A DCT AND IDCT

The discrete cosine transform (DCT) is a mathematical tool that can be employed to shift an image from the spatial domain to the frequency domain. By initiating a search from lower frequencies and progressing to higher ones, one can effectively pinpoint an adversarial sample, thereby reducing the number of required queries. DCT represents an image as a sum of sinusoids of varying magnitudes and frequencies. Specifically, for an input image $X \in \mathbb{R}^{d\times d}$, the DCT transform $V = DCT(X)$ is:

$$V_{m,n} = \alpha_m \alpha_n \sum_{i=0}^{d-1}\sum_{j=0}^{d-1} X_{i,j} \cos\frac{\pi(2i+1)m}{2d} \cos\frac{\pi(2j+1)n}{2d} \tag{7}$$

where

$$\alpha_m = \begin{cases} \sqrt{\frac{1}{d}}, & m = 0 \\ \sqrt{\frac{2}{d}}, & 1 \le m \le d-1 \end{cases} \tag{8}$$

and

$$\alpha_n = \begin{cases} \sqrt{\frac{1}{d}}, & n = 0 \\ \sqrt{\frac{2}{d}}, & 1 \le n \le d-1 \end{cases} \tag{9}$$

for $0 \le m, n \le d-1$.

The values $V_{m,n}$ are called the DCT coefficients. The DCT is an invertible transform, and its inverse IDCT is given by:

$$X_{i,j} = \sum_{m=0}^{d-1}\sum_{n=0}^{d-1} \alpha_m \alpha_n V_{m,n} \cos\frac{\pi(2i+1)m}{2d} \cos\frac{\pi(2j+1)n}{2d} \tag{10}$$

for $0 \le i, j \le d-1$. The basis functions are:

$$\alpha_m \alpha_n \cos\frac{\pi(2i+1)m}{2d} \cos\frac{\pi(2j+1)n}{2d} \tag{11}$$

The inverse DCT (IDCT) is an inverse phase of DCT. The IDCT equation can be interpreted as meaning that any $d \times d$ image can be written as a sum of basis functions. The DCT coefficients $V_{m,n}$ can be regarded as the weights applied to each basis function, with lower frequencies represented by lower m, n. Especially for 8×8 images, the 64 basis functions are illustrated by Figure 6. Horizontal

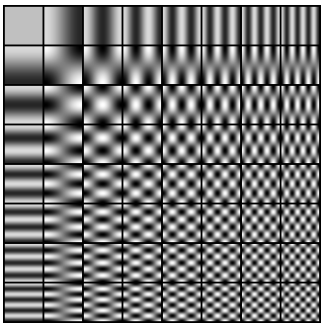

Figure 6: The 64 Basis Functions of an 8-by-8 Image.

frequencies increase from left to right, and vertical frequencies increase from top to bottom.

B AVERAGE INFERENCE ATTACK

In this section, we will examine a different attack: an adversary may send the same adversarial input multiple times and average their outputs to bypass the RPNet's defense that adds Gaussian noise with zero means. We call this attack as *average inference attack*. This type of attack is designed to exploit the fact that RPNet adds noise to the output probability distribution in each query, which has a zero mean and could potentially be averaged out by repeated queries.

Table 4: The attack performance under different query numbers of Average PNet Attack.

N_q	CIFAR-10						Diabetic Retinopathy					
	Clean Accuracy		Average Queries		Defense Success Rate		Clean Accuracy		Average Queries		Defense Success Rate	
	Untar.	Target	Untar.	Target	Untar.	Target	Untar.	Target	Untar.	Target	Untar.	Target
1	74.55%		199.4	299.7	63.36%	91.88%	68.09%		48.9	52.0	66.41%	88.67%
2	74.55%		200	302	62.19%	86.99%	68.09%		49.5	51.5	**58.53%**	**71.85%**
3	74.55%		199.6	299.3	59.08%	**83.52%**	68.09%		48.3	49.3	60.88%	79.01%
4	74.55%		200	300	**54.76%**	87.31%	68.09%		50.5	49	69.09%	85.96%
5	74.55%		200.4	299.8	61.93%	91.98%	68.09%		50.2	48.4	71.67%	89.44%

Firstly, we investigate whether sending the same adversarial input multiple times and averaging their outputs can decrease the Defense Success Rate (DSR) of RPNet. By analyzing the results of this investigation in Table 4, we can gain a better understanding of RPNet's robustness against this type of attack and identify potential countermeasures to further improve its defense effectiveness.

In our experiments, the number of queries for identical adversarial inputs (N_q) varied from 1 to 5. Table 4 reveals that for the CIFAR10 dataset, DSR decreases from 91.88% to 86.99% when N_q is set to 2, and further decreases to 83.52% when N_q is set to 3. However, when N_q is increased to 5, the DSR increases to 91.98%.

This trend can be explained by the fact that when N_q is increased, the attacker needs to query N_q times to add noise once, which means that the total number of queries available for adding noise decreases as N_q increases. Therefore, with a fixed number of total queries, the attacker will add less noise when N_q is larger. For example, under a fixed total number of queries, e.g., 300, an attacker can add noise up to 300 times when $N_q = 1$ but only up to 60 times when $N_q = 5$.

C Average Inference Attack Defense

In this section, we propose two defense methods to counteract the average inference attack. The first method suggests using a Gaussian distribution with a non-zero mean (μ) instead of a standard Gaussian distribution. The second method recommends adding noise at the penultimate layer rather than at the last layer. To assess their effectiveness, we measured RPNet's DSR when (μ) was set to 3 and when noise was added at the penultimate layer. We determined that N_q should be set to values that result in lower DSRs based on Table 3: $N_q = 4$ for untargeted attacks against CIFAR10, $N_q = 3$ for targeted attacks on CIFAR10, and $N_q = 2$ for both untargeted and targeted attacks on Diabetic Retinopathy.

Based on the results presented in Table 5, we can observe that using a non-zero mean Gaussian distribution renders the Average PNet Attack ineffective while maintaining a high Clean Accuracy (ACC) similar to that of using a zero mean Gaussian distribution. For instance, on the CIFAR10 dataset, the DSR of targeted attacks increased from 83.52% to 90.47%, and the DSR of untargeted attacks increased from 54.76% to 64.19%, while the ACC remained at 74.53%. Similarly, adding noise at the penultimate layer instead of the last layer also improved DSR. For example, on CIFAR10, the DSR of targeted attacks increased from 83.52% to 85.77%, and the DSR of untargeted attacks increased from 54.76% to 59.92%.

Table 5: The defense performance against Average PNet Attack using two proposed methods.

Schemes	CIFAR-10						Diabetic Retinopathy					
	Clean Accuracy		Average Queries		Defense Success Rate		Clean Accuracy		Average Queries		Defense Success Rate	
	Untar.	Target	Untar.	Target	Untar.	Target	Untar.	Target	Untar.	Target	Untar.	Target
RPNet	74.55%		200	299.3	54.76%	83.52%	68.09%		49.5	51.5	58.53%	71.85%
$\mu = 3$	74.53%		200	300	64.19%	90.74%	68.16%		50.5	50.5	64.96%	86.75%
Penultimate layer	70.83%		201	300.6	59.92%	85.77%	66.89%		50.5	49	61.56%	77.30%

D IMPLEMENTATION DETAILS

Methodologies Study. For attack methods, we compare prior works including SimBA-DCT (Guo et al., 2019) and Square attack (Andriushchenko et al., 2020) with our PNet-Attack without a schedule and with a schedule in Figure 3(a)(b) and Table 2 in the main paper. For defense methods, we compare prior works RND, its variant RND-GF (Qin et al., 2021) and Adversarial Training (Goodfellow et al., 2014) with our techniques including RPNet, RPNet with input noise, and RPNet-DNT. RPNet simply adds noise in the output layer shown in Equation 3 of main paper. RPNet+Input noise means adding the noise with a different scaling factor in the input layer. RPNet-DNT further incorporates the DNT technique.

Parameter Settings. We set the maximum number of queries for a single evaluated image as 300/100 for the targeted/untargeted attacks, respectively. For Adversarial Training(AT), we add adversarial examples that are generated by SimBA-DCT, and the ratio of adversarial examples is 10% of the entire training data. For PNet-Attack, the cycle of schedule T is 400, ϵ is 1, and λ_{min} is 0.5, λ_{max} is 1.5. For RPNet, we set σ as 0.1. The scaling factor of input noise is set as 0.05. For the defense method, the σ_{max} is set as 0.25. More experimental settings are included in Appendix. The results of the MNIST are shown in Appendix.

Cryptosystems Settings. For cryptosystems of PNet, one can follow the LoLa (Brutzkus et al., 2019), where the BFV scheme in SEAL (SEAL) is used. For MNIST and CIFAR-10, the plaintext modulus $m = 2148728833 \times 2148794369 \times 2149810177$, modulus degree $N = 16384$, coefficient modulus $q =\sim 440$ bits. The security level is larger than 128 bits which are verified by $lwe\_estimator$ (Albrecht et al., 2015). To run PNets, one can run all experiments on the same Azure standard B8ms virtual machine with 8 vCPUs and 32GB DRAM.

E FURTHER EXPERIMENTS

E.1 RESULTS ON MNIST DATASET

In Figure 7, we compare different defense methods on MNIST. We show that compared with the traditional methods, the DSR of RPNet proposed in our paper is greatly improved.

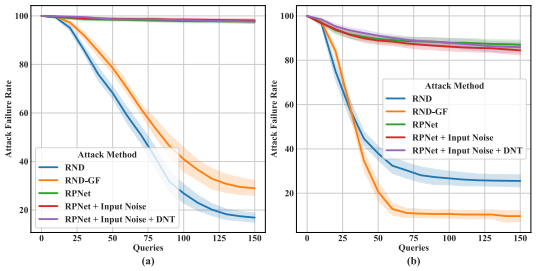

Figure 7: Defense results on MNIST.

E.2 RESULTS ON LARGER DATASETS

In investigating the performance of proposed methods on larger datasets, we conducted experiments on CIFAR-100 and ImageNet datasets. The results in Table 8 illustrate that PNet Attack and RPNet Defense work well on larger datasets. For ImageNet, we downsample each original image to $3 \times 64 \times 64$.

Table 6: PNet-Attack and RPNet defense on larger datasets.

Datasets	PNet-Attack			RPNet Defense		
	CACC(%)	Avg.Queries	ASR(%)	CACC(%)	Avg.Queries	ASR(%)
CIFAR-100	43.08	901.6	92.40	41.25	901.5	18.13
ImageNet	12.35	2300.4	79.13	11.18	2303.3	14.64

E.3 DEFENSE PERFORMANCE AGAINST OTHER ATTACKS

In Table 7, we deploy RPNet against another two commonly used attacks, i.e., SimBA-DCT (Guo et al., 2019) and Square Attack (Andriushchenko et al., 2020). The results show that RPNet can achieve effective defense on different attacks and it outperforms prior defenses such as RND-GF because RPNet is specifically optimized for private inference.

Table 7: RPNet defense on SimBA-DCT and Square Attack.

Attacks	SimBA-DCT			Square Attack		
	CACC(%)	Avg.Queries	ASR(%)	CACC(%)	Avg.Queries	ASR(%)
No Defense	77.3	302.4	74.0	77.3	301.6	71.6
RND-GF	71.6	299.4	44.9	71.6	300.2	38.2
RPNet + Input + DNT	74.6	299.1	5.9	74.6	303.5	14.2

E.4 COMPUTATION OVERHEAD OF PNET ATTACK AND RPNET DEFENSE

To evaluate overhead of the deployment of proposed attack and defense methods, we conducted experiments, and the results in Table 8 show the performance of our attack and defense on different datasets, including larger datasets and high-dimensional models. In particular, for PNet-Attack and our RPNet, average query number, ASR, private inference attacking time, and communication cost are listed, respectively. Our defense RPNet significantly enlarges the attack overhead, such as average query number, private inference latency, and communication cost, while achieving a similar ASR. For example, on the CIFAR-10 dataset, to achieve $> 60\%$ ASR, the attacking time is enlarged to $1,851$ seconds from 55.62 seconds after our defense RPNet.

Table 8: Attack overhead and defense effect on private inference. Average query number, ASR, attacking time, and communication cost are included. For ImageNet, we firstly downsample the $3 \times 224 \times 224$ images to $3 \times 64 \times 64$ then use a network that consists of three convolution blocks and two fully connected layers.

Datasets	PNet-Attack				With RPNet Defense			
	Avg.Queries	ASR(%)	Time	Communication Cost	Avg.Queries	ASR(%)	Time	Communication Cost
Diabetic Retinopathy	47.7	62.81	60.69 s	0.93 MB	802.3	63.26	1021 s	15.6 MB
CIFAR-10	42.1	66.14	55.62 s	0.85 MB	1401.2	62.07	1851 s	28.3 MB
MNIST	50.1	85.67	12.04 s	0.34 MB	348.5	84.71	83.75 s	2.37 MB
ImageNet	2300.4	79.13	9623 s	88.4 MB	9547.2	37.64	39242 s	426 MB