Harmonizing Feature Maps: A Graph Convolutional Approach for Enhancing Adversarial Robustness

31520231154326 Kejia Zhang*, 31520231154311 Siyi Sun, 31520231154273 Ziyu Xu

The Department of Artificial Intelligence, Xiamen University, Xiamen, China

*kejiazhang@stu.xmu.edu.cn

Abstract

During the inference of deep neural networks, adversarial attacks subtly manipulate feature maps, introducing 'noise' that disrupts the process and undermines the model's robustness. To address this issue, we observe that low-frequency components in adversarially perturbed feature maps encapsulate more essential information compared to their high-frequency counterparts. Therefore, we propose a novel feature denoising strategy that leverages Graph Convolutional Networks (GCNs) and low-pass filters to address this issue. Different from traditional denoising techniques, our method can identify and exploit correlations within and among the feature maps. The discerned similarities are harnessed to reconstruct the graph, which is then utilized for feature enhancement in a denoising block based on a Graph Convolutional Network (GCN). Our approach is compatible with various neural network architectures, thereby offering a versatile solution to combat adversarial noise in feature maps. Experimental results on the CIFAR-10 dataset validate that our method can improve the robustness against different adversarial attacks.

Introduction

Deep neural networks have significantly improved the performance of computer vision tasks, notably image classification and object detection [11, 32]. This advancement has unlocked a multitude of applications in various industries. Despite their effectiveness, the reliability of such models is frequently compromised by adversarial attacks, such as the Fast Gradient Sign Method (FGSM) [8], Projected Gradient Descent (PGD) [16], and Autoattack [3]. These techniques subtly perturb the input data, thereby introducing significant "noise" into the neural networks' feature maps [31, 26]. Although these changes are generally imperceptible to humans, they can significantly disrupt the model's inference process, posing a substantial challenge to its robustness.

Recently, various strategies, such as adversarial training [1, 19, 12] and model correction [7, 41], have been proposed to enhance model resilience against adversarial attacks. However, existing methods primarily focus on perturbations in input data, largely neglecting the model's internal features. The network's inference process significantly transforms these internal features when under attack. To exploit the influence of internal feature correlation, we leverage the Fourier transform to reconstruct and filter feature maps during the compromised inference process. Our experimental results show that high-frequency components within the feature maps are detrimental to performance, indicating the presence of noise and redundant information. In contrast, low-frequency components preserve beneficial information, enhancing the model's robustness and performance. This insight motivates us to perform the denoising at the feature-level, removing high-frequency noise and redundant data, thereby improving the model's robustness and generalization capabilities.

Although the filtering of high-frequency features from feature maps can reduce redundant information and noise, this method often overlooks the inherent spatial relationships within feature maps. In Convolutional Neural Networks (CNNs), the practice of sharing convolutional kernels frequently leads to different feature maps extracting analogous features from similar positions and orientations [39]. This phenomenon is particularly evident in deep CNNs, where the spatial relationships between feature maps can offer critical insights into their similarities [24, 34].

To leverage these insights, we introduce a novel strategy that utilizes the similarities between feature maps to construct a graph representing their relationships and interactions. We employ a similarity metric to discern correlations among feature maps, establishing the basis of our graph structure. Subsequently, we perform graph convolution operations possessing low-pass filtering characteristics to denoise the features. This approach signifies a unique method of incorporating feature information and understanding correlations between feature maps during neural network inference, a perspective often neglected by prior methods. Our initial results demonstrate our model's heightened resilience against various adversarial attacks. Applicable to different CNN-based architectures, our method offers a potent and flexible solution for enhancing model robustness against adversarial noise in feature maps.

To our knowledge, this paper represents a pioneering effort in exploiting the correlations between feature maps to integrate Graph Neural Networks (GNNs) into visual tasks. This strategy provides a unique perspective and an efficient countermeasure against adversarial attacks, which we call 'Harmonizing Feature Maps'. Experimental results on the

*The lead author and main contributor.

Deep Learning Class - Xiamen University@Group of Kejia Zhang

CIFAR-10 dataset with different CNN architectures validate the effectiveness of our proposed methods, which can improve the robustness of CNNs under different type of attacks.

To sum up, the main contributions of this study are as follow:

- By reconstructing the feature maps with different frequency components through Fourier transform, we identify that the low-frequency components plays a critical role for the accuracy of CNN models.
- We propose a graph convolutional-based denoising method for improving the robustness, which jointly consider the inter feature map correlations and low-pass filtering property of GCN.

Related works

Adversarial Attacks and Defense

Adversarial attacks [9] pose a substantial threat in the field of computer vision, with the ability to mislead state-of-the-art Convolutional Neural Networks (CNNs) by adding human-imperceptible perturbations. Recently, several attack strategies have been proposed to further improve the disruptive ability of adversarial samples, such as the Fast Gradient Sign Method (FGSM), Projected Gradient Descent (PGD), and AutoAttack (AA) [8, 16, 3]. On the other hand, to deal with the influence of adversarial attacks, various defense strategies have been proposed to improve the robustness of CNN models [37, 35]. The adversarial training and architectural modifications are two representative defense strategies.

Adversarial training mainly aimed at enhancing the model's resilience by integrating adversarial examples into the training data, and has seen several significant advancements. For instance, GradAlign [1] is a regularization technique that maximizes gradient alignment within the perturbation set to prevent severe overfitting, thus improving the efficacy of FGSM solutions. Furthermore, the "learnable attack strategy" has been introduced in [12], which automates the generation of attack strategies to boost model robustness. Despite these advancements, adversarial training is computationally intensive and may fail to generalize to unseen attacks due to the model's inherent bias towards the adversarial examples used during training.

Architectural modifications constitute another defense strategy. Defense-GAN [22] harnesses the expressive power of generative models to establish a defense mechanism against adversarial attacks. Capsule networks, proposed by Sabour et al. [17], exhibit increased resilience to adversarial attacks by taking into account the hierarchical relationships between simple and complex objects. Despite their potential, such architectural modifications may introduce added complexity, possibly influencing the model's performance on clean data by overemphasizing adversarial noise at the cost of recognizing unperturbed inputs.

Recent studies have integrated Graph Neural Networks (GNNs) into CNN architectures for image denoising [25, 15, 6]. These methods effectively leverage both local and non-local similarities within image data, offering innovative solutions in areas such as high-resolution image denoising and evolutionary neural architecture search. Despite their contributions, these methods often neglect the relationship among different feature maps and lack a feature-centric justification for the feasibility of GNNs. Therefore, their applicability may be limited by the quality of graph construction and task specificity.

Although existing defense methods provide promising solutions, they still insufficient to deal with various adversarial attacks. The pursuit of comprehensive, efficient, and scalable defenses against these sophisticated attacks remains a thriving area of research.

Graph Neural Networks

Graph Neural Networks (GNNs) have proven to be a powerful tool in a wide range of applications due to their capability to process and model relational data. The inherent structure of GNNs facilitates the capture and leverage of relationships between different nodes, rendering them effective in different fields. For example, in social network analysis, GNNs can effectively detect and neutralize malicious nodes and edges [36, 27]. Similarly, in natural language processing, GNNs are used to eliminate adversarial noise from text data, thereby increasing the models' resistance to attacks [28, 33].

The low-pass filtering characteristics of GNNs make them highly effective against adversarial attacks, as highlighted in previous studies [13, 29]. These properties enable GNNs to preserve essential signal features while filtering out high-frequency adversarial noise, as previously demonstrated by [20, 21]. This feature proves particularly useful when dealing with adversarial perturbations that typically exploit high-frequency data components.

Considering the inherent vulnerabilities of Convolutional Neural Networks (CNNs) to adversarial attacks, this paper suggests employing the denoising capabilities of Graph Neural Networks (GNNs) to counter adversarial perturbations in the feature maps of CNN-based models. Given their successful application in enhancing model robustness across various domains [5, 4, 43], we hypothesize that integrating GNNs into CNN architectures, with their low-pass filtering properties and the ability to discern benign from malicious connections, can substantially improve model resilience against adversarial attacks. Consequently, our work aims to harness the potential of GNN-based denoising techniques to strengthen the robustness of CNN-based models by effectively mitigating adversarial noise in their feature maps.

New Insights into Feature Maps: Frequency Analysis

In this section, we investigate the frequency characteristics of feature maps and discuss their utilization for feature denoising. Specifically, we concentrate on the frequency analysis of feature maps and examine the impact of various frequencies on downstream tasks, such as image feature extraction and classification.

Reconstructing Signals from Fourier Coefficients

First, we evaluate the effects of various frequency features on the performance of Convolutional Neural Networks (CNNs),

utilizing the Fast Fourier Transform (FFT) and its inverse (IFFT). Based on the FFT and IFFT, we can convert signals between the time and frequency domains, therefore identifying critical frequency components contributing to the network's output [42].

Signal filtering in the frequency domain is pivotal for various image processing tasks. By removing extraneous frequency components, including adversarial perturbations, we can enhance the security and robustness of the model. Previous researches [40, 18] have shown that low-frequency features contain substantial structural and regional information, while high-frequency features capture intricate details and boundaries. Consequently, we investigate the effects of different frequency features on CNN performance, aiming to enhance model robustness by identifying frequency components that preserve valuable information during adversarial attacks.

Given the feature $\boldsymbol{X} \in R^{H\times W\times C}$ from a convolutional layer, we first convert the feature map of each channel c_i into one-dimensional feature vector $\boldsymbol{x_i} \in R^{H\times W}$. Then, we employ FFT on each flattened feature vector $\boldsymbol{x_i}$ to obtain a frequency spectrum for each feature map. This step can be denoted as:

$$\boldsymbol{f}_i = \mathit{FFT}(\boldsymbol{x}^i). \tag{1}$$

Next, we apply a low-pass filter on the frequency spectrum, preserving only the smallest p frequency components. We can obtain a new a frequency spectrum, $\boldsymbol{f}_i^{'}$:

$$\boldsymbol{f}_{i,k}^{'} = \begin{cases} \boldsymbol{f}_{i,k}, & if\ k \in I_p \\ 0, & otherwise \end{cases} \tag{2}$$

Here, I_p is the set of indices of the frequency components, corresponding to the smallest p frequency components in the FFT results.

Subsequently, we apply the IFFT to the low-pass filtered coefficient sequence $\boldsymbol{f}_i^{'}$ to reconstruct the input feature map. The output sequence, $\boldsymbol{x}_i^{'}$, is a real-valued feature vector reconstructed from the p smallest frequency components of the input feature vector:

$$\boldsymbol{x}_i^{'} = \mathit{IFFT}(\boldsymbol{f}_i^{'}). \tag{3}$$

Finally, after computing the reconstructed feature vector for each channel c_i, we then reshape the denoised feature vector back to its original dimensions, represented as $\boldsymbol{X'} \in R^{H\times W\times C}$. Subsequently, we can further forward the denoised feature maps into the following layers of the CNN to compute a new output after denoising.

Evaluating the Impact of Frequency Components on Classification Performance

In this section, we assess the influence of varying frequency components of the feature maps on the classification performance. Specifically, we use the first convolutional layer in the Wide Residual Network (WRN32-10) architecture trained on the CIFAR-10 dataset. The feature from the first convolutional layer of WRN32-10 contains 16 feature channels with a size of 32×32. We subsequently implement the FFT and IFFT using the Discrete Fourier Transform (DFT) and Inverse Discrete Fourier Transform (IDFT) to reconstruct the feature maps with varying frequency components.

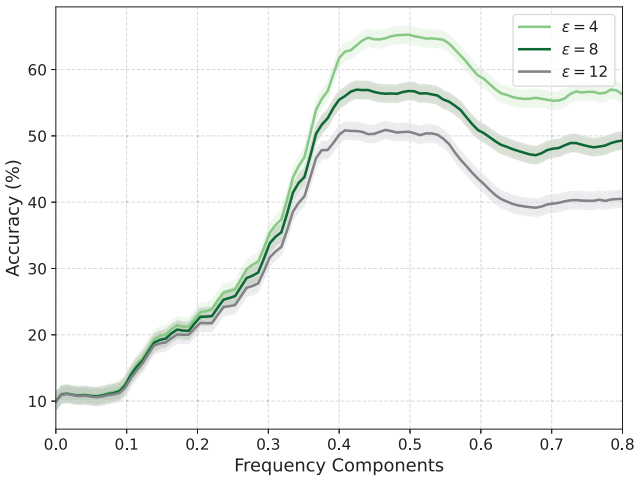


Figure 1: Impact of frequency components on classification accuracy.

Figure 1 illustrates the classification performance of the WRN32-10 in image classification tasks under different degrees of FGSM attacks. As shown in Figure 1, we can find that preserving approximately the first 40% of low-frequency signals enhances classification performance. In contrast, retaining high-frequency signals has a negative impact on the accuracy, especially under stronger attacks. This evidence suggests that selective filtering of feature maps reduces the amount of high-frequency noise and retain beneficial low-frequency components, thereby improving the robustness of CNN models.

Leveraging Inter-Feature Map Correlations for Graph-Based Feature Denoising

Based on the above analysis, we present a novel graph-based denoising method for improving the robustness of CNN models. Our method mainly consists of two parts. Firstly, we construct a graph to compute the inter-feature correlations between different feature channels. Besides, we leverage Graph Convolutional Neural Networks (GCNs) as a noise filter, which can effectively retain the key structural information from the input features. Compared with the conventional denoising methods, our method leverages the correlations among feature maps which have been overlooked by other methods. Additionally, our method exploits the inherent low-pass filtering properties of GCNs to mitigate high-frequency noise.

Constructing Inter-Feature Map Correlations

Recent studies [39, 10] have revealed that there are strong inter-feature map correlations within Convolutional Neural Networks (CNNs) since their features share local structures across different local paths in the spatial domain. However, these correlations are often overlooked by existing denoising methods. To address this issue, we introduce a graph-based

denoising approach that explicitly leverages inter-feature map information.

Given the feature $\boldsymbol{X}$, we first apply the global average pooling (GAP) on each feature channel $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$. Then, we have the mean-pooled feature vectors $\{\bar{\boldsymbol{x}}_1, \bar{\boldsymbol{x}}_2, \ldots, \bar{\boldsymbol{x}}_n\}$. This step serves to reduce the spatial dimensions of the feature maps while preserving their depth, which is beneficial for the subsequent steps of our approach. Next, we construct a similarity matrix S based on their Euclidean distance:

$$S_{i,j} = \exp(-\frac{\|\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}}_j\|^2}{\sigma^2}), \tag{4}$$

where $S_{i,j}$ represents the similarity score between two feature maps $\bar{\boldsymbol{x}}_i$ and $\bar{\boldsymbol{x}}_j$, and σ is a parameter that controls the decay rate of the similarity with the distance. To ensure that feature maps are not compared with themselves, we set the diagonal entries of S to negative infinity, *i.e.*, $S_{i,i} = -\infty$.

To exploit the inter-feature map correlations, we redefine the structure of our feature graph $G = (V, E)$, where V corresponds to the feature maps, and E contains edges between nodes. Specifically, for each feature map $\bar{\boldsymbol{x}}_i$, we consider it as a node in the graph. We then use the similarity matrix S to determine the edges in E.

In detail, for each node corresponding to $\bar{\boldsymbol{x}}_i$, we examine the i-th row of S, which contains the similarity scores between $\bar{\boldsymbol{x}}_i$ and all other feature maps. We then choose the top k nodes with the highest similarity scores and connect $\bar{\boldsymbol{x}}_i$ to these nodes, creating an edge in E. By doing so, we reconstruct G to better reflect the inter-feature map correlations.

With the graph G and the flattened feature map vectors $\boldsymbol{f}_i = vec(\boldsymbol{x}_i)$ for $i = 1, 2, ..., n$, we can represent the graph signal matrix F as:

$$F = [\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_n]^\top, \tag{5}$$

where each $\mathbf{f}_i$ corresponds to the flattened feature map of node i.

Graph Convolution as a Low-Pass Filter

Graph Convolutional Networks (GCNs) can be considered as low-pass filters for graph signals. They mitigate high-frequency components linked to local structures while enhancing low-frequency components indicative of global properties [30, 2]. To better illuminate this concept, we provide a simplified representation.

1. Transformation of Input Signal: The primary goal of GCNs is to convert an input signal X into an output signal y utilizing the normalized graph Laplacian L:

$$y = h_\theta(L)X, \tag{6}$$

This equation underscores that a GCN fundamentally acts as a graph signal filter.

2. Filtering in the Fourier Domain: This transformation, when represented in the Fourier domain, takes the form:

$$y = U h_\theta(\hat{L})\hat{X}, \tag{7}$$

where U contains orthogonal eigenvectors, $\hat{X}$ is the matrix of Fourier coefficients, and $\hat{L}$ is a matrix containing eigenvalues of L.

3. Low-Pass Filtering using Chebyshev Polynomial: We implement this low-pass filter using a Chebyshev polynomial approximation. This approximation is simplified by truncating the Chebyshev polynomial to order $K - 1$:

$$h_\theta(\lambda) = \sum_{k=0}^{K-1} \theta_k T_k \left(\frac{2L}{\lambda_{max}} - I \right). \tag{8}$$

This equation demonstrates how to filter out high-frequency components by adjusting the ratio of eigenvalue λ to the cut-off frequency λ_c. The term $(\frac{2L}{\lambda_{max}} - I)$ is crucial as it ensures the values of the normalized Laplacian's eigenvalues, which are the arguments of the Chebyshev polynomial, fall within its defined range of $[-1, 1]$. The Laplacian L of a graph, after normalization, has eigenvalues in the range $[0, 2]$. By scaling these eigenvalues with $\frac{2L}{\lambda_{max}}$ and subtracting the identity matrix I, we ensure that the values are within the necessary range for proper calculation of the Chebyshev polynomial. By adjusting these coefficients, the GCN effectively filters out high-frequency components.

Hence, by functioning as low-pass filters, GCNs effectively dampen the high-frequency components (*i.e.,* larger λ values) more than the low-frequency ones (*i.e.,* smaller λ values).

Designing Graph Convolution Blocks for Feature Denoising

In the previous section, we have analyzed the critical role played by low-frequency components in the reliable extraction and classification of image features. Additionally, Graph Convolutional Networks (GCNs) can effectively perform low-pass filtering for graph signals. As a result, we propose the use of a GCN-based denoising block to reduce noise within feature maps.

Our goal is to increase the robustness and resilience of CNNs against adversarial attacks by preserving low-frequency components and attenuating high-frequency components in feature maps, thereby improving subsequent image processing tasks. By transforming CNN feature maps into graph signals and applying GCN-based low-pass filtering, we can maintain the global image features and decrease redundancy during processing more effectively. Consequently, our approach encourages more efficient image feature extraction and classification by retaining essential information in the feature maps while reducing noise interference. The model architecture is depicted in Fig. 2.

Our proposed denoising block leverages the low-pass filtering property of GCNs to process the input flattened feature maps F (as defined in Eq. 5) and the graph $G = (V, E)$, constructed using inter-feature map information (Section). The GCN convolution operation, inclusive of residual connections, is defined as:

$$F' = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}XW) + F \tag{9}$$

Here, $\tilde{A}$ denotes the adjacency matrix of the graph G with self-connections, $\tilde{D}$ is the corresponding diagonal degree matrix, F is the input feature, and W is the weight matrix. After applying the GCN convolution to the input graph signal

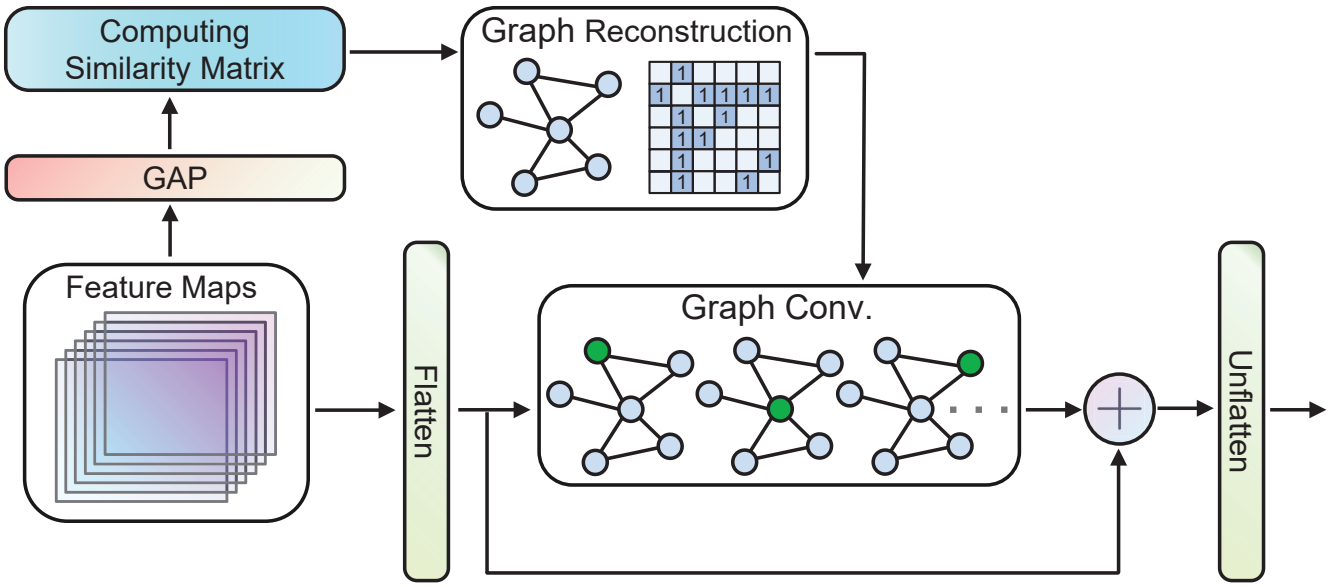


Figure 2: Illustration of the proposed GCN-based denoising block.

matrix, we reshape the output to original dimension to get the denoised feature maps.

Our proposed GCN-based denoising block can be seamlessly integrated into existing CNN architectures and applied to multiple convolutional layers within a CNN.

Experiments

Experimental Setting and Implementation Details: We assess the effectiveness of our proposed GCN-based denoising block by analyzing the performance of the Wide Residual Network 32-10 (WRN32-10) [38] on the CIFAR-10 dataset [14]. We use the CIFAR-10 dataset in this evaluation, which contains 50,000 training images and 10,000 testing images, each with a size of 32×32 pixels and 3 color channels (RGB).

Our GCN-based denoising block is integrated into the WRN32-10 architecture after several group convolutions, and we evaluate its performance under both regular and adversarial training methods. We perform adversarial attacks using several different methods, such as FGSM, PGD-20, PGD-100, and Auto-Attack, which consist of $['apgd-ce', 'apgd-dlr', 'fab-t', 'square']$. To accelerate the adversarial training process, we employed the "free" adversarial training method [23] with $m = 8$. And adversarial training parameters were adopted from the Free-8 model as detailed in [23]. This method concurrently updates model parameters and generates adversarial examples during forward and backward passes, which helps to overcome the potential computational load of our GCN-based denoising block. Besides, adversarial perturbations are constrained by the L_∞ norm with $\epsilon = 8$.

We use the SGD optimizer for optimization with a momentum of 0.9 and a weight decay of 0.0002. We set the initial learning rate to 0.1 and reduce it by 1/10 at the 100th and 150th epochs. The whole training epochs is 200, and the batch size is 128. To reconstruct the graph for the GCN-based denoising block, we restructure the graph underlying the features by selecting the top-5 most similar feature maps for each node. Our experiments were conducted on 4 NVIDIA RTX A4000 GPUs.

Enhancing Model Robustness via GCN-based Denoising Block

In this section, we present the evaluation of integrating the proposed GCN-based denoising into different positions of the WRN32-10 model. The results of regular training and adversarial training are reported in Table 1 and Table 2, respectively.

Our results, as demonstrated in Table 1, show that the inclusion of denoising blocks in the models can markedly improve their robustness against adversarial attacks, especially for PGD attacks. For instance, when denoising blocks are integrated into both the first and second convolutional layers, the accuracy of the model under the PGD-20 attack increases to 27.32% from 16.72% in the baseline model. This indicates a significant improvement in the model's resilience. Furthermore, the robustness of the model exhibited a direct correlation with the number of denoising blocks incorporated. For instance, when denoising blocks were included at all three convolutional layers, the accuracy under the PGD-20 attack increased even further to 28.19%.

On the other hand, as seen in Table 2, the introduction of denoising blocks did not yield a significant improvement in the performance of models under Auto Attack during adversarial training. This could be attributed to the optimization challenges presented by the inclusion of denoising blocks.

Overall, our results, as shown in Tables 1 and 2, confirm that the integration of the proposed GCN-based denoising block can significantly enhance the model's defense against adversarial attacks. While the improvement is particularly noticeable with PGD attacks and when denoising blocks are

Table 1: Robustness (accuracy(%)) comparison of WRN32-10 models with and without denoising blocks on CIFAR-10. Results with higher accuracy are **bold**.

Denoising Block Position			Natural	Accuracy under White-box Attack ($\epsilon = 8$)			
Conv.1	Conv.2	Conv.3		FGSM	PGD-20	PGD-100	AA
			95.88	53.25	16.72	16.61	13.73
✓			95.76	55.84	23.87	23.76	24.77
	✓		95.85	54.80	18.49	18.61	18.21
		✓	**96.03**	53.55	17.21	16.91	16.89
✓	✓		95.91	**59.64**	27.32	27.04	29.24
✓	✓	✓	95.80	58.84	**28.19**	**27.95**	**30.19**

Table 2: Comparison of WRN32-10 models on CIFAR-10: Accuracy (%) under "free" adversarial training. Results with higher accuracy are **bold**.

Denoising Block Position			Natural	Accuracy under White-box Attack ($\epsilon = 8$)			
Conv.1	Conv.2	Conv.3		FGSM	PGD-20	PGD-100	AA
			80.21	75.70	46.05	45.98	26.98
✓			**81.01**	**77.38**	47.87	47.85	30.05
	✓		80.72	77.15	47.87	47.85	29.75
		✓	80.35	76.45	47.09	46.92	27.15
✓	✓		80.32	76.16	48.33	47.01	31.49
✓	✓	✓	80.09	76.74	**48.88**	**48.70**	**32.91**

incorporated into the earlier convolutional layers, further research is warranted to address the optimization challenges encountered during Auto Attacks.

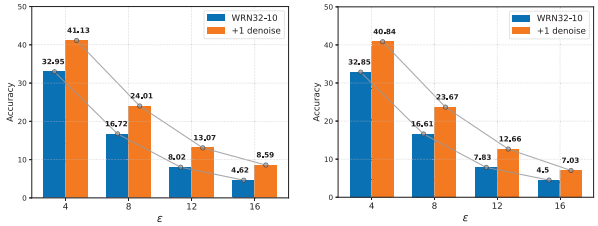
Scalability of the Proposed Approach to ResNet-18 Architecture

To further evaluate the scalability of the proposed GCN-based denoising block, we incorporated it into the ResNet-18 architecture. From Table 3, we can have following observations. **(1)** Adding the denosining block into the normal trained model can improve the robustness of ResNet-18 against various adversarial attacks. For example, the accuracy will increase from 16.29% to 20.67% under the PGD-20 attack. **(2)** The denosining block in also effective in the adversarial trained ResNet-18. For instance, the accuracy of the model under the PGD-20 attack increased from 43.82% to 45.40% when the denoising block was included.

These results demonstrate the versatility and scalability of our proposed GCN-based denoising block as a defense mechanism against adversarial attacks. The block has demonstrated effectiveness in improving model robustness across different deep learning architectures, such as WRN32-10 and ResNet-18, and under varying adversarial attack conditions.

Effectiveness of the Proposed GCN-based Denoising Block under Various ϵ Attacks

In the section, we evaluate the robustness of our proposed GCN-based denoising block under different values of the adversarial attack magnitude ϵ (4, 8, 12, and 16). We add a single GCN block after the first convolutional layer in the



(a) PGD-20 (b) PGD-100

Figure 3: Comparative analysis of WRN32-10's robustness: with and without the GNN-based denoising block under varying ϵ attacks

WRN32-10 model and test the performance under the PGD-20 and PGD-100 attacks. From the result in Figure 3, we can find that adding the denoising block can consistently improved the model's performance. Notably, the accuracy gains ranged between 2.53% and 8.17% under different attack magnitudes compared to the baseline model.

Concurrently, we observe that the model's accuracy decreased as the ϵ value increased. For example, under PGD-20 and PGD-100 attacks, model accuracy dropped roughly by 30% when the ϵ value escalated from 4 to 16. This observation underscores the susceptibility of the model to more intensive white-box attacks, further emphasizing the necessity for robust and effective defense strategies against such attacks.

Table 3: Comparison of Robustness (accuracy(%)) for ResNet-18 model with and without our GCN-based denoising block and adversarial training on CIFAR-10. The denoising block is added after the first convolutional layer. Higher accuracy results are highlighted in **bold**.

Model Information		Natural	Accuracy under White-box Attack ($\epsilon = 8$)			
Denoising Block	Adv. Training		FGSM	PGD-20	PGD-100	AA
		95.03	50.44	16.29	16.06	14.10
	✓	76.00	71.32	43.82	43.80	22.43
✓		94.93	51.69	20.67	20.47	21.54
✓	✓	76.46	**72.97**	**45.40**	**45.35**	**24.35**

Conclusion and Limitations

In this study, we aim to integrate Graph Neural Networks (GNNs) and Convolutional Neural Networks (CNNs) through the introduction of a novel GCN-based denoising block. This unique design significantly enhances the robustness of CNNs against adversarial attacks and highlights the untapped potential of leveraging GCNs for image processing tasks.

Despite these advancements, our research has certain limitations, primarily the added computational complexity resulting from the GCN-based denoising block. It may increase the resource demand, which could potentially hinder the model's scalability and efficiency, especially in resource-constrained settings. Moreover, our study primarily focused on the application of Graph Convolutional Networks. Future research could explore other types of GNNs, such as Graph Attention Networks or Graph Isomorphism Networks, for potential integration and their possible impacts on adversarial robustness.Despite these challenges, our study provides a promising foundation for future research, aimed at fortifying the synergy between GCNs and CNNs, and bolstering CNN resilience against adversarial attacks.

References

[1] Andriushchenko, M.; and Flammarion, N. 2020. Understanding and improving fast adversarial training. In *NeurIPS*.

[2] Chen, L.; Wu, L.; Hong, R.; Zhang, K.; and Wang, M. 2020. Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach. In *AAAI*, volume 34, 27–34.

[3] Croce, F.; and Hein, M. 2020. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2206–2216.

[4] Dai, E.; Zhao, T.; Zhu, H.; Xu, J.; Guo, Z.; Liu, H.; Tang, J.; and Wang, S. 2022. A comprehensive survey on trustworthy graph neural networks: Privacy, robustness, fairness, and explainability. *arXiv preprint arXiv:2204.08570*.

[5] Geisler, S.; Schmidt, T.; Şirin, H.; Zügner, D.; Bojchevski, A.; and Günnemann, S. 2021. Robustness of Graph Neural Networks at Scale. In *NeurIPS*.

[6] Gidaris, S.; and Komodakis, N. 2019. Generating Classification Weights With GNN Denoising Autoencoders for Few-Shot Learning. In *CVPR*.

[7] Goel, A.; Agarwal, A.; Vatsa, M.; Singh, R.; and Ratha, N. K. 2020. DNDNet: Reconfiguring CNN for Adversarial Robustness. In *CVPR Workshops*.

[8] Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

[9] Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and harnessing adversarial examples. In *ICLR*.

[10] Guo, X.; Yang, K.; Yang, W.; Wang, X.; and Li, H. 2019. Group-wise correlation stereo network. In *CVPR*, 3273–3282.

[11] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

[12] Jia, X.; Zhang, Y.; Wu, B.; Ma, K.; Wang, J.; and Cao, X. 2022. LAS-AT: Adversarial Training With Learnable Attack Strategy. In *CVPR*.

[13] Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

[14] Krizhevsky, A.; and Hinton, G. 2009. CIFAR-10 (Canadian Institute For Advanced Research). Technical report, Canadian Institute for Advanced Research.

[15] Li, Y.; Fu, X.; and Zha, Z.-J. 2021. Cross-Patch Graph Convolutional Network for Image Denoising. In *ICCV*.

[16] Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *ICLR*.

[17] Marchisio, A.; Nanfa, G.; Khalid, F.; Hanif, M. A.; Martina, M.; and Shafique, M. 2019. Capsattacks: Robust and imperceptible adversarial attacks on capsule networks. *arXiv preprint arXiv:1901.09878*.

[18] Miangoleh, S. M. H.; Dille, S.; Mai, L.; Paris, S.; and Aksoy, Y. 2021. Boosting Monocular Depth Estimation Models to High-Resolution via Content-Adaptive Multi-Resolution Merging. In *CVPR*.

[19] Mummadi, C. K.; Brox, T.; and Metzen, J. H. 2019. Defending Against Universal Perturbations With Shared Adversarial Training. In *ICCV*.

[20] Oyallon, E.; Belilovsky, E.; and Zagoruyko, S. 2017. Scaling the scattering transform: Deep hybrid networks. In *ICCV*.

[21] Saikia, T.; Schmid, C.; and Brox, T. 2021. Improving Robustness Against Common Corruptions With Frequency Biased Models. In *ICCV*.

[22] Samangouei, P.; Kabkab, M.; and Chellappa, R. 2018. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *ICLR*.

[23] Shafahi, A.; Najibi, M.; Ghiasi, M. A.; Xu, Z.; Dickerson, J.; Studer, C.; Davis, L. S.; Taylor, G.; and Goldstein, T. 2019. Adversarial training for free! *NeurIPS*.

[24] Sung, F.; Yang, Y.; Zhang, L.; Xiang, T.; Torr, P. H.; and Hospedales, T. M. 2018. Learning to Compare: Relation Network for Few-Shot Learning. In *CVPR*.

[25] Valsesia, D.; Fracastoro, G.; and Magli, E. 2020. Deep Graph-Convolutional Image Denoising. *TIP*, 29: 8226–8237.

[26] Vellaichamy, S.; Hull, M.; Wang, Z. J.; Das, N.; Peng, S.; Park, H.; and Chau, D. H. P. 2022. DetectorDetective: Investigating the Effects of Adversarial Examples on Object Detectors. In *CVPR*, 21484–21491.

[27] Wu, C.; Wu, F.; Lyu, L.; Qi, T.; Huang, Y.; and Xie, X. 2022. A federated graph neural network framework for privacy-preserving personalization. *Nature Communications*, 13(1): 3091.

[28] Wu, L.; Chen, Y.; Ji, H.; and Liu, B. 2021. Deep learning on graphs for natural language processing. In *SIGIR*.

[29] Wu, Z.; Pan, S.; Long, G.; Jiang, J.; and Zhang, C. 2022. Beyond Low-Pass Filtering: Graph Convolutional Networks With Automatic Filtering. *KDD*, 1–12.

[30] Wu, Z.; Pan, S.; Long, G.; Jiang, J.; and Zhang, C. 2022. Beyond low-pass filtering: Graph convolutional networks with automatic filtering. *KDD*.

[31] Xie, C.; Wu, Y.; Maaten, L. v. d.; Yuille, A. L.; and He, K. 2019. Feature Denoising for Improving Adversarial Robustness. In *CVPR*.

[32] Xie, X.; Cheng, G.; Wang, J.; Yao, X.; and Han, J. 2021. Oriented R-CNN for Object Detection. In *ICCV*, 3520–3529.

[33] Yang, J.; Liu, Z.; Xiao, S.; Li, C.; Lian, D.; Agrawal, S.; Singh, A.; Sun, G.; and Xie, X. 2021. GraphFormers: GNN-nested Transformers for Representation Learning on Textual Graph. In *NeurIPS*, volume 34, 28798–28810.

[34] Yang, T.; Zhu, S.; Chen, C.; Yan, S.; Zhang, M.; and Willis, A. 2020. Mutualnet: Adaptive convnet via mutual learning from network width and resolution. In *ECCV*, 299–315.

[35] Yin, D.; Gontijo Lopes, R.; Shlens, J.; Cubuk, E. D.; and Gilmer, J. 2019. A fourier perspective on model robustness in computer vision. *NeurIPS*, 32.

[36] You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; and Shen, Y. 2020. Graph Contrastive Learning with Augmentations. In *NeurIPS*.

[37] Yuan, X.; He, P.; Zhu, Q.; and Li, X. 2019. Adversarial examples: Attacks and defenses for deep learning. *TNNLS*, 30(9): 2805–2824.

[38] Zagoruyko, S.; and Komodakis, N. 2016. Wide residual networks. *arXiv preprint arXiv:1605.07146*.

[39] Zeiler, M. D.; and Fergus, R. 2014. Visualizing and understanding convolutional networks. In *ECCV*, 818–833.

[40] Zeiler, M. D.; and Fergus, R. 2014. Visualizing and understanding convolutional networks. In *ECCV*.

[41] Zhang, S.; Gao, H.; and Rao, Q. 2021. Defense Against Adversarial Attacks by Reconstructing Images. *TIP*, 30: 6117–6129.

[42] Zhou, K.; Yu, H.; Zhao, W. X.; and Wen, J.-R. 2022. Filter-enhanced MLP is all you need for sequential recommendation. In *WWW*, 2388–2399.

[43] Zügner, D.; and Günnemann, S. 2019. Certifiable robustness and robust training for graph convolutional networks. In *SIGKDD*, 246–256.