Learned Convolutional Denoising

Dash Bodington
Stanford University
dashb@stanford.edu

1. Introduction

State of the art denoising algorithms, while well designed and effective, are generally not as intelligent as they could be. Methods such as BM3D and nonlocal mean algorithms use collaborative filtering between similar blocks in an image, but they still cannot consider the content of an image in the sense of recognition [2]. With the recent trend toward deep learning in computer vision with convolutional neural networks, and increasing development of libraries and hardware for its acceleration, the time is right to apply some elements of computer vision to denoising to create more intelligent denoisers which may be able to perform better.

This project begins to develop a more intelligent denoising method than the current state of the art, in the form of a convolutional network, by applying machine learning techniques to learn various filters which are used for feature detection and reconstruction.

2. Related Work

Learned denoising is not a new concept. Attempts at learned denoising with various types of neural networks have been made with good results in the past, but many current models have limitations.

Learned denoising algorithms which use neural networks are often restricted to processing images of a single pixel dimension because every image dimension requires a different number of connections for neurons in the model, and only one set of connections can be learned. Purely convolutional neural networks, neural networks with fixed numbers of local connections and shared weights have a practical advantage over any network with fully-connected neurons [1,4,5].

While convolutional neural network denoising methods exist, modern learned denoising methods are designed almost exclusively to denoise monochromatic images. While any of these methods could be expanded to color images by processing each color channel separately, they ignore correlations between features in the various channels which may aid in denoising.

Beginning with a monochromatic convolutional model, this project develops a convolutional denoising method which can be trained on three color channels simultaneously to take advantage of feature color correlations which may exist, and which can be applied without regard for image dimensions.

3. Model Approach

All denoising methods presented are a variation of a simple model comprised of recognition filters and blurring filters. A detection system composed of linearly applied detection kernels and nonlinear steps learns to pick from a set of blurring kernels which are used to denoise images. Both sets of filters are learned from a random initialization through iterative training.

3.1. Model Development

The models implemented, whether color or monochromatic, are trained in the same way. Filter weights are randomly initialized, trained and optimized with stochastic gradient descent, and tested on a set of images independent from the training data. While multiple model types were implemented, those presented contain only one set of filters in the detection and blurring methods.

3.2. Model Training Details

After the model is defined, it is applied to noisy training images where the noiseless ground truth is known. By calculating the model gradients with respect to the error in image reconstruction, the model is iteratively optimized with stochastic gradient descent using the following update rule for each image.

$$W_{n+1} = W_n + \alpha\gamma^b((1-m)\frac{de_i}{dW_n} + m\frac{de_{i-1}}{dW_{n-1}}))$$

α is the learning rate,γ is the learning rate decay factor, b is the training epoch number, m is the update's momentum, e is the image reconstruction error, i is the image number, and n is the training iteration number.

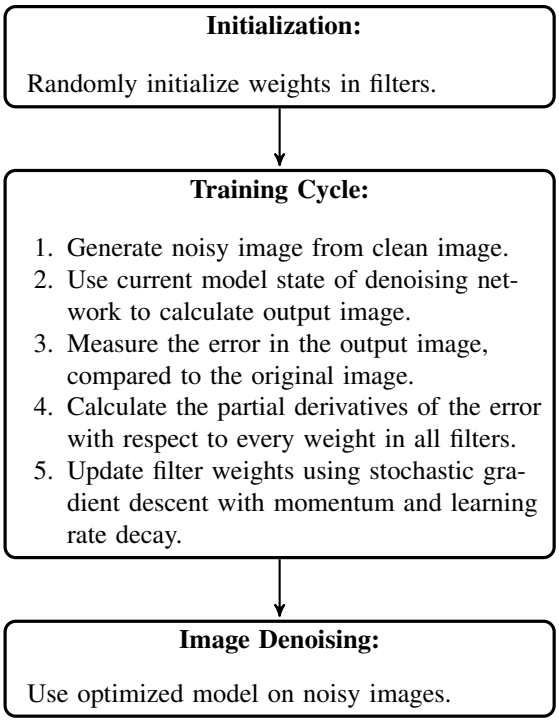

Figure 1. A diagram of the system simulated in this project. Training is the most time consuming process.

Because of the costly iteration and image operations involved in training, all gradients must be formulated analytically to be calculated in a reasonable amount of time, and some operations are performed on a GPU for further acceleration. Calculating numerical derivatives extends the training time by three orders of magnitude.

For this project the only measure of error used is the RMS error between images. This optimizes the model for RMS error, but it would also be possible to use a perception-based image error metric such as PSNR-JND to optimize for more eye-pleasing results, or a metric like the RMS difference in image gradients, which may improve sharpness [3].

3.3. Model Assumptions and Noise

Because the noise used in this project was known to be zero-mean, it was assumed that the mean pixel value of an image or an image channel should not change before and after denoising. This is an important assumption because there is no constraint that the weights in each blurring kernel should sum to 1, so scaling a denoised image compensates for the denoising filters. Depending on what the detection kernels match in a test image, denoising could either increase or decrease the amount of energy in an image by picking more blurring kernels which brighten an image, or more blurring kernels which darken an image.

For the most part, additive gaussian noise was used in training and testing the models. Uniformly distributed noise was experimented with, but not focused on, and Poisson distributed noise should be tested in the future. When an alternative colorspace, such as YCbCr was used for training and testing, all noise addition and testing error calculation was performed in the RGB domain, as it would be recorded by a camera sensor.

3.4. Training and Testing Data

All models were trained on a set of 98 randomly selected photographs taken by the author. Training images were resized to a maximum side dimension of 256 pixels, while maintaining aspect ratio, and pixel values were linearly scaled from their 0 to 255 values to double precision values ranging from 0 to 1.

The test image set is comprised of 38 randomly selected photographs, independent of and not overlapping the training set, which were scaled and resized in the same manner as the training data.

3.5. BM3D

BM3D (Block Matching and 3D Filtering), is considered to be a state of the art denoising algorithm. It performs denoising by matching similar blocks in a noisy image and applying collaborative filtering to the similar blocks. BM3D, or CBM3D (Color BM3D) is used as a denoising algorithm baseline in this project.

4. Evaluation

One of the most important factors in the results was the time-accuracy ratio for models. Because training complicated or large models is very time consuming, the best results are generally from small models which could be trained relatively quickly. Models which took a long time to train simply required too much hand-tuning of training variables and waiting, to develop well in the project timeframe.

4.1. Basic Monochromatic Model

The most basic model applied in this project is a system of two sets of filters applied to monochromatic images. This model consists of one set of detection filters, and one set of blurring filters where each set is of the same size, and there is a one-to-one correspondence between filters in each filter set. When denoising a single pixel of an image, every detection filter was applied to the neighborhood of the pixel, and the best fitting detection filter's corresponding blurring filter was applied to this same neighborhood to calculate the new value for the pixel. By iteratively learning both sets of filters, a well-performing model was developed. Image padding was performed by replicating border elements so that image edges could be denoised.

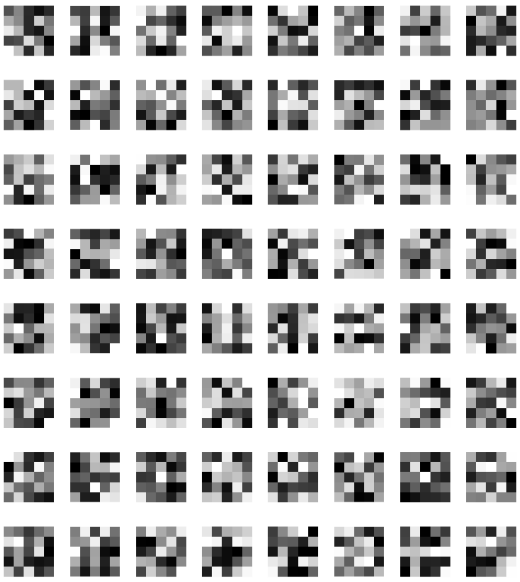

Figure 2. This set of filters is used to recalculate the pixel value at its center during denoising. Which filter is used for a given pixel depends on the fit of a second set of filters to the pixel's neighborhood.

At the point where the monochromatic model was set aside to focus on color models, it was able to reduce the RMS error between a denoised noisy image and a noiseless original by a factor of 3. BM3D performed better, decreasing the RMS error by a factor of 5. This monochrome model used sets of 64 5x5 filters for detection and blurring.

4.2. Color Models

With the success of the monochromatic model, further development resulted in the color model, which has an identical structure of detection and blurring filters, but each filter has a depth of 3 instead of a depth of 1. For the purposes of testing and training, noise was added to each RGB color channel with the same standard deviation. The resulting model denoises impressively, and even outperforms BM3D when high noise levels prevent good block matching.

The learned color denoising method performs best when high noise prevents BM3D from doing effective box matching. Unlike BM3D, learned denoising forces blurring to occur even when detectors may not be able to detect features as intended, which reduces noise. As the noise level decreases, BM3D outperforms the learned method slightly when RMS error is used as a metric, however, at this image size the results of BM3D are often sharper.

Results presented are from an RGB model for filtering,

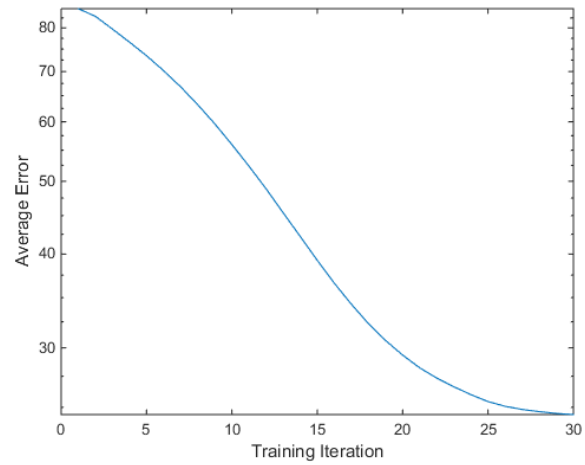

Figure 3. Training a color model to near asymptotic convergence of the training error takes approximately 1 hour with 128 5x5x3 filters in each filter set.

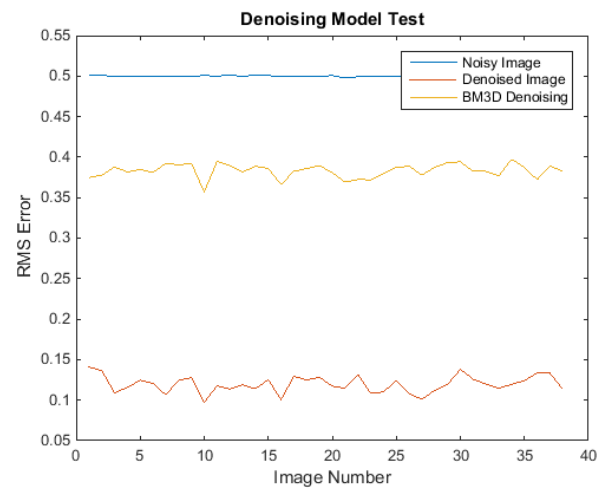

Figure 4. RMS error, the standard deviation of the image from the original, can be improved better than CBM3D when noise is very high. This learned model contains 128 5x5 filters per set.

but the same model trained and tested on YCbCr images showed no significant difference in performance.

5. Discussion

As the results show, the color model with filter sets containing 128 5x5x3 filters can often nearly match and outperform BM3D when it comes to RMS error, but subjective results are more mixed. With the pixel dimensions (256 maximum) of the images shown in the figures, the learned method appears to blur images and make them patchier than the BM3D result. This is because the learned method, which by its definition attempts to minimize the RMS error, is not sensitive to blurring and patchiness and does not take

Figure 5. As noise decreases, BM3D begins to match and slighty outperform the learned model.

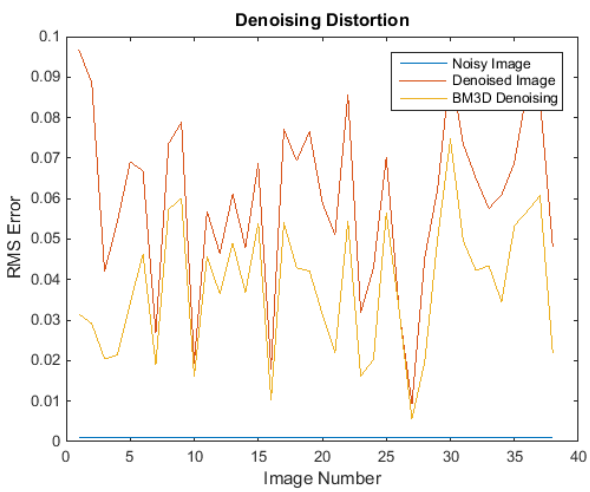

Figure 6. When there is no noise in the input image, both models cause distortion, with BM3D being slightly better.

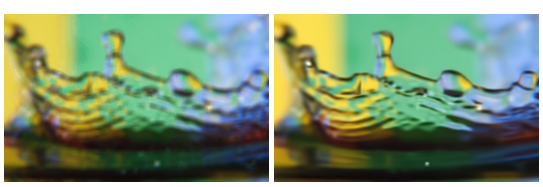

Figure 7. When there is no noise in the input image, the learned denoising result (left) appears more blurred than the BM3D result (right). These images correspond to test image 6 in Figure 6.

advantage of the collaborative filtering possibilities of the image. The fixed filter size is also more apparent when images are smaller. The results of BM3D are often sharper and smoother, but this is no guarantee of accuracy, and prevents the model from working on high-noise images.

This patchiness-blurring relationship is also a function of

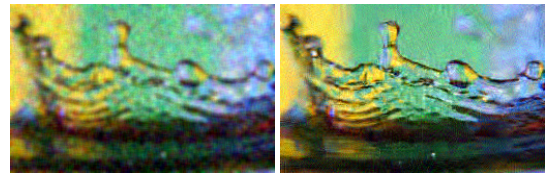

Figure 8. When the noise has a standard deviation of one quarter the pixel range, the learned denoising result (left) and the BM3D result (right) have very close error, but the BM3D result appears sharper. These images correspond to test image 6 in Figure 5.

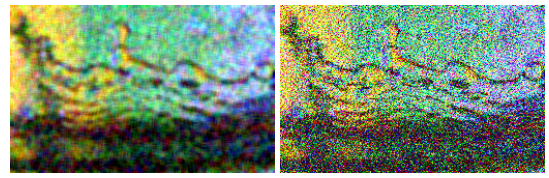

Figure 9. When the image noise is high enough, BM3D (right) becomes ineffective.These images correspond to test image 6 in Figure 4.

image size. Because filters are fixed at 5x5 pixels, the apparent patchiness and blur of the learned method decreases as pixel density increases, making subjective results better. BM3D, on the other hand, appears to perform worse as image size grows. Possibly due to fixed numbers of blocks, BM3D gains patchiness as pixel density increases.

In general, the learned denoising method performs well, and with further development and adjustment, may be able to match BM3D.

6. Future Work

This project has demonstrated the potential for learned convolutional denoising, but the concept requires a great deal of refinement. Several normalizations or assumptions, such that the noise is zero-mean, must be built into the model which can be expanded significantly.

The first modification to the model should be to encorporate larger filters. Whether this should be a multiscale method (containing trained detectors and blurrers of varying sizes), or whether it is possible to train a set of very large filters which can be resampled to different sizes according to image dimensions, would be an interesting investigation.

Adding another layer, or multiple layers to the network could also be effective. Having nonlocal connections or methods in deeper layers of the network would allow for an equivalent to collaborative filtering, which may be better learned than designed.

Once a high capacity model is developed and accelerated enough to train quickly, different error metrics and metric ensembles should be tested to see what yields the best output from the point of view of error, sharpness, and subjective appearance.

Figure 10. When the noisy image (top) is large (12 megapixels), BM3D (bottom) becomes patchier than the learned method (middle).

References

[1] H.C. Burger, C.J. Schuler, and S. Harmeling. Image denoising: Can plain neural networks compete with bm3d? In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2392–2399, June 2012.

[2] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising with block-matching and 3d filtering. In *Electronic Imaging 2006*, pages 606414–606414. International Society for Optics and Photonics, 2006.

[3] G. Dimauro. A new image quality metric based on human visual system. In *Virtual Environments Human-Computer Interfaces and Measurement Systems (VECIMS), 2012 IEEE International Conference on*, pages 69–73, July 2012.

[4] Viren Jain and Sebastian Seung. Natural image denoising with convolutional networks. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 769–776. Curran Associates, Inc., 2009.

[5] Junyuan Xie, Linli Xu, and Enhong Chen. Image denoising and inpainting with deep neural networks. In *Advances in Neural Information Processing Systems*, pages 341–349, 2012.