parameters in input transformation. In order to differentiate defense effectiveness under a strong reprogramming setting, we use a local-connected layer as input transformation. The advantage of local-connected layer is that we can easily control the number of parameters by setting different kernel sizes. We found that using a 3×3 kernel lead to 95.07% accuracy which is similar to reported accuracy in [Elsayed *et al.*, 2018]. We show experiment results using other kernel sizes in the following and it is clear that using a larger kernel (a large number of parameters) will lead to higher reprogramming accuracy. However, under all experiment settings we found our proposed defense demonstrate much stronger defense (lower reprogramming accuracy) compared to other defenses.

F Supplementary Experiments

F.1 The Effect of Increasing Switching Blocks

In Table 5 we compare HRS models with 1, 2 and 3 switching blocks. For all models in the comparison, there are 5 channels in each block. Thus the parameter size of these 3 models are the same. It is noted that by increasing the number switching blocks, the resistance against adversarial attacks can be improved. So the benefit of using more switching blocks is increasing model variation given certain parameter size, and thus improving the defending effectiveness. Yet the improvement is traded with more test accuracy drop. The number of blocks of a HRS model can be treated as a defense strength controlling factor.

Table 5: Defense Effectiveness (in terms of ASR) of HRS with different number of blocks. The attack here is CW-PGD with different ϵ bounds.

Strength	1/255	2/255	4/255	6/255	8/255
1-block	5.6%	16.4%	46.7%	76.7%	85.5%
2-block	3.3%	10.0%	35.6%	64.3%	81.2%
3-block	2.9%	13.1%	33.7%	55.2%	65.9%