Image Denoising using Convolutional Auto-encoders

Abhijit Mahalle
University of Maryland
College Park, MD
abhimah@umd.edu

Rohan Purekar
University of Maryland
College Park, MD
rpurekar@umd.edu

Nipun Kumar
University of Maryland
College Park, MD
npatturu@umd.edu

Abstract

In this project, we developed a very deep, fully convolutional auto-encoder network for image denoising, which utilizes symmetrical convolutional and deconvolutional layers to create an end-to-end mapping from corrupted images to original ones. This method uses a convolutional auto-encoder architecture consisting of two parts: an encoder and a decoder. The encoder is a convolutional neural network that takes an image as input and outputs a set of features. The decoder then takes these features and reconstructs the image. Symmetric-skip connections are added between every corresponding encoder and decoder layer, allowing information to flow in both directions. This helps to reduce the amount of noise in the image and improve its overall quality. We link the convolutional and deconvolutional layers with skip-layer connections, which help to make the training of deep networks easier and improve restoration performance. These skip connections allow the signal to be back-propagated to lower layers and pass image details from convolutional layers to deconvolutional layers. As a result, the single model can deal with different levels of corruption.

1. Introduction

For nearly a decade, computer vision has seen the involvement of machine learning and deep learning methods to solve classical and complex problems: Image restoration, image denoising, inpainting, etc. Yet, it is being actively researched even in today's time, and with the advancement of technology and hardware, the research is simply going to increase further. In this project, we are trying to solve the image-denoising problem using convolutional auto-encoders.

Noise gets introduced in images for a variety of reasons ranging from sensor size and artifacts due to lens construction to environmental conditions, which are often unavoidable. Additionally, image-denoising plays an important role in a wide range of applications, such as image segmentation, image classification, object detection, etc. It is necessary to denoise an image because it might affect the accuracy of the results otherwise. A fundamental challenge of image-denoising is to remove the noise from the image and attempt to predict the original image. The recent success of Deep Neural Networks (DNN) and Convolutional Neural Networks (CNN) in computer vision tasks has prompted us to explore how to integrate CNNs to attempt to solve the image-denoising problem.

In this work, we have built a deep network architecture for image denoising, and the chain of layers consists of symmetric convolutional layers and deconvolutional layers. The convolutional layers extract the features and encode the primary components of the image. The convolutional layers also attempt to eliminate the corruptions. The deconvolutional layers are used to decode the image and recover the original details of the image. Furthermore, we are using skip connections between convolutional and deconvolutional layers, which divide the network into several blocks.

To better implement the denoising solution, we are using the Smartphone Image Denoising Dataset (SIDD) which contains notably more noise than images from advanced cameras. SIDD model contains image samples having different types of noise like the Gaussian, speckle, salt and pepper, etc. This makes our model diverse enough to remove any kind of noise from the given image unlike the baseline REDNet model that was trained on Berkeley Segmentation Dataset (BSD) which contains only Gaussian noise.

The remaining report is organized in the following manner. We explain the related work in Section 2. We then explain the dataset and its exploratory analysis in detail in Section 3. Section 4 and Section 5 discuss the methods and experiments, respectively. Finally, we present the limitations and conclusion in Section 6 and Section 7.

2. Related Work

For nearly a decade, researchers have been actively involved in investigating the various image denoising method. Deep learning has been a boon to this field and has acceler-

ated the research significantly as it has with other computer vision applications. A promising category of methods used for image denoising is the neural network based methods. The difference between neural network based methods and other methods is that they learn the parameters for denoising an image from the training data directly.

We found that image denoising methods can be classified into three broad categories [1]: Classical Denoising Methods, Transform Techniques in image denoising, and CNN based denoising methods.

Classical methods are subdivided into two types, namely Spatial Domain filtering and Variational denoising methods. Spatial domain filtering deals with applying linear and non-linear filters to the spatial domain of the images. However, variational methods are techniques that use a variational approach to reduce noise in an image. Image denoising is done using the knowledge of the image prior. In Sparse Representation [2], which is also a variational method, images are divided into patches and each patch is represented by a linear combination of several patches from an over-complete dictionary. Variational denoising methods are Total Variation Regularization, Sparse Representation, Low-Rank Minimization, etc.

The goal of transform techniques is to reduce the amount of noise while preserving the details in the image. Some commonly used techniques include median filtering, wavelet denoising, non-local means denoising, and total variation minimization. In transform techniques, images are first transformed into frequency domain, and then the noise is reduced using filters such as the Kalman filter or median filter [3]. Finally, the denoised image is transformed back into the spatial domain. Median filtering is a technique used to reduce noise by replacing each pixel with the median of the surrounding pixels. Wavelet denoising [4] is a technique that uses wavelets to reduce noise. Non-local means denoising and uses a non-linear approach to reduce noise. Lastly, total variation minimization is a technique that minimizes the total variation of the image.

CNN based [5] denoising techniques are being rapidly developed in recent times. CNN methods can detect the structure of noise and remove it without affecting the details in the image. The core idea is to optimize the loss function on a training set that contains degraded images. There are two major approaches in CNN based techniques: MLP models [6], and Deep learning [7] based denoising models. These approaches have been effective in removing noise from the images.

3. Data

Due to the advancements in camera technology, there has been a massive shift from DSLR and point-and-shoot cameras to smartphone cameras. Small cameras are also extensively used in robotic applications where cameras with large sensors are not feasible to use. However, due to the small aperture of the camera sensors, artifacts introduced due to improper lens, and surrounding lighting conditions, a significant amount of noise is introduced into the images. Since denoising smartphone images is an active area of research, we have used the Smartphone Image Denoising Dataset (SIDD) for our denoising project.

The SIDD dataset that we used consists of 160 images pairs of noisy and groudth truth clean image of size 256x256 having three channels, i.e., RGB. The 160 image pairs have been captured using smartphones from different companies: Apple, Google, Samsung, Motorolla, and LG. The total size of the dataset is approximately 6GBs. A characteristic feature of this dataset is that the images have been captured by adjusting the ISO levels and the shutter speed for each phone to introduce variations and noise. Additionally, the brightness levels have also been adjusted for each image.

To better understand the data, we performed an exploratory data analysis to understand the main characteristics of the dataset we are using. Essential information about the images has been encoded in the directory and image names. So we extracted characters from the directory name to identify the number of images per phone, as shown in Figure 1.

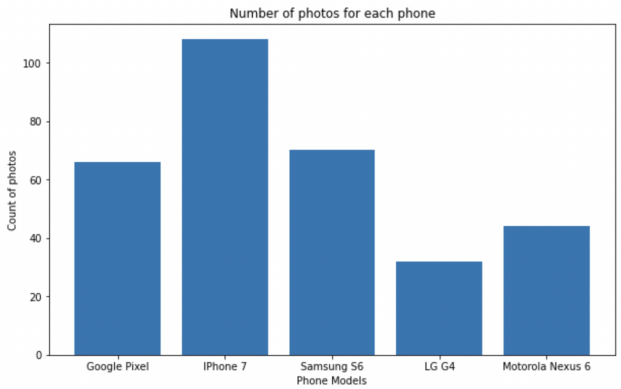

Figure 1. Number of images per phone

Since various ISO levels and shutter speeds have been used while capturing the images, we are mapping the number of images at each ISO level and shutter speed to understand how many images have been captured at a particular setting. These settings affect the color and brightness of the image. If a lower ISO setting is used, the image turns out to be darker and vice-versa. Similarly, if the shutter speed is low, the image turns out to be brighter and vice-versa.

As shown in Figure 2, iPhone has the highest number of images at ISO level 100. Furthermore, it is evident that most of the images have been clicked at low ISO settings, such as 100 and 800. As with Shutter Speeds, most of the images have been clicked at a shutter speed of 100, followed by 400

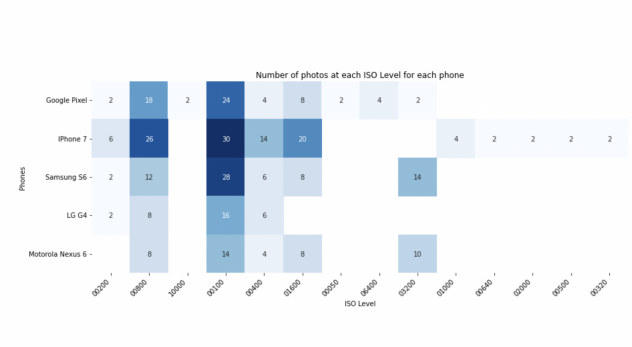

Figure 2. Number of photos at each ISO level per phone

and 800, as shown in Figure 3.

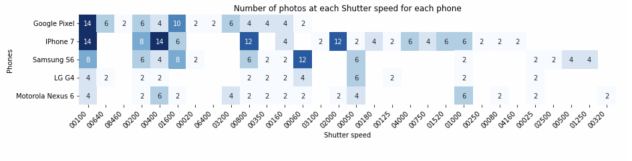

Figure 3. Number of photos at each shutter speed per phone

We also wanted to understand the mean and standard deviation on all three channels of all images, so we plotted them as shown in Figure 4. From figure 4, it can be seen that the lower to medium values of the mean denote that the images are predominantly of darker or medium brightness and not a lot of them are very bright. We also wanted to understand if the pixel intensity distribution for both the ground truth and the noisy images is similar. From Figure 5 and Figure 6, we can see that that they are similar and distributed almost normally for both the types of images.

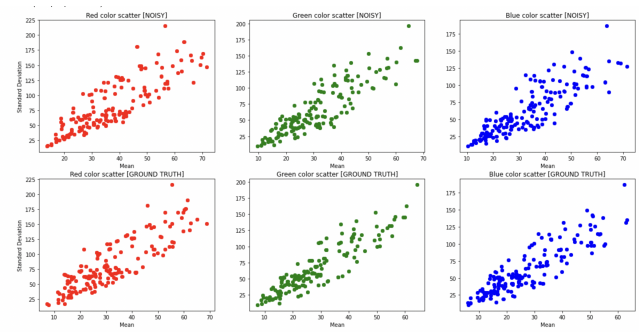

Figure 4. Mean and standard deviation of images across the three channels

Finally, we plotted the RGB channels of the noisy and the ground truth images as shown in Figure 7, to better understand the pixel insities of both the images. It wasn't surprising that the noisy images had a smooth distribution of pixels as compared to the the ground truth images. The smooth distribution of pixels hints that the camera failed to capture the color information for those pixels and hence it added some random value to those no-color pixels. The pixel intensities smoothed out because of these random values.

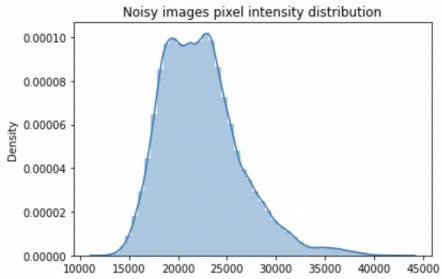

Figure 5. Pixel Intensity Distribution for Noisy Images

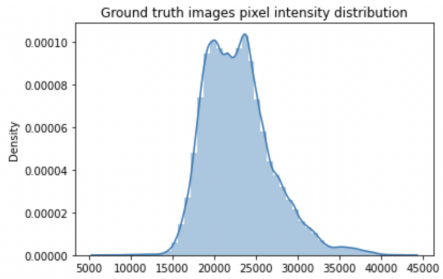

Figure 6. Pixel Intensity Distribution for Ground Truth Images

Data augmentation has been performed by flipping and rotating the images, changing the colors' contrast, hue, and saturation levels so that the model does not learn spurious features. The model takes a noisy and corrupted images

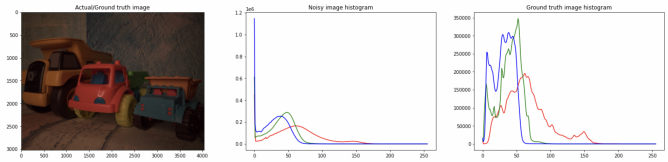

Figure 7. Pixel Intensity Distribution for Ground Truth Images

as an input and spits out the clean image as the output. The model has 15 convolutional layers followed by 15 deconvolutional layers. The loss function used is the mean squared error loss that computes the pixel wise Euclidean distance between the two images and Adam optimizer is used to converger the loss function.

The performance of the model has been evaluated using two ratios PSNR and SSIM. The Peak Signal to Noise Ratio (PSNR) is used as a quality measurement between the original and a compressed image. The higher the PSNR, the better the quality of the reconstructed image. Structural Similarity Index (SSIM) is another metric that quantifies image quality degradation caused by image processing.

Higher the value of these ratios, better the model in recovering the clean image from the noisy image. The performance of the proposed model and the baseline model have been compared.

4. Methodology

The baseline "RED-Net"-Residual Encoder Decoder Network has a Encoder-Decoder architecture with only 5 convolutional and 5 deconvolutional layers to extract features and upsample them. The baseline network is not very deep as deeper networks are difficult to train due to the problem of vanishing gradient. As we learnt in the class, more the number of convolutional layers in a network, better is the model in learning "true" image features. So, to learn these true features of an image, we increased the number of convolutional layers. Similarly, more deconvolutional layers were used to upsample these true features. Deeper networks tend to destroy the image details and may also learn image noise as one of the feature. Also, as mentioned earlier, they suffer from the problem of vanishing gradient. To solve these two problems, skip connections were added in between every corresponding convolutional and deconvolutional layer. The number of convolutional layers used for feature extraction was 15 and the number of deconvolutional layers to upsample the extracted features was also 15. Also, ReLu activation function was used after every layer to introduce non-linearity in the model which was absent in the baseline model. Also, batch normalization was performed after every layer so that the output of every layer is zero mean and unit standard deviation. Batch normalization is absent in the baseline model. Significant improvement in the evaluation metric was observed by increasing the the number of convolutional-deconvolutional layers, adding skip connections in between the layers, adding ReLu activation function after each layer, and performing batch normalization on the output of every layer. As learnt in the class, most of these techniques are quite recent and hence, they were implemented to improve the performance of the baseline model.

The baseline model was trained on the Berkeley Segmentation Dataset (BSD) which contains natural images of stationary objects. The main intent of the dataset is to act as a training data for the models designed to perform image segmentation. In the original paper, the model was artificially corrupted with Gaussian noise with a zero mean and standard deviation i.e. all the images in the data had Gaussian noise drawn from the same mean and same given standard deviation. This leads to poor performance of the baseline on test images drawn from the distribution with Gaussian noise of different mean and different standard deviation as the model implicitly learns the mean and standard deviation and is bias towards removing the noise with those parameters. To overcome this problem, we used a Smartphone Image Denoising Dataset (SIDD) which contains 160 image pairs of natural noisy images and ground truth images from 21 scene instances. The noisy images were captured by the cameras of different smartphones and hence, better resembles the noise distribution that may occur in an image naturally. Since the noise in the SIDD occurs due to various factors like the size of the camera lens, shutter speed, lightning conditions, etc., the dataset is not generated from some Gaussian noise distribution with fixed parameters. The SIDD not only has Gaussian noise with different mean and variance, but also has different types of noise such as the salt and pepper noise, speckle noise, Poisson noise, etc. It is important to note that these noises can occur naturally in an image and the deep learning model should be effective in removing them irrespective of their type. The ground truth images are captured using a DSLR camera.

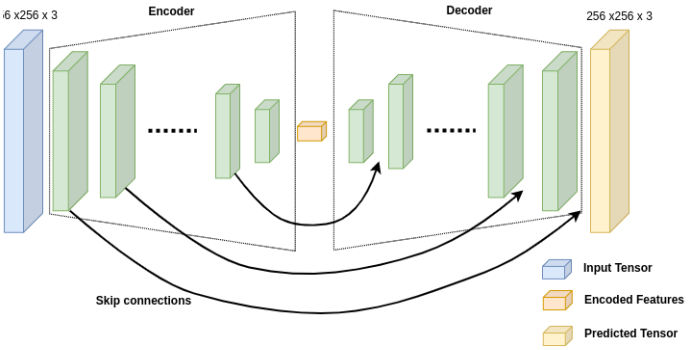

Figure 8. Proposed model architecture

5. Results

The model was trained for 48 epochs over the training set of 256 image pairs (noisy and ground-truth) with a cosine scheduled learning rate. The loss function used is the Mean Squared Error and the optimizer used is Adam with an initial learning rate of 0.001. The plot of loss vs. number of epochs can be seen in figure 9.

The PSNR and the SSIM value from the proposed model improved by 9.8% and 1.5% respectively over the baseline model. The absolute values can be seen in the table below.

Model	PSNR	SSIM
Baseline	33.49	0.9290
Proposed	36.76	0.943

5.1. Analysis on using the different dataset

We observed that the auto encoder convolutional layer neural network is better at learning the true features of images by eliminating the spurious noise features or details at

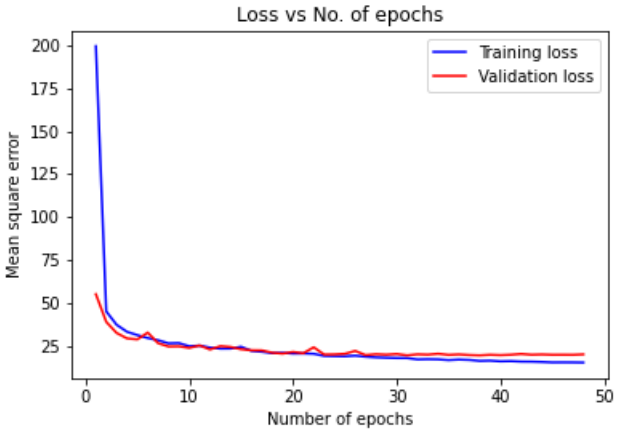

Figure 9. Loss vs. no. of epochs

the encoding stage. The architecture learns these true features by eliminating every noise features coming from different distributions like the Gaussian, salt and pepper, Poisson noise, salt and pepper noise. In this project, we observed that the basic architecture of the baseline model can be leveraged to learn the true features unlike the baseline model that was trained to remove noise coming specifically from the Gaussian distribution of given meand and standard deviation.

5.2. Visual Results

Some visual results are shown in Figure 12. The first observation is that our method better recovers the image details, as we can see from the third rows, which is due to the high PSNR we achieve by minimizing the pixel-wise Euclidean loss. Moreover, we can observe that our network obtains more visually smooth results. This may due to the testing strategy which average the output of different orientations.

6. Experiments

In this section, we first provide some experimental results and analysis on different parameters, including filter number, filter size, training patch size and skip connection step size, of the network. Then, evaluation of image denoising task is conducted and compared against a few existing state of-the-art methods. Peak Signal-to-Noise Ratio (PSNR) and Structural SIMilarity (SSIM) index are calculated for evaluation. For our method, we implemented two versions: one with 10 convolutional and deconvolutional and the other with 15 convolutional and deconvolutional layers.

We have observed that deeper networks tend to achieve better image restoration performance. We also carried out image denoising experiments on three folds: (a) filter number, (b) filter size, (c) training patch size and (d) step size of skip connections, to show the effects of different parameters. For different filter numbers, we fix the filter size as 3×3, training patch size as 256×256 and skip connection step size as 1. Different filter numbers of 32, 64 and 128 are tested, and the PSNR values recorded on the validation set during training are shown in Figure 10. To converge, the training iterations for different number of filters are similar, but better optimum can be obtained with more filters. However, a smaller number of filters are preferred if a fast testing speed is desired.

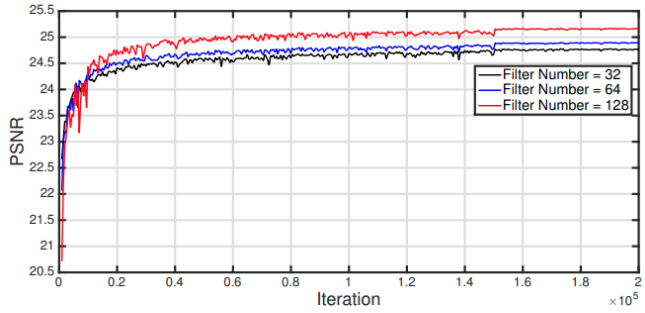

Figure 10. PSNR values on the validation set during training with different number of filters.

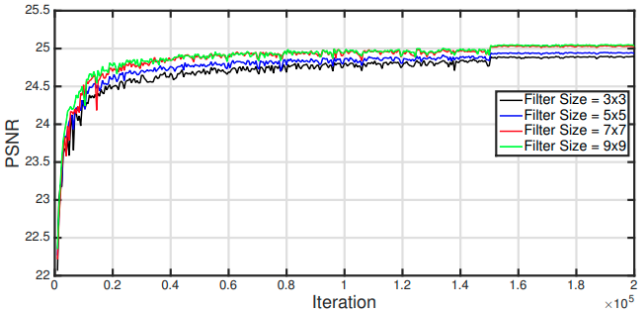

Figure 11. PSNR values on the validation set during training with different size of filters.

For the experiments on filter size, we set the filter number to be 64, training patch size as 256×256, skip connection step size as 1. Filter size of 3×3, 5×5, 7×7, 9×9 are tested. Figure 11 shows the PSNR values on the validation set while training. It is clear that larger filter size leads to better performance. Different from high-level tasks which favor smaller filter sizes, larger filter size tends to obtain better performance in low-level image restoration applications.

However, there may exist a bottle neck as the performance of 9×9 is almost as the same as 7×7 in our experiments. The reason may be that for high-level tasks, the networks have to learn image abstraction for classification, which is usually very different from the input pixels. Larger filter size may result in larger respective fields, but also

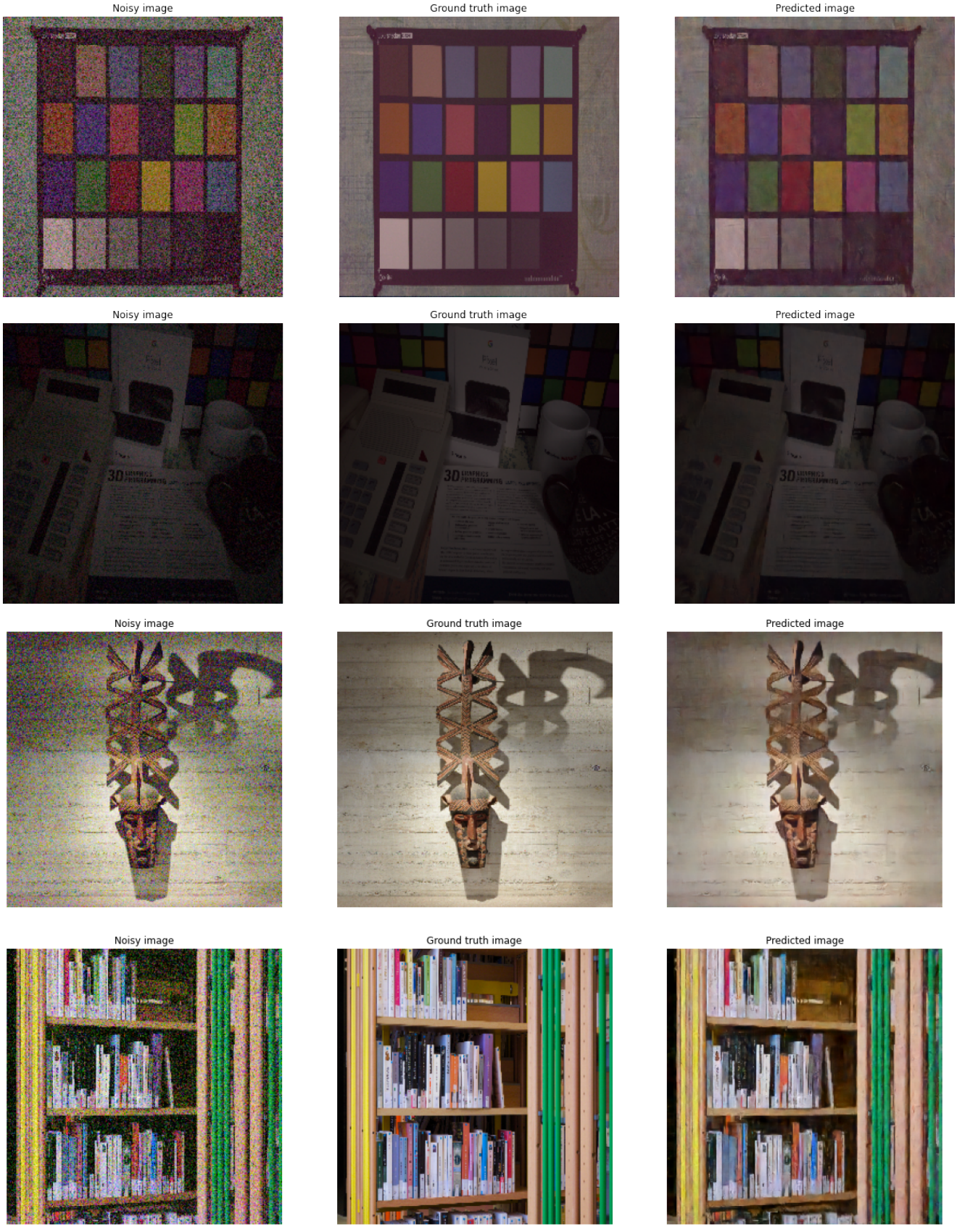

Figure 12. Noisy image, ground truth, and predicted image from the proposed model

made the networks more difficult to train and converge to a poor optimum. Using smaller filter size is mainly beneficial for convergence in such complex mappings.

In contrast, for low-level image restoration, the training is not as difficult as that in high-level applications since only a bias is needed to be learned to revise the corrupted pixel. In this situation, utilizing neighborhood information in the mapping stage is more important, since the desired value for a pixel should be predicted from its neighbor pixels. However, using larger filter size inevitably increases the complexity (e.g., filter size of 9×9 is 9 times more complex as 3×3) and training time. For the training patch size, we set the filter number to be 64, filter size as 3×3, skip connection step size as 1. T

We also provide the experiments of different step sizes of shortcuts, as shown in Figure 12. A smaller step size of shortcuts achieves better performance than a larger one. We believe that a smaller step size of shortcuts makes it easier to back-propagate the gradient to bottom layers, thus tackle the gradient vanishing issue better. Meanwhile, a small step size of shortcuts essentially passes more direct information.

7. Limitations

Due to increased number of convolutional and deconvolutional layers compared to the baseline model, the sampling time has increased i.e. the time required to remove noise from the test image has increased because of the increase in the number of operations.

The computation power and memory required to store the weights have increased due to the increased layer of convolutional and deconvolutional layers, hence, the model cannot be deployed in mobile phones and computers that do not have high memory and computation power.

8. Conclusion

In this project, we have developed a deep encoding and decoding framework for image restoration. Convolution and deconvolution are combined, modeling the restoration problem by extracting primary image content and recovering details.

More importantly, using skip connections in deep layers helps in recovering clean images and tackles the optimization difficulty caused by gradient vanishing, and thus obtains performance gains when the network goes deeper. Experimental results and our analysis show that our network achieves better performance than the baseline "REDNet" for image denoising and the proposed architecture can be deployed for other image restoration task like image super-resolution, JPEG deblocking and image inpainting with minimal changes.

References

[1] Abdelrahman Abdelhamed, Lin S., Brown M. S. "A High-Quality Denoising Dataset for Smartphone Cameras", IEEE Computer Vision and Pattern Recognition (CVPR), June 2018.

[2] Xiao-Jiao Mao, Chunhua Shen, Yu-Bin Yang "Image Restoration Using Convolutional Network"

[3] Fan, L., Zhang, F., Fan, H. et al. Brief review of image denoising techniques. Vis. Comput. Ind. Biomed. Art 2, 7 (2019)

[4] Zhang KB, Gao XB, Tao DC, Li XL (2012) Multi-scale dictionary for single image super-resolution. In: Abstracts of 2012 IEEE conference on computer vision and pattern recognition. IEEE, Providence, pp 1114–1121.

[5] Gonzalez RC, Woods RE (2006) Digital image processing, 3rd edn. Prentice-Hall, Inc, Upper Saddle River

[6] Liu T (2010) The nonlocal means denoising research based on wavelet domain. Dissertation, Xidian University

[7] Chen YY, Pock T (2017) Trainable nonlinear reaction diffusion: a flexible framework for fast and effective image restoration. IEEE Trans Pattern Anal Mach Intell 39(6):1256–1272.

[8] Vincent P, Larochelle H, Bengio Y, Manzagol PA (2008) Extracting and composing robust features with denoising autoencoders. In: Abstracts of the 25th international conference on machine learning. ACM, Helsinki, pp 1096–1103.

[9] Zhang K, Zuo WM, Chen YJ, Meng DY, Zhang L (2017) Beyond a Gaussian denoiser: residual learning of deep CNN for image denoising. IEEE Trans Image Process 26(7):3142–3155.