IMPLEMENTATION AND EVALUATION OF A DENOISING COVOLUTIONAL NEURAL NETWORKS ALGORITHM

TNM089, IMAGE TECHNOLOGY

Rasmus Hogslätt
rasho692@student.liu.se

Wednesday 4^{th} October, 2023

Abstract

This report explores the implementation and evaluation of the Denoising Convolutional Neural Networks (DnCNN) model for image denoising, compared to traditional Gaussian kernel-based denoising methods. DnCNN utilizes residual learning and batch normalization to predict and subtract residual noise from images. The model was trained on a generated dataset of clean and noisy image pairs using a CPU, due to resource constraints, limiting the scope of training. Evaluation utilized standard denoising metrics on test images subjected to Gaussian and Salt and Pepper noise. The results show the trained DnCNN models, performed comparably to traditional methods. The findings indicate DnCNN's potential for superior denoising performance with optimized training and suggest further exploration with improved computational resources and different architectures for improved denoising performance.

1 Introduction

Image denoising is a fundamental task in image processing, which aims to remove noise from images while preserving the underlying image structure. Image noise can be caused by a variety of factors, such as sensor noise, quantization noise, and transmission noise. Gaussian noise is one of the most common types of image noise, which can be modeled as a random variable with a normal distribution.

Traditional denoising methods are usually utilizing Gaussian kernels and convolutional operations. They average the values in the area of the kernel, which produces a smoothing of the image. This smoothing effect removes noise, but does so by introducing blur. Methods exist where adaptive kernel sizes can be applied to different part of the image based on regional differences noise levels. This requires a notion of measuring what part of the image is more noisy than another, which may not always be a trivial task.

Deep learning-based denoising methods have recently emerged as a new class of denoising algorithms that have achieved state-of-the-art results. Many deep learning-based models have drawbacks such as being effective only on a given noise level or certain types of images. A new method, DnCNN (Deep Convolutional Neural Network for Image Denoising), was proposed by a group of reasearchers in 2016 [1] which became one of the most popular deep learning-based denoising algorithms. DnCNN models are trained on a large dataset of noisy and clean images, where the noise levels differ. They then learn to denoise images by predicting the residual noise between the noisy image and the clean image, even for random levels of noise which previous deep-learning based models did not.

This report aims to implement the proposed DnCNN model and compare its performance to more traditional methods of denoising. Due to limited computational resources, only a few smaller models are trained and evaluated.

2 Background

The DnCNN model is a deep learning architecture designed explicitly for image denoising. Its fundamental difference to other deep learning based models is the utilization of residual learning and batch normalization. Also, DnCNN uses convolutional neural networks (CNN), which are suitable for image data due to their abitiliy to preserve spatial relationships within the image.

2.1 Convolutional Neural Networks

Convolutional neural networks, CNNs, consist of multiple layers, each designed to extract different features from the input image. Convolutional layers, employ filters or kernels that slide over the image, performing element-wise multiplication followed by summation to generate a feature map. The convolution operation can be represented by the equation:

$$F_{ij} = \sum_{m} \sum_{n} I_{(i-m)(j-n)} \cdot K_{mn} \tag{1}$$

where F is the feature map, I is the input image, and K is the kernel.

These layers are adept at capturing local patterns and spatial hierarchies in images, making them instrumental for tasks like image denoising. The same convolutional operations are employed in traditional denoising methods, but the size of the filter or kernel is usually fixed. The DnCNN model, after being trained, learns to adapt the size of the filter or kernel in order to minimize unnecessary distortion.

2.2 Residual Learning

One of the innovative aspects of DnCNN is its utilization of residual learning. Instead of denoising the image directly, DnCNN seeks to predict the residual noise which, when subtracted from the noisy image, results in a denoised image. This approach simplifies the learning process, as learning a residual map is often easier than learning the denoised image outright. The residual learning can be formulated as:

$$D(x) = x - F(x) \tag{2}$$

where $D(x)$ is the denoised image, x is the noisy image, and $F(x)$ is the residual noise learned by the network.

2.3 Training DnCNN

Training a DnCNN model requires a dataset comprising pairs of clean and noisy images. The model learns to minimize the difference between the predicted residual noise and the actual noise during training. A loss function, typically Mean Squared Error (MSE), gauges the difference between the predicted and actual noise, guiding the optimization process to tune the model parameters for better performance. The MSE loss function is given by:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (F(x_i) - (x_i - y_i))^2 \tag{3}$$

where N is the total number of training samples, x_i is the noisy image, y_i is the clean image, and F represents the network. In Figure **??** is an example of the subset of the training to train the model implemented in this report.

2.4 Batch Normalization and ReLU Activation

DnCNN incorporates Batch Normalization after each convolutional layer, except for the last one. Each layer will thus be normalized to a range, resulting in cheaper, faster and more accurate training. The Rectified Linear Unit (ReLU) activation function is employed to introduce non-linearity into the model, aiding in learning complex patterns with faster training than many other activation function, such as the sigmoid function.

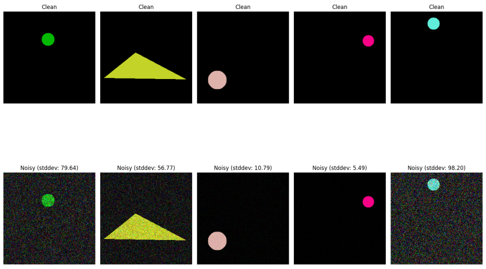

Figure 1: Upper row: Clean images. Lower row: Gaussian noise added with random standard deviation.

2.5 Performance Evaluation

Performance evaluation is crucial to ascertain the efficacy of DnCNN in comparison to traditional denoising methods. Common metrics like Peak Signal-to-Noise Ratio (PSNR), Mean Square Error (MSE) and Mean Absolute Error (MAE) are employed to quantify the denoising performance. PSNR is defined as:

$$\text{PSNR} = 20 \cdot \log_{10}\left(\frac{\text{MAX}_I}{\sqrt{\text{MSE}}}\right) \tag{4}$$

where MAX_I is the maximum possible pixel value of the image.

3 Results

Images in Figure 5 through 6 show of three models on two different test images with various standard deviations. They are compared to the best PSNR given by the traditional Gaussian kernel denoising method with kernel sizes 3x3, 5x5, 7x7, 9x9 and 11x11. Similar comparisons was performed in Figure 7 where salt and pepper noise was added to the test image. The models were still trained on Gaussian noise. Further, the three models tested were trained on CPU, with validation loss given in Figures 2 through 4 respectively. Images have been re-scaled and interpolated to fit the format of a report, thus, they may not be accurately depicted.

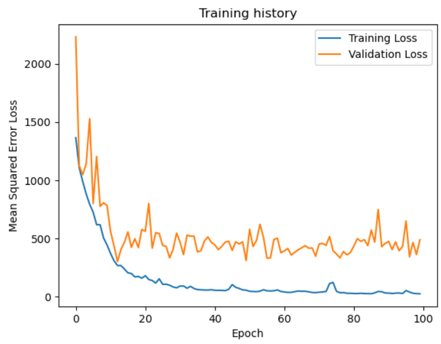

Figure 2: Validation loss during training model with training images of size 64x64 pixels for 100 epochs.

Figure 3: Validation loss during training model with training images of size 128x128 pixels for 40 epochs.

4 Discussion

This section provides an analysis of the results obtained from the experiments conducted, and compares the performance of the DnCNN model with traditional denoising methods. The key points discussed in this section include the effectiveness of deep learning-based denoising, computational requirements, and potential areas for further research.

Figure 4: Validation loss during training model with training images of size 256x256 pixels for 40 epochs.

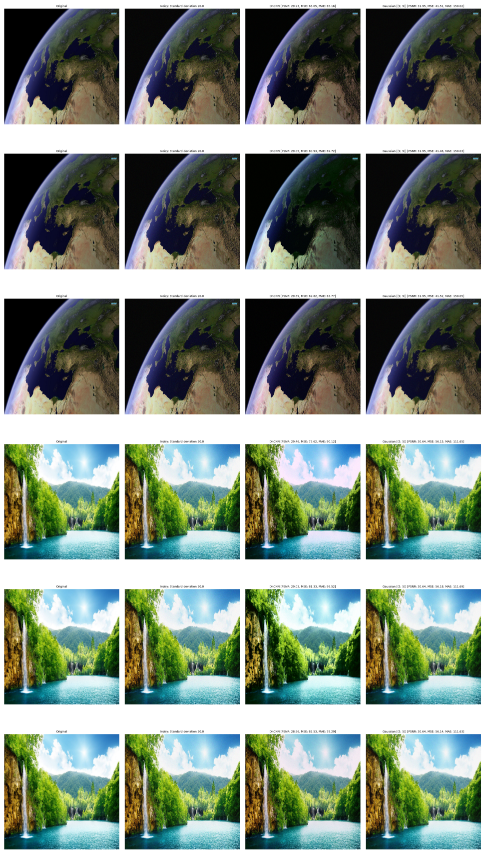

Figure 5: Results for three models trained on 64, 128 and 256 images respectively. Test images have Gaussian noise with standard deviation of 20.

4.1 Performance Comparison

The results shown in Figures 5 and 6 indicate that this DnCNN model performs similar metrics as regular Gaussian kernels applied to the test image. Although the metrics are overall rather similar, the DnCNN model typically retained more of the details, not blurring the image as Gaussian kernels do.

As seen in Figure 7, the DnCNN model performed slightly better than the Gaussian kernel method, despite being trained on Gaussian noise. The difference was not significant, but the difference may indicate that a more well trained model could outperform traditional methods on various different types of noise.

4.2 Computational Resources

The limitation of computational resources impacted the scope of this study significantly. Training larger models or training for more epochs could potentially improve the denoising performance of DnCNN further. Moreover, the training was carried out on a CPU, which is considerably slower compared to a GPU. Future work with better computational resources could explore these aspects in depth.

4.3 Model Training and Evaluation

The validation loss plots shown in Figures 2 through 4 provide insight into the training process. The validation loss tends to decrease with the number of epochs, although the 256x256 model's validation loss, seen in Figure 4, occasionally spiked, indicating that the training was not optimal. This might be due to overfitting or inoptimal choice of model parameters.

4.4 Adaptibility

In order to fairly compare the DnCNN model to regular Gaussian kernel convolution, several kernel sizes were tested and only the best was compared to the DnCNN model. This may have given an unfair advantage, not highlighting the adaptibility of the DnCNN model. It did provide similar results regardless of the

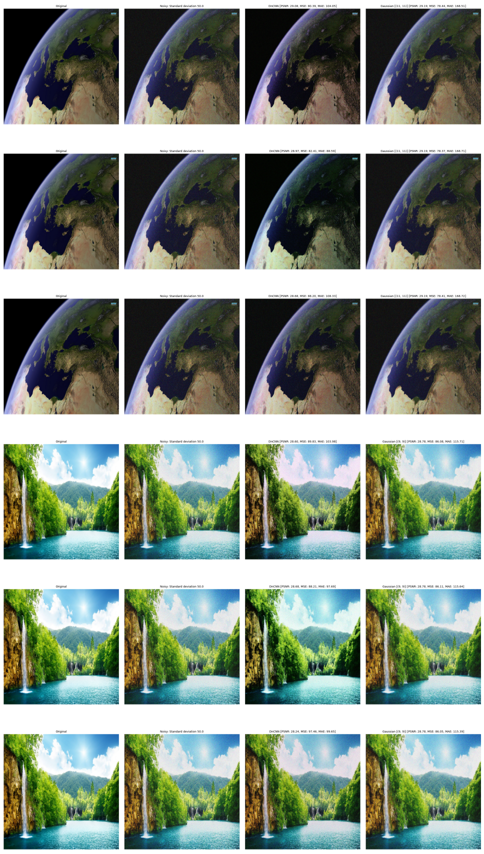

Figure 6: Results for three models trained on 64, 128 and 256 images respectively. Test images have Gaussian noise with standard deviation of 50.

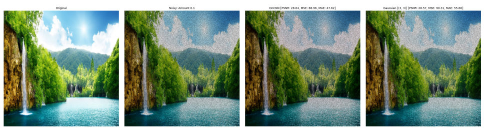

Figure 7: The model trained on 256x256 images compared to Gaussian convolutional method on Salt and Pepper noise.

standard deviation of the Gaussian noise that was added, thus reducing runtime costs compared to traditional methods that usually need to, in addition to the convolutional operations, determine the best kernel size for various regions in the image.

4.5 Future Work

Future work could involve experimenting with different architectures, loss functions, or training parameters to improve the denoising performance. Additionally, applying DnCNN to other types of noise beyond Gaussian and Salt and Pepper noise could be valuable. The DnCNN model was proposed in the paper [1] in 2016. Thus, how this model stands in comparison to any more recent models may also be explored further.

5 Conclusion

DnCNN is a powerful deep learning-based denoising algorithm that can outperform traditional denoising methods based on Gaussian kernels on a variety of image datasets. DnCNN is able to remove noise while preserving image details, making it a valuable tool for a wide range of image processing applications. Unfortunately, only being able to train the model on a CPU limited the achieved performance of the trained models. It is possible that models trained for longer or on larger datasets would have achieved better performance.

References

[1] Yunjin Chen Deyu Meng Lei Zhang Kai Zhang, Wangmeng Zuo. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *arXiv preprint arXiv:1608.03981*, 2016.