EE-559: Deep Learning Miniproject 1 Report

Mahammad Ismayilzada, Batuhan Faik Derinbay, Maciej Styczen

May 27, 2022

Abstract

In this report, we present our Gaussian nOise Reducing Autoencoder (GORA) model trained using convolutional and transposed convolutional layers that can effectively learn the noise distribution of an image just by looking at corrupted images and restore the images with high quality. Our model GORA, achieves a PSNR score of 25.3470.

1 Introduction

Traditionally, signal reconstruction from noisy data has been based on a simple approach of learning to map corrupted observations to the unobserved clean versions. However, recent work [LMH$^+$18] shows that it is possible to achieve the same goal without using any clean target images. More specifically, we can build a Convolutional Neural Network model that can learn the distribution of noise present in the image just by looking at 2 noisy images. In this project, we reproduce the work done in [LMH$^+$18] using the popular Pytorch [PGM$^+$19] deep learning library and in addition, we also apply standard data augmentation techniques to improve our model. Below we walk through our model architecture and data augmentation methods.

2 Methodology

2.1 Model Architecture

In this section, we outline our model architecture in more details. Our model architecture is based on the popular fully convolutional neural network architecture UNet [RPB15] and has been adapted for our use with few modifications from the unofficial Pytorch implementation [1] of the Noise2Noise paper [LMH$^+$18]. In particular, we reduced the model size from eight blocks to six blocks and decreased the number of channels by four folds according to the pruning schema introduced in the later UNet++ [ZSTL19] paper. We also tried several architectural improvements introduced by similar models such as Eff-UNet [BIGT20] and simple blocks such as batch normalization however as not much performance increase was observed, we decided to keep a simple UNet-like architecture. For further details of our model, you can refer to `utils/gora.py`.

2.2 Data Augmentation

In order to help reduce quick overfitting, we introduced augmentation to our data. Using PyTorch's own `torchvision` package, we integrated eight image transformations. These are namely, vertical flip, horizontal flip, brightness, contrast, gamma, hue, saturation adjustments and rotation which are represented in Figure 1.

Every noisy image pair was applied with either none, one or a combination of these selected transformations with a 50% chance. The parameters for the brightness, contrast, gamma, hue, saturation adjustments are chosen both experimentally and visually in order to increase the variance while respecting the visual representation of the image. These are one of 0.75, 0.90, 1.10 and 1.25 for the brightness, contrast and gamma, whereas 0.05 for hue and saturation. When it comes to the rotations, only one of 90, 180 and 270 degrees was applied with the same probability.

[1] https://github.com/joeylitalien/noise2noise-pytorch

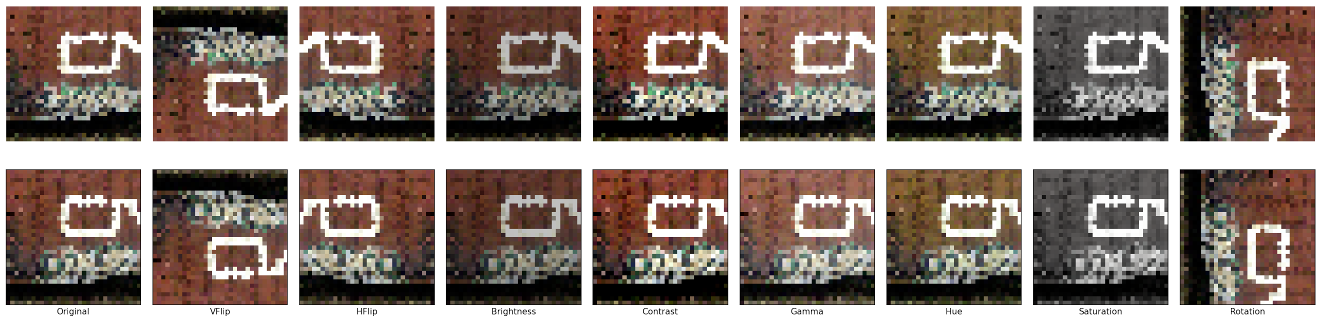


Figure 1: List of Augmentations (Rows are noisy image pairs)

We tested several combinations of our augmentations, compared it to our baseline model GORA without any augments and reported its results in Table 1. Code for our augmentation methods can be found in `utils/augmentation.py`.

3 Experiments

3.1 Setup

As a noisy dataset, we are given a training set of 50,000 and a validation set of 1000 noisy image pairs. Each image has 3 channels and is of size 32x32. Each pixel is a value between 0 and 255 and we found that not normalizing it by dividing with 255 works better than normalizing our inputs to a $[0-1]$ range.

The training setup consisted of a workstation containing one NVidia A5000 GPU with 24GB of VRAM that was rented by the hour. Initially our experiments used a very large batch size, precisely 2^{14} (16384). However, after further experimentation, we observed that smaller batch size, around the magnitude of $2^5 - 2^7$, along with small learning rates resulted in better PSNR scores, albeit longer training times.

3.2 Hyperparameter Tuning

In order to find the hyperparameters that result in the best generalizable model, we conducted a thorough hyperparameter search. Initially, we tried several fundamental configurations such as L1 loss vs L2 loss as our loss function and SGD vs Adam as our optimizer. Eventually, we decided to keep L2 as our loss function and Adam as our optimizer in order to provide better comparability between peers, the benchmarks and to cut down on computing resources. For all of our experiments, we used a Reduce LR on Plateau LR scheduler with a patience of 10 and a factor of 0.1.

Moreover, we tested training our model with and without augmentation and with several combinations of augmentations. We applied these augmentations either per batch or per single pair. We tested learning rates of 0.01, 0.001, 0.0001, and batch sizes of 32, 64, 128 throughout hyperparameter tuning. Since, the number of experiments grew exponentially with every hyperparameter, we used Weights & Biases (W&B) to keep track of our more than a hundred runs. Latest 48 of these are also recorded in our W&B report [2].

Model	**Augmentation**	**PSNR**
GORA (Baseline)	None	24.4621
GORA	Flip	**25.6912**
GORA	Rotation	24.9827
GORA	Flip and Rotation	25.2412
GORA	BCGHS* Adjustment	24.6819
GORA	All	25.3470

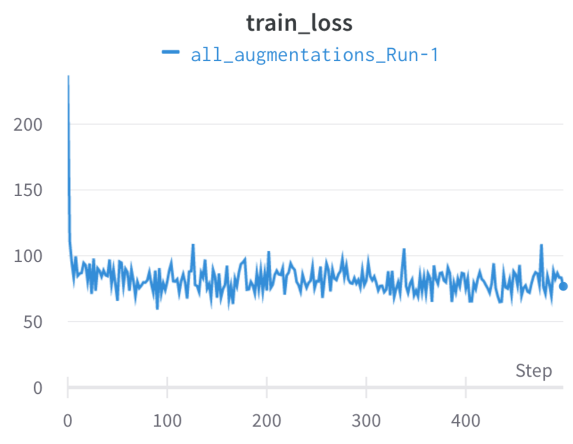
Table 1: PSNR Scores of GORA with Various Augmentations
*BCGHS: Brightness, Contrast, Gamma, Hue, Saturation

[2]Hyperparameter Search Report

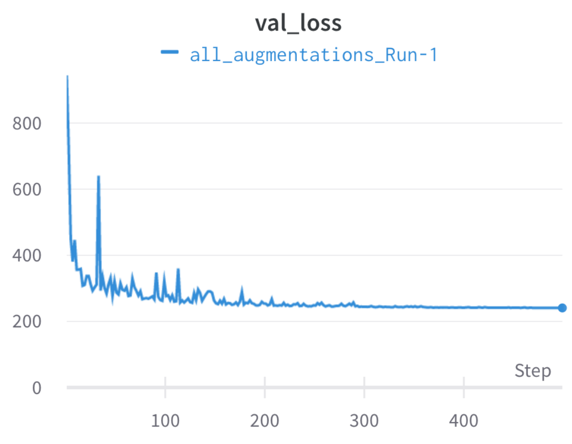
4 Results

As a result of our hyperparameter tuning, we identified the following best parameters: `num_epochs`=250, `batch_size`=32, `learning_rate`=0.0001, `shuffle=True` and `augmentation='all'`. In Figure 2, we present the results for the training and validation loss for this model. In Figure 3, we can see a comparison between the noisy input image, our model's prediction, and the target ground truth which shows that our model has learned to remove the noise.

Throughout our report, we refer to our best model as the one that does not achieve the highest PSNR score but rather the one that generalizes the most. Therefore, the more generalizable model that is trained with all the aforementioned augmentations in subsection 2.2 is reported as our *best* model which achieves an outstanding PSNR score of 25.3470 dB.

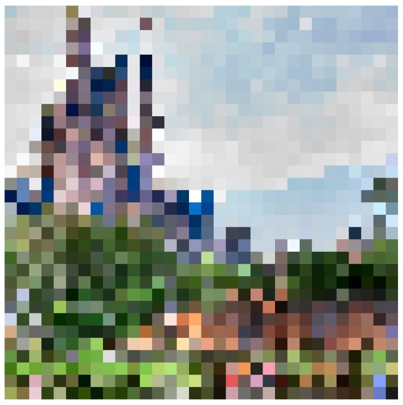


(a) Train Loss

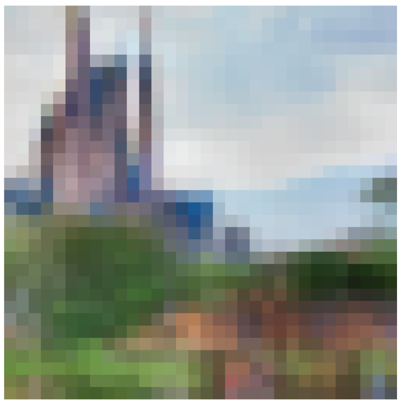


(b) Validation Loss

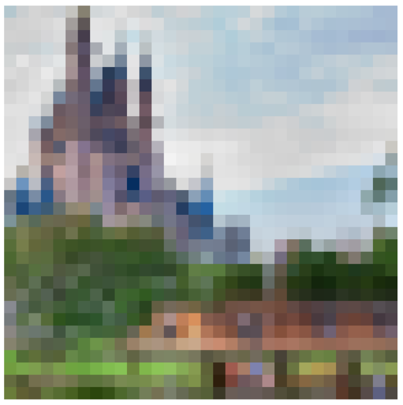
Figure 2: Train and Validation Loss Curves of the Best Model



(a) Noisy Input Image



(b) Predicted Image



(c) Ground Truth

Figure 3: Example of a Noisy Input, Model's Prediction and the Ground Truth

5 Conclusion

We have shown empirically that our model GORA that comprises of convolutional and transposed convolutional layers, can effectively learn to recover signals in noisy data without using any clean examples. For future work, the network can be improved with a thorough architectural study and should be benchmarked on more difficult datasets.

References

[BIGT20] Bhakti Baheti, Shubham Innani, Suhas Gajre, and Sanjay Talbar. Eff-unet: A novel architecture for semantic segmentation in unstructured environment. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1473–1481, 2020.

[LMH+18] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2Noise: Learning image restoration without clean data. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2965–2974. PMLR, 10–15 Jul 2018.

[PGM+19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[RPB15] O. Ronneberger, P.Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015. (available on arXiv:1505.04597 [cs.CV]).

[ZSTL19] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: Redesigning skip connections to exploit multiscale features in image segmentation, 2019.