Report on Image Denoising Using Autoencoders

Title
"De-Noise: Image Restoration using Autoencoders"

Introduction
Artificial Neural Networks (ANNs) are a set of algorithms modeled after the biological neural networks in the human brain. ANNs have been successfully used in various applications, including image processing, speech recognition, and natural language processing. One popular type of ANN is the autoencoder, which is a neural network that learns to reconstruct inputs from a compressed representation, known as a latent space. In this project, we aim to use an autoencoder to denoise images that have been corrupted with Gaussian noise.

Project Goal
The goal of this project is to develop a denoising autoencoder that can remove noise from images. The objective is to improve the visual quality of noisy images and make them clearer and more useful for various applications. In order to achieve this, a neural network architecture known as a denoising autoencoder was implemented. The autoencoder was trained on a dataset of human faces that contained noisy images with Gaussian noise added to them. The neural network was designed to learn the underlying patterns in the data and then use that knowledge to remove the noise from new images. The ultimate goal was to develop a model that could effectively denoise any given image and produce high-quality results. The success of this project would demonstrate the usefulness of denoising autoencoders

Project Description
In this project, we have built and trained a denoising autoencoder neural network from scratch. The data used in this project is the human face dataset from Kaggle, which includes 10,000 samples of both genders and

people of different ages. We experimented with different types of noise, including Poisson noise, dropout noise, salt and pepper noise, and Gaussian noise. The noise used in this project is Gaussian noise. Then we started experimenting with different loss functions like MSE Loss, L1 Loss etc in order to get better results. We then just chose MSE loss because it measures the difference between the input and the reconstructed output, encouraging the model to generate outputs that are close to the original inputs, which helps to reduce the noise present in the input data.

Preprocessing

Before training the denoising autoencoder, we preprocessed the images by normalizing them to a range of 0 to 1. Then we created the dataset and dataloaders using PyTorch modules.

Model Architecture

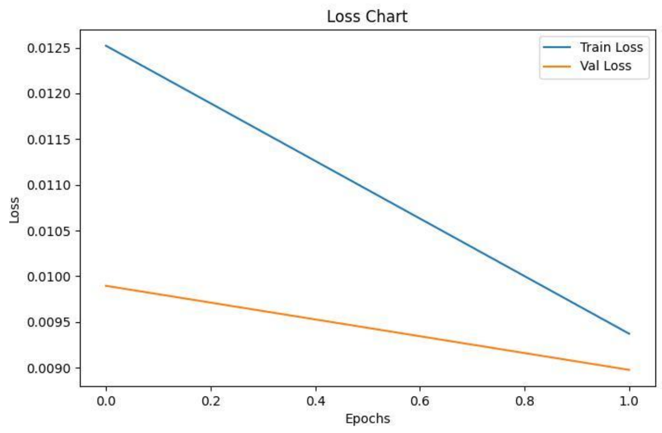
For the encoder of the autoencoder, we used Convolution 2d layers with a stride of 1 and a kernel size of 5. Similarly, for the decoder part of the autoencoder, we used ConvTranspose 2d layers with a stride of 1 and a kernel size of 5. The activation function used in these 2d layers is ReLU, and batch normalization has also been used. We did not use max pooling because we wanted to keep spatial information when upsampling the image. The activation layer at the end of the decoder is sigmoid. The loss function used is MSELoss.

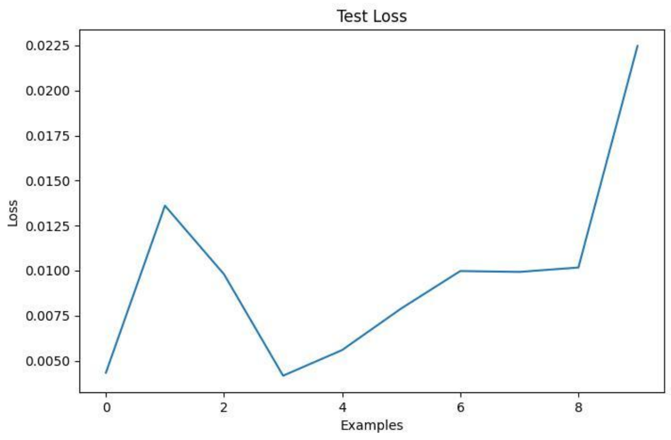
Training and Evaluation

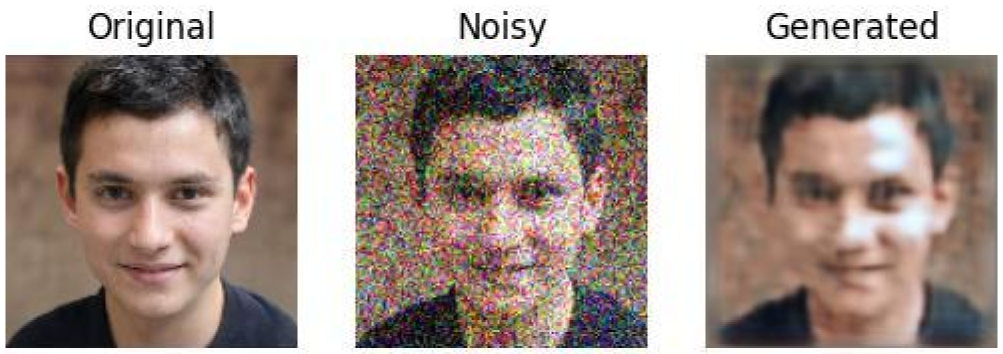
After preprocessing the images and designing the autoencoder architecture, we trained the model using the dataset and data loaders. We validated the model on a validation set and then tested it on a separate test set. The loss during training is in the range of 0.004 to 0.012 for all the train, validation, and test sets. No overfitting is occurring, and the results are good.

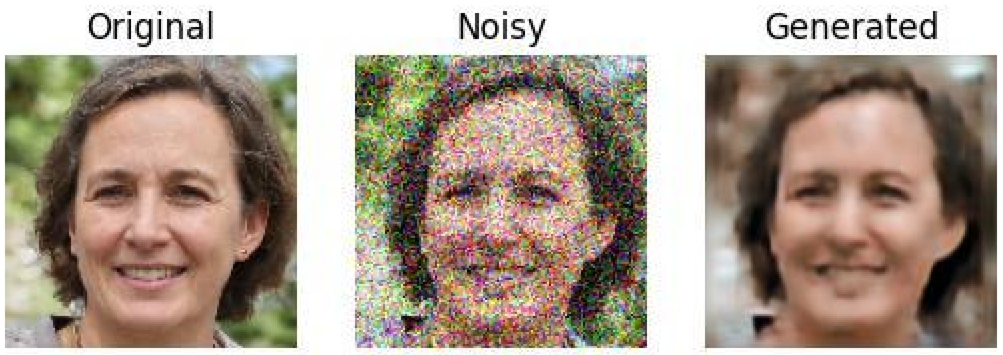
Results

The autoencoder was validated after every epoch and also tested at the very end of the training session. Following were the losses.









Conclusion

In this project, we successfully denoised images using a denoising autoencoder. We experimented with different types of noise and found that Gaussian noise is suitable for this task. Our denoising autoencoder architecture consists of Convolution 2d layers for the encoder and ConvTranspose 2d layers for the decoder. The activation function used is ReLU, and batch normalization has been used. We used PyTorch to preprocess the images and train the denoising autoencoder. The denoising autoencoder was able to remove the noise from the images and produce a clean image. The results were evaluated using the mean squared error loss function and showed good performance.

Future Work

In future work, we could experiment with different types of noise and compare the denoising performance of the autoencoder. We could also explore different architectures for the autoencoder, such as using residual connections, different types of activation functions, or using a U-Net architecture. Additionally, we could train the denoising autoencoder on larger datasets and evaluate its performance on images with different resolutions and sizes.

Member Contributions / Work Division

All members contributed equally to the project and the writing of this report.

- Saleh Ahmad: Data preprocessing, model design and implementation, experimentation, and writing the report.
- Muhammad Ammar: Experimentation, result analysis
- Muhammad Usman: Experimentation, result analysis