Image Denoising using Deep Residual Blocks with Fourier Transform

[*]Note: Authors listed in last name alphabetical order, around the same contribution

Ruijie Hou
Department of Computer Science
University of Toronto
ruijie.hou@mail.utoronto.ca

Xiaoran Liu
Department of Computer Science
University of Toronto
xiaoran.liu@mail.utoronto.ca

Wenzhe Xu
Department of Computer Science
University of Toronto
wz.xu@mail.utoronto.ca

***Abstract*—Image Denoising has been intensively studied in the image processing pipeline and computer vision tasks. Existing developed approaches are mainly traditional spatial filtering algorithms and real-valued deep learning methods. However, few have attempted to incorporate Fourier transforms into inner network structures for image denoising. In this paper, we present an innovative neural network architecture based on previous image processing research, filtering out noise in both the time and frequency domains and using residual blocks. We name the proposed architecture as FFTResCNN, and compare its performance against several other denoising methods, namely BM3D, DnCNN, and ResDnCNN, using average PSNR as the performance metric. The experimental results demonstrate FFTResCNN's superior performance in image denoising, which successfully integrates time and frequency residual information to preseve both high- and low-frequency details.**

I. Introduction

In recent years, images have become an essential medium for humans to obtain information. Noise is inevitably mixed (Fan et al., 2019) during acquisition, compression, and transmission due to the influence of the environment, transmission channel, and other factors, resulting in distortion and loss of image information. These noises interfere with the human understanding of the original picture and subsequent image-processing applications. Thus, Image Denoising, which removes noise from an input image while maintaining high-frequency details, is critical in the image processing pipeline and computer vision tasks. So far, researchers have proposed various strategies for reducing noise, ranging from classic spatial domain filtering to neural networks (Fan et al., 2019).

With the more profound exploration of deep learning methods, in addition to the traditional real-valued neural networks, some researchers have begun to investigate the use of the complex domain nature of networks and images. One motivation that should be considered is the difference between the spectrums of the clear image and the corresponding noisy image following (Fast) Fourier Transformation, as Fig. 1 shows. However, few have investigated adding Fourier transforms into inner network structures for image-denoising tasks. This research aims to propose an innovative neural network architecture for image denoising based on previous image processing research, filtering out noise in both the time and frequency domains and using residual blocks.

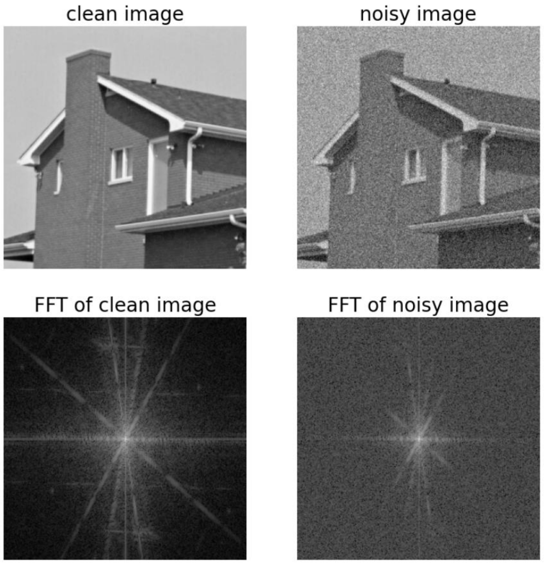

Fig. 1: The spectrums of the clear image and the corresponding noisy image following Fourier Transformation

This report reviews previous related literature, followed by detailed descriptions of the data and statistical methods used and the results of the analysis, including discussions of the preliminary findings, limitations, and future steps of the study.

II. Related Work

Image denoising has been studied extensively, and there are many techniques with both advantages and disadvantages. The existing developed methods are mainly traditional filtering algorithms and deep-learning-based methods. Traditional algorithms like Spatial Domain Filtering and Transform Domain Filtering generally target noise that can be easily removed, and the computational cost is relatively moderate. Nevertheless, Spatial Filtering blurs and reduces image sharpness. Some Transform Domain Filtering methods are time-consuming and rely on the behaviour of the filter function and cut-off frequency (Fan et al., 2019).

Because of its advantages, such as significant feature learning capability, convolutional neural network (CNN) and its variants have gained popularity and advancements with the growth of computational power. In particular, the studies on neural networks are no longer limited to the field of real numbers. According to recent research, complex-valued CNN (CVCNN), which is built on the basic CNN with complex number operations and computations, offers another promising deep-learning method for image denoising (Quan et al., 2021). This innovation focuses on neural networks' structure, while the complex-valued properties in the frequency domain are disregarded. Since the Fourier transform can turn an image into the frequency domain, researchers have shown that applying the CVCNN to noisy images after Fourier transform, most of the time, outperforms some state-of-the-art deep learning methods (Pham et al., 2021). However, such processing of images and networks is relatively complicated and is difficult to generalize. For other image processing tasks, there have been attempts to combine Fourier transform with the internal neural network structure. Recently, a method inserting Fourier transforms to the inner layer structure of the neural networks has achieved competitive PSNR and SSIM scores on image deblurring, which demonstrates the field's potential and inspires this report (Mao et al., 2021).

III. Theory

A. BM3D

The Block-matching and 3D filtering (BM3D) algorithm is one of the most effective expansions of the Non-Local Means algorithm, which achieves excellent performance in a wide range of image-denoising tasks and is regarded as the most powerful of the traditional denoising approaches (Lebrun, 2012). BM3D is an image-denoising technique that groups like-minded 2D picture fragments into 3D data arrays to create an enhanced sparse representation in the transform domain (Dabov et al., 2007). It is separated into two stages, including the following components: matched-blocks grouping, collaborative filtering, and aggregation, where collaborative filtering in step one employs hard thresholding and Wiener filtering in step two. It initially takes the mean of all identical patches to generate a 3D matrix, which is then decomposed using a 3D unitary sparsifying transform. The sparse representation is subsequently filtered via collaborative filtering to remove high-frequency sounds by achieving coefficient shrinkage. The filtered block is subjected to an inverse transform, yielding a denoise 3D block. The target patch for denoise is then estimated by taking the weighted average of all patches in the block. The scheme of BM3D is shown in Fig. 2.

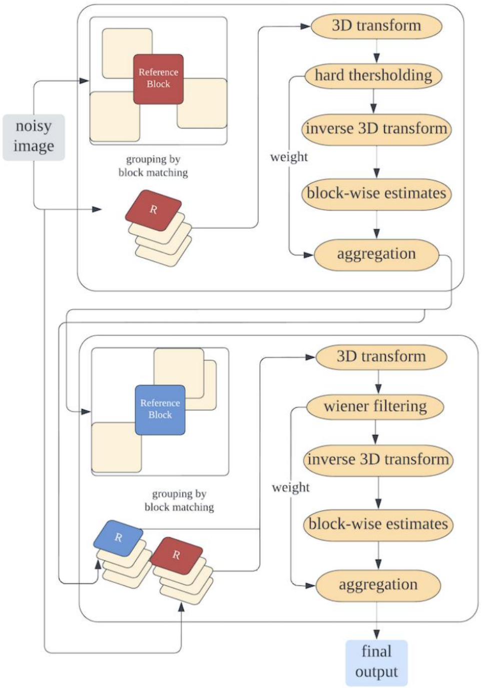

Fig. 2: The architecture of BM3D

B. DnCNN

The DnCNN is a denoising convolutional neural network that eliminates noise from a Gaussian noisy input picture (Zhang et al., 2017). The denoised image is computed by subtracting the residual image from the noisy image. In this algorithm, residual learning and batch normalization are used to accelerate the training process and improve denoising performance. DnCNN's residual learning technique implicitly eliminates the latent clean image in the hidden layers, making the inputs to each layer Gaussian dispersed, less correlated, and less relevant to the picture content. Furthermore, unlike previous discriminative denoising models that typically train a specific model for additive white Gaussian noise at a specified noise level, the DnCNN model can handle Gaussian denoising at an unknown noise level (i.e., blind Gaussian denoising). The architecture of DnCNN is shown in Fig. 3 (Zhang et al., 2017).

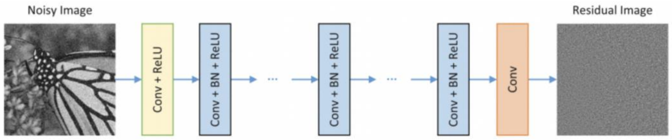

Fig. 3: The architecture of DnCNN network

C. Proposed Structures

The DnCNN is a powerful deep-learning-based denoising model, and many of the most influential modern denoising models nowadays are developed based on DnCNN. The network structure of DnCNN is relatively simple, and research shows that the DnCNN model has the property of fast convergence and efficiency in processing small datasets (Murali and Sudeep, 2020). With that in mind, we create a novel architecture based on DnCNN by incorporating Fourier transforms into inner network structures, which allows us to obtain the desired outcomes with limited resources. We propose a residual block with the fast Fourier transform (FFTResBlock), as illustrated in Fig. 4, to learn the features of an image in the frequency domain. This block contains the following three streams: (a) a regular convolution stream in the middle; (b) the FFT-based convolution stream on the left; and (c) a standard spatial residual stream on the right. Regular convolution operators can learn high-frequency details since they usually extract informative features from edges. The FFT-based convolution stream has a bigger receptive field, or more low-frequency global information, because of the characteristics of the Fourier transform and frequency domain. This block's ability to comprehend both high-frequency and low-frequency information is thus a benefit. We also use the spatial residual stream to balance the complexity increase brought on by the FFT convolution stream through the advantages of residual blocks, such as supporting larger receptive field size, deeper networks, and faster convergence during training.

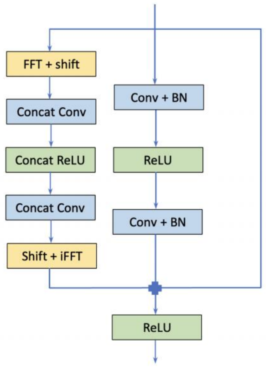

Fig. 4: Residual block with the fast Fourier transform

In the proposed FFTResBlock, the detailed processing of the FFT-based convolution stream for feature map $\mathcal{I}$ is as follows. Considering the time complexity, 2D real FFT and its inverse transform are used in this project.

1) Real FFT is performed on $\mathcal{I}$ to obtain $\mathcal{F}(\mathcal{I})$. $\mathcal{F}(\mathcal{I})$ values are complex numbers, each consisting of real and imaginary parts.
2) Concatenate the real part and imaginary part of $\mathcal{F}(\mathcal{I})$ together along the dimension of the channel.
3) Employ two convolution layers with a ReLU activation layer in between to the new feature map.
4) The real part and the corresponding imaginary part are added together to form complete complex numbers, and then apply the inverse real FFT.

Furthermore, we propose a neural network named FFTResCNN by plugging the FFTResBlock mentioned above into the DnCNN. Fig. 5 depicts the specific insertion procedure. In the middle layers of the DnCNN, a convolution layer, a batch normalization layer and a ReLU activation layer are regarded as one unit. We combine two consecutive units and replace a certain number of units with the proposed

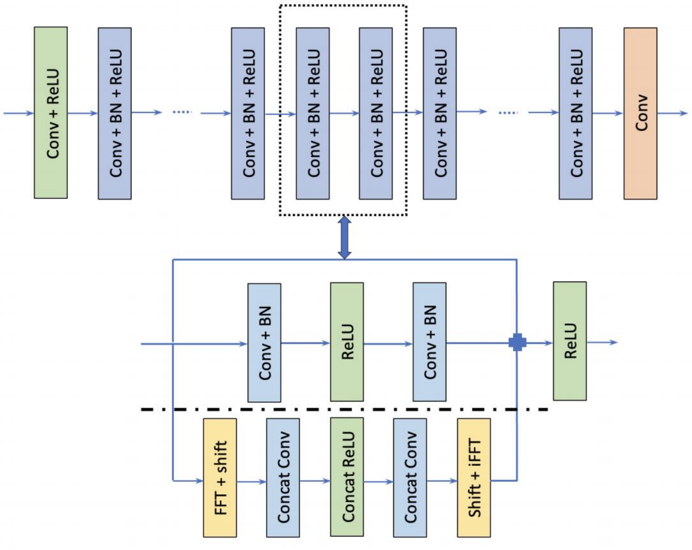

Fig. 5: The architecture of FFTResCNN

FFTResBlock. If the number of total layers is N, then the number of such blocks is no larger than $\frac{N-2}{2}$. This way, we can generate FFTResCNN without altering the ordinary convolution structure. By doing so, it is more reasonable to infer that the FFT-based convolution stream is the primary cause of the discrepancy between the two models instead of other factors. In addition, we remove the FFT convolution stream to create a basic residual block to rule out the possibility that the improvement on model performance comes from the spatial residual stream rather than the FFT convolution stream. The ResDnCNN network is created by inserting the basic residual blocks into the DnCNN in a similar manner. In the following sections, we analyze and compare the BM3D, DnCNN, ResDnCNN, and FFTResCNN mentioned above to explore the effect of FFT-based convolution stream on deep-learning-based denoising models.

D. Evaluation metric

Various metrics can be employed to measure the similarities between the output and the original image when comparing them. We use the Peak signal-to-noise ratio (PSNR) value as a quantitative evaluation metric, which is defined as

$$PNSR = 10log_{10}(\frac{MAX_i^2}{MSE}) \tag{1}$$

where MAX$_i$ is the image's dynamic range pixel value and MSE is defined as

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2 \tag{2}$$

where I indicates the original image and K indicates the approximation.

IV. Analysis and Evaluation

In order to make a fair comparison to the previous works, for image denoising methods investigated in this paper, i.e., BM3D, DnCNN, ResDnCNN, and FFTResCNN, we make use of the same datasets by Zhang et al. (2017), who proposed the DnCNN model. Following the conventions from Zhang et al. (2017), the noise level σ stated in this paper refers to the standard deviation of $\frac{\sigma}{255}$ for additive white Gaussian noise. The training and test datasets are preprocessed in the following manner.

A. Training

Due to limitations in computational resources and time, we only considered the noise level $\sigma = 25$ for blind Gaussian denoising. To generate the training data for our denoising experiments, we first crop 25 patches of size 50×50 from each of the 400 grayscale images of size 180×180 used by Zhang et al. (2017), which were previously cropped from the images of the Berkeley segmentation dataset (BSDS500) (Martin et al., 2001). The resulting dataset is also augmented by a factor of 2 with random rotation/flip-based operations. The noisy images are then generated by adding Gaussian noise with noise level $\sigma = 25$ to each cropped patch. Our training set thus contains 20,000 patches, i.e., roughly 50 million pixels, which is about 5% of the data used for blind Gaussian denoising by Zhang et al. (2017).

B. Parameter Setting

Following Zhang et al. (2017), who set the optimal network depth to 20 for their DnCNN model used for blind Gaussian denoising, we tested different network depths in the range of d $\in [16, 24]$ for our FFTResCNN model to capture enough spatial information for image denoising. As in Eqn. (2), the averaged mean squared error between the ground truth image and the estimated denoised image is adopted as the loss function to learn the trainable parameters. We initialize the parameters via Kaiming initialization (He et al., 2015), and train the models using the Adam optimization algorithm (Kingma and Ba, 2014) with a learning rate of 0.001, betas of (0.9, 0.999), and a batch size of 64. The training epochs are fixed to 50 for all three deep-learning-based models for a fair performance comparison.

C. Validation and Testing

At the end of each training epoch of DnCNN, ResDnCNN, and FFTResCNN, the denoising performance of the trained model is evaluated on the 12 images in the SET12 dataset as shown in Fig. 6, which are widely used for the evaluation of Gaussian denoising methods (Zhang et al., 2017) and are not part of our training dataset.

At test time, we use two different test datasets for comprehensive evaluation. In addition to the 12 images used for validation, we also evaluate our models on the 68 natural images from the test section of the Berkeley dataset (BSD68), which is suggested by (Roth and Black, 2005) and later widely used for measuring image denoising algorithms performance. Note that all test images are strictly separated from the images in the training dataset.

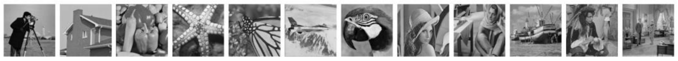

Fig. 6: SET12 - the 12 widely used testing images

D. Experimental Setting and Results

We use the PyTorch implementation of the DnCNN model written by (Yan, 2017) as the basis for the implementation of our deep-learning-based models. Unless otherwise specified, all experiments are carried out with an Nvidia GeForce RTX 2070 GPU. It takes about 60, 60 and 80 minutes to train the DnCNN, ResDnCNN, and FFTResCNN models on GPU respectively.

Based on the average PSNR on the test datasets, the optimal number of layers and residual blocks are determined to be 18 and 6 respectively. Fig. 7 shows the Gaussian denoising results of the three deep-learning-based models on the validation dataset with respect to epochs. Fig. 8 and Fig. 9 shows the sample noisy and denoised images generated during training (epoch 15) and at the end of training (epoch 50) respectively. These results demonstrated FFTResCNN's superior performance at each intermediate training step.

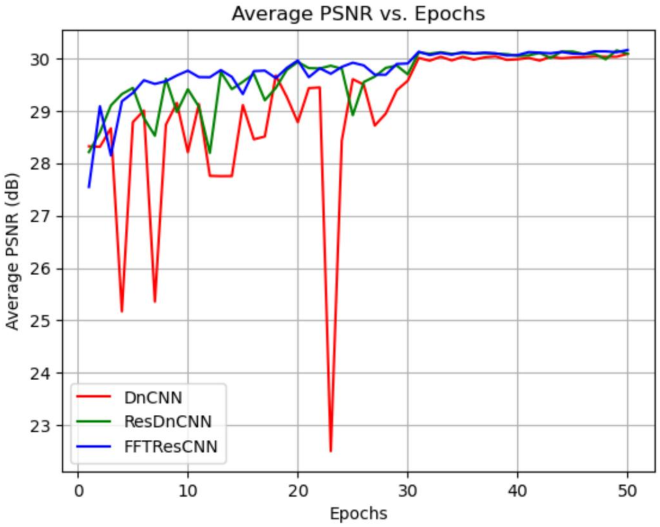

Fig. 7: Gaussian denoising results under Adam optimization algorithm, with respect to epochs

V. Results

The performance of all the models is evaluated by the average PSNR with different noise levels on the two test datasets. Table I shows the average PSNR evaluated on the SET12 dataset, and Table II shows the average PSNR evaluated on the BSD68 dataset. Fig. 10 shows the final denoising results of one image from the test dataset using DnCNN and FFTResCnn.

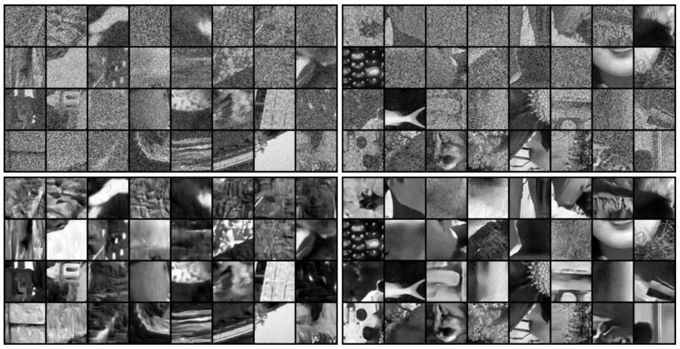

Fig. 8: Sample noisy and denoised images at epoch 15; Left: DnCNN, Right: FFTResCNN

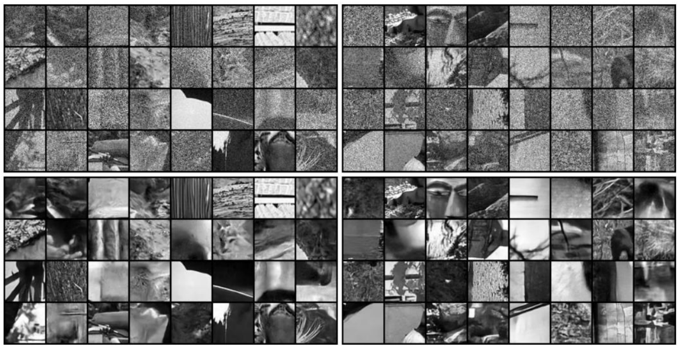

Fig. 9: Sample noisy and denoised images at epoch 50; Left: DnCNN, Right: FFTResCNN

A primary finding is that in most circumstances, FFTResCNN has the best performance as expected, outperforming ResDnCNN, DnCNN and BM3D, which suggests that the proposed FFTResBlock is able to improve the denoising performance. This improvement benefits from FFT-based convolution stream rather than just the residual stream. Nevertheless, an interesting observation is that when the noise level is relatively high (e.g., the noise level is 50), the ResDnCNN performs comparably to the FFTResCNN (on SET12) and sometimes even slightly outperforms the FFTResCNN in terms of PSNR (on BSD68).

Noise Level	BM3D	DnCNN	ResDnCNN	FFTResCNN
15	26.184	32.342	32.411	**32.527**
25	25.451	30.027	30.089	**30.139**
50	23.172	26.739	26.855	**26.876**

TABLE I: Comparisons on SET12 Average PSNR (in dB)

Noise Level	BM3D	DnCNN	ResDnCNN	FFTResCNN
15	24.461	31.322	31.390	**31.466**
25	23.805	28.883	28.953	**28.992**
50	22.297	25.908	**26.027**	25.982

TABLE II: Comparisons on BSD68 Average PSNR (in dB)

Fig. 10 shows the denoised results of one image example from the SET12 test dataset using BM3D, DnCNN and FFTResCnn. It can be seen that BM3D denoising makes the noisy image look smooth and natural, but its downside is that many details are lost, and the denoised image is blurry. The denoised results of DnCNN and FFTResCnn are much better than BM3D. It is worth noting that in the denoised image of FFTResCnn, compared to that of DnCNN, the background of the sky looks smoother, while the ground and the crease in the pants is more apparent, involving more detailed information. This finding might support our conclusion that the proposed architecture is able to comprehend both high- and low-frequency information.

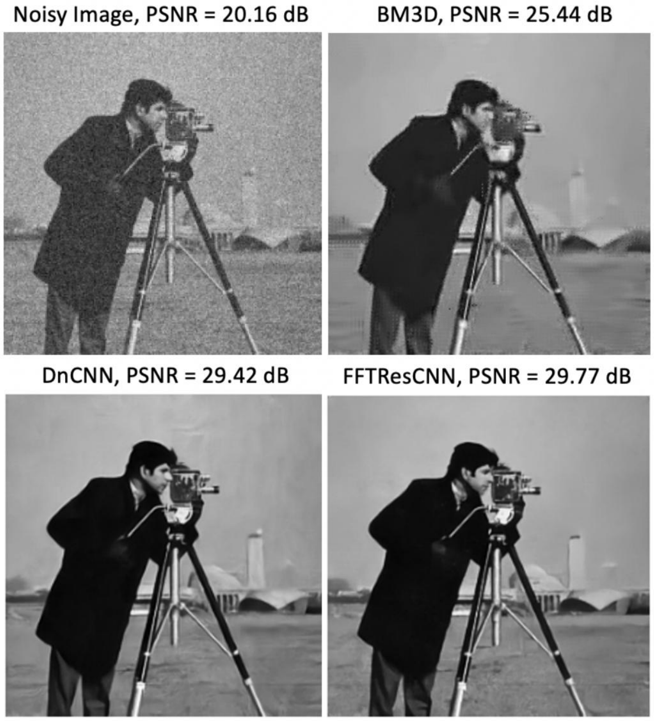

Fig. 10: BM3D vs. DnCNN vs. FFTResCNN: final denoising results of one image from the test dataset ($\sigma = 25$)

VI. Discussion and Conclusion

In this project, we implemented four denoising algorithms, one traditional spatial method: BM3D, and three deep learning methods: DnCNN, ResDnCNN and FFTResCNN. In general, an analysis of the findings suggests that the proposed architecture in this study helps to improve the denoising performance of the original network. This enhancement likely comes from the properties of the proposed FFTResBlock, learning from both the time and frequency domain and integrating both low- and high-frequency information.

In the Results section, a finding worth discussing is that although FFTResCNN generally outperforms other methods, ResDnCNN is competitive on relatively high noise levels and sometimes yields even better results. We speculate that one reason may be the size of the noise level applied during the blind training. Recall that in the training steps, the noise levels are randomly selected from 0 to 55. A noise level of 50 is close to the endpoint, so there might not be enough training data around that noise level. Another conjecture is that when the noise level is too large, the frequencies are similar in the frequency domain, so the global information learned from spectrums might be less helpful.

In the context of this report, it must also be mentioned that there exist certain limitations. These limitations need to be addressed in future research to further validate our claims. One main limitation is the limited number of training data. Previous research used around 200,000 samples in training, while we only trained models on 20,000 samples due to the limited time and hardware resources. Thus, more training data and data augmentation methods should be involved. Also, since we only investigate the performance on grey scale pictures, training and testing data may be altered in the future to validate the performance using different datasets, such as colour RGB photos or videos. Another limitation is the noise level discussed before. Further studies could try to contrast and observe the denoising performance of each algorithm at more training noise levels and observe the performance with the changes in noise levels. Moreover, we would generalize and apply the proposed blocks or similar ideas to other networks for future work.

REFERENCES

Dabov, K., Foi, A., Katkovnik, V., & Egiazarian, K. (2007). Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on image processing*, *16*(8), 2080–2095.

Fan, L., Zhang, F., Fan, H., & Zhang, C. (2019). Brief review of image denoising techniques. *Visual Computing for Industry, Biomedicine, and Art*, *2*(1), 1–12.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *Proceedings of the IEEE international conference on computer vision*, 1026–1034.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lebrun, M. (2012). An analysis and implementation of the bm3d image denoising method. *Image Processing On Line*, *2012*, 175–213.

Mao, X., Liu, Y., Shen, W., Li, Q., & Wang, Y. (2021). Deep residual fourier transformation for single image deblurring. *arXiv preprint arXiv:2111.11745*.

Martin, D., Fowlkes, C., Tal, D., & Malik, J. (2001). A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, *2*, 416–423.

Murali, V., & Sudeep, P. (2020). Image denoising using dncnn: An exploration study. In *Advances in communication systems and networks* (pp. 847–859). Springer.

Pham, M. T., Nguyen, V. Q., Hoang, C. D., Vo, H. L., Phan, D. K., & Nguyen, A. H. (2021). Efficient complex valued neural network with fourier transform on image denoising. *The 5th International Conference on Future Networks & Distributed Systems*, 48–57.

Quan, Y., Chen, Y., Shao, Y., Teng, H., Xu, Y., & Ji, H. (2021). Image denoising using complex-valued deep cnn. *Pattern Recognition*, *111*, 107639.

Roth, S., & Black, M. J. (2005). Fields of experts: A framework for learning image priors. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, *2*, 860–867.

Yan, Y. (2017). Dncnn-pytorch [Available at https://github.com/SaoYan/DnCNN-PyTorch].

Zhang, K., Zuo, W., Chen, Y., Meng, D., & Zhang, L. (2017). Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, *26*(7), 3142–3155.