# CSCI 3485 Lab 6: Image Denoising

Rafael Almeida, Ryan Novitski

November 15, 2024

## 1 Introduction

Image denoising is a fundamental task in computer vision, aimed at bringing low-resolution images that have been corrupted by noise to their original, high-quality state. This technique has many real-world applications, including improving the quality of medical images, enhancing low-light photography, and removing any compression found in digital artifacts. Denoising is additionally a crucial process for many other fields in computer vision such as classification or object detection, where increasing the resolution of an image can significantly benefit their overall accuracy.

The progression of denoising technology was greatly influenced by the use of Convolutional Neural Networks(CNNs), allowing the model to extract features directly from localized data. This lead to the creation of the Autoencoder model, which builds upon the idea of CNNs. Autoencoders are specifically trained to compress input images to a lower resolution maintaining only the important features with an encoder, then to reconstruct the original image at its original resolution as closely as possible with a decoder.

Although the Autoencoder method garnered great successes in comparison to previous methods, the U-Net provided a few modifications to the model's structure that increased its accuracy. By using skip connections that directly link encoder and decoder layers, the U-Net is able to retain spatial context to a much greater degree of accuracy that the Autoencoder was incapable of.

## 2 Methodology

### 2.1 Data

Both the Autoencoder and U-Net models were trained on the MNIST dataset consisting of handwritten digits, with a training set of 60,000 images and a testing set of 10,000 images.

### 2.2 Noise Addition

We introduced noise to the input images using the formula:

$$Y = (1 - \mu) \cdot I + \mu \cdot N \tag{1}$$

Where:

- $I$ is the original image, normalized to have pixel values between 0 and 1.
- $N$ is the noise matrix, generated with random values from a Gaussian distribution.
- $\mu \in [0, 1]$ is the noise factor that controls the level of noise.
- $Y$ is the resulting noisy image.

The noise factor $\mu$ was varied to simulate different levels of noise:

- Mild noise: $\mu = 0.25$
- Medium noise: $\mu = 0.6$
- Severe noise: $\mu = 0.9$

## 2.3 Models and Implementations

We implemented two models (an Autoencoder and a U-Net) to perform the task of image denoising. Each of these models were trained using the same adjustable parameters(learning rate, optimizer, dataset, and epochs).

### 2.3.1 Autoencoder

**Encoder** The encoder takes a 28x28 input image and flattens it to a vector of size 784, followed by three fully connected linear layers. The first linear layer reduces the dimensionality to 112, followed by a ReLU activation function. The second layer reduces the dimensionality to 56, again followed by a ReLU activation function. Finally, the third layer compresses the data to a latent representation of 28 dimensions. This architecture allows the Autoencoder to extract the most critical features from the image while discarding irrelevant information.

**Decoder** The decoder takes the latent representation from the encoder and reconstructs the original input. It consists of three fully connected layers in reverse order; the first layer expands the dimensionality to 56, followed by a ReLU activation. The second layer increases the size to 112, again with ReLU activation, and the third layer reconstructs the output back to a vector of size 784. Finally, the output is reshaped to 28x28 and passed through a Sigmoid function to normalize pixel values between 0 and 1.

**Model Summary** The Autoencoder model consists of a total of 95,844 trainable parameters, allowing it to effectively learn a compressed representation of the data. The full model summary is shown in Table 1.

| Layer (type) | Output Shape | Param # |
|---|---|---|
| Flatten | [-1, 784] | 0 |
| Linear | [-1, 112] | 87,920 |
| ReLU | [-1, 112] | 0 |
| Linear | [-1, 56] | 6,328 |
| ReLU | [-1, 56] | 0 |
| Linear | [-1, 28] | 1,596 |
| ReLU | [-1, 28] | 0 |
| Linear | [-1, 56] | 1,624 |
| ReLU | [-1, 56] | 0 |
| Linear | [-1, 112] | 6,384 |
| ReLU | [-1, 112] | 0 |
| Linear | [-1, 784] | 88,592 |
| Sigmoid | [-1, 784] | 0 |

Table 1: Autoencoder Model Summary

### 2.3.2 U-Net

**Encoder** The encoder portion of the U-Net consists of convolutional layers with increasing filter sizes to progressively capture complex features of the input image. Each convolutional layer is followed by a Batch Normalization layer and a ReLU activation function to stabilize learning. Additionally, Max Pooling layers are used to reduce the spatial dimensions, allowing for extraction of high-level features.

The encoder begins with an initial convolution that increases the number of channels from 1 to 4, followed by another convolution to refine these features. This process continues with subsequent convolutional blocks where the channels are doubled until it reaches 32, with each step followed by downsampling using Max Pooling.

**Decoder** The decoder is the counterpart to the encoder, consisting of a series of transposed convolutional layers (de-convolutions) used to upsample the image. Each upsampling step is followed by convolutional layers to refine the upsampled features. The skip connections from the encoder are concatenated with the corresponding decoder layers to help retain spatial information that would otherwise be lost.

The decoder begins by upsampling from 32 channels to 16 using a transposed convolution, followed by convolutions to refine the output. This pattern

| Layer Block | Output Shape | Total Params # |
|---|---|---|
| **Initial Conv Block (x2)**<br>Conv2d → BatchNorm2d → ReLU (Repeated Twice) | [-1, 4, 28, 28] | 196 |
| **Downsampling Block 1**<br>MaxPool2d → Conv2d → BatchNorm2d → ReLU (x2) | [-1, 8, 14, 14] | 912 |
| **Downsampling Block 2**<br>MaxPool2d → Conv2d → BatchNorm2d → ReLU (x2) | [-1, 16, 7, 7] | 3,872 |
| **Downsampling Block 3**<br>MaxPool2d → Conv2d → BatchNorm2d → ReLU (x2) | [-1, 32, 3, 3] | 14,016 |
| **Upsampling Block 1**<br>ConvTranspose2d → Conv2d → BatchNorm2d → ReLU (x2) | [-1, 16, 7, 7] | 11,096 |
| **Upsampling Block 2**<br>ConvTranspose2d → Conv2d → BatchNorm2d → ReLU (x2) | [-1, 8, 14, 14] | 2,800 |
| **Upsampling Block 3**<br>ConvTranspose2d → Conv2d → BatchNorm2d → ReLU (x2) | [-1, 4, 28, 28] | 748 |
| **Final Output Layer**<br>Conv2d → Sigmoid | [-1, 1, 28, 28] | 37 |

Table 2: Condensed U-Net Model Summary

continues until the number of channels is reduced to the original single channel with a final convolutional layer, and a Sigmoid activation is applied to produce the output between 0 and 1.

**Model Summary** The U-Net model used in this study consists of 30,677 trainable parameters, a more lightweight variant adapted to handle the smaller input size of MNIST images. The complete model summary is presented in Table 2.

**Implementation** Both models were implemented in PyTorch and trained on an NVIDIA A100 GPU. Each model had its own advantages and limitations, which will be further discussed in the following sections.

### 2.4 Experiment Setup

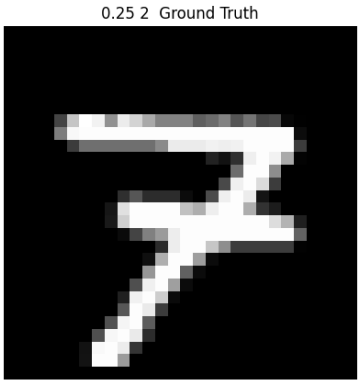
- **Epochs**: Both models were trained for 20 epochs each training session.
- **Optimizer**: Both models were trained using the Adam optimizer, which was chosen for its adaptive learning rate capabilities. The initial learning rate was set to 1e-3.
- **Learning Rate Scheduler**: A step-wise learning rate scheduler was used to reduce the learning rate by half every 10 epochs, allowing for a more stable convergence as training progressed.
- **Batch Size**: A batch size of 64 was used for all experiments, striking a balance between computational efficiency and effective gradient estimation.
- **Loss Function**: The Mean Squared Error (MSE) loss was used as the objective function for the denoising task. MSE is well-suited for pixel-wise comparison of the noisy image to its ground truth.

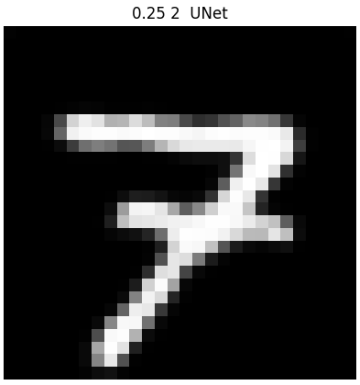
## 3 Results

### 3.1 Qualitative Analysis

#### 3.1.1 Noise level: 0.25

These are the ground truth images as well as the results for the Autoencoder and U-Net for noise levels of 0.25.

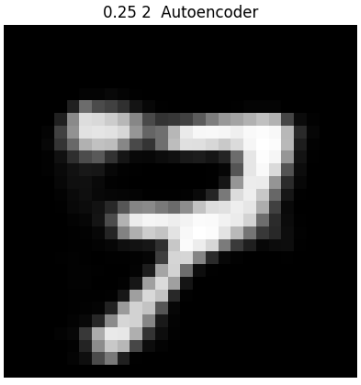


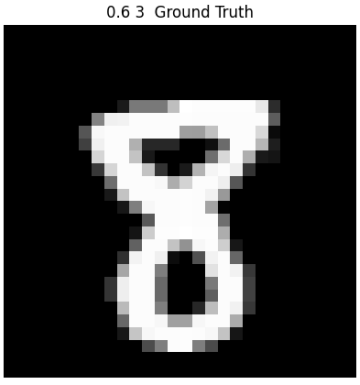



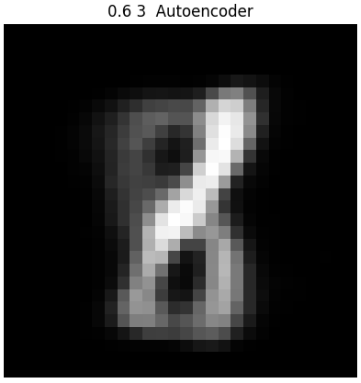

**0.25 Results** Looking at the images qualitatively, it is already becoming clear the better performance in the U-Net over the Autoencoder, even with a noise level as low as 0.25. The image produced by the U-Net is nearly identical to the ground truth image. While the Autoencoder does a good job producing a similar image to the ground truth, it is definitely of lower quality and accuracy than the U-Net's.

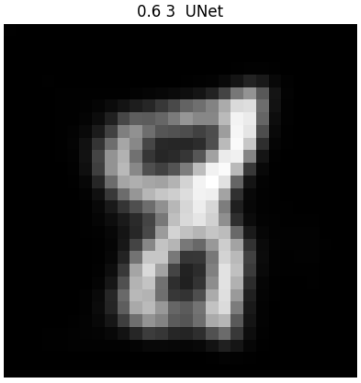
### 3.1.2 Noise level: 0.6

These are the ground truth images as well as the results for the Autoencoder and U-Net for noise levels of 0.6.



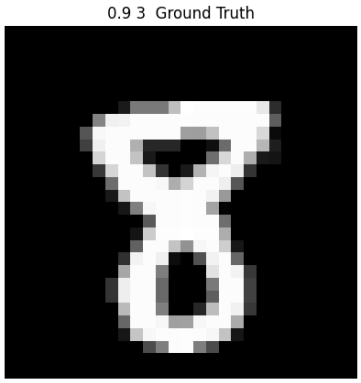


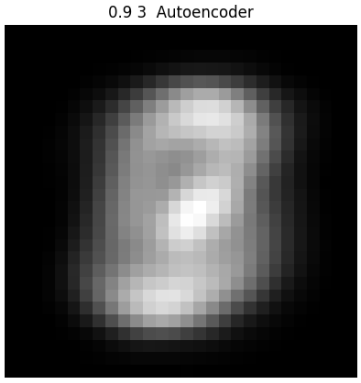



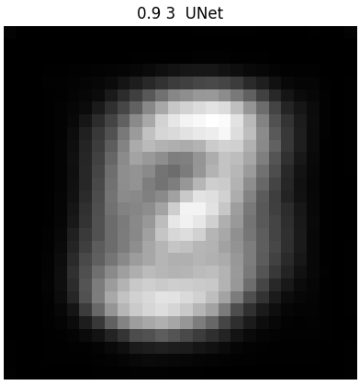

**0.6 Results** With these results, the U-net's superiority in retaining spacial context becomes clearer. It is easy to see that the U-net output is significantly closer to the ground truth, but you can also begin to see some of the issues arising with the Autoencoder. Although the Autoencoder catches the left-to-right diagonal of the '8' pretty accurately, the rest of the number seems to be duller and more like a typewritten '8' instead of the handwritten nuances found in the ground truth. Additionally, although the U-Net's reconstruction was a little thinner, the overall shape and style of the handwritten 8 was retained in its entirety.

#### 3.1.3 Noise level: 0.9

These are the ground truth images as well as the results for the Autoencoder and U-Net for noise levels of 0.9.






**0.9 Results** With this level of noise, although the U-Net is slightly clearer, both models produced a very blurry image that strayed relatively far from the ground truth image. It seemed they both began constructing the proper details extracted from the ground truth, but did not have enough time to properly denoise the image to a successful level.

### 3.2 Quantitative Analysis

To quantitatively evaluate the performance of both the Autoencoder and U-Net models on the denoising task, we used the Mean Squared Error (MSE) as our primary metric. MSE was computed for each of the three noise levels, providing a numerical value indicating how close the denoised image was to the original ground truth. A lower MSE indicates better performance, as it implies a smaller difference between the denoised output and the ground truth image.

Table 3 shows the results for both models at each noise level.

| Noise Level ($\mu$) | Model | MSE |
|---|---|---|
| 0.25 | Autoencoder | 0.0165 |
| | U-Net | 0.0047 |
| 0.6 | Autoencoder | 0.0433 |
| | U-Net | 0.0370 |
| 0.9 | Autoencoder | 0.0672 |
| | U-Net | 0.0664 |

Table 3: Mean Squared Error (MSE) for Autoencoder and U-Net across different noise levels.

## 4 Discussion

Based off of these results, it is clear that the U-Net consistently outperformed the Autoencoder in terms of MSE at lower and medium noise levels ($\mu = 0.25$ and $\mu = 0.6$). For the mild noise level ($\mu = 0.25$), the U-Net had a significantly lower MSE of 0.0047 compared to the Autoencoder's 0.0165, indicating that the U-Net retained more spatial details and better approximated the ground truth.

At the medium noise level ($\mu = 0.6$), the U-Net also showed better performance with an MSE of 0.0370 compared to 0.0433 for the Autoencoder. This suggests that U-Net's skip connections allowed it to recover finer details even when more noise was introduced.

However, at the highest noise level ($\mu = 0.9$), the gap between the two models narrowed, with both models producing similar MSE values—0.0664 for the U-Net and 0.0672 for the Autoencoder. This indicates that both models struggled to effectively denoise the images under severe noise conditions, with their denoised outputs being more similar to each other.

The trends observed suggest that while the U-Net performs consistently better across various noise levels, its advantage decreases as the noise level increases. This may be due to the challenge of distinguishing signal from noise at high levels, where the original image details become heavily obscured. The Autoencoder, lacking the skip connections present in the U-Net, faced more difficulty in preserving spatial coherence, which is especially evident in the high noise conditions.

Overall, the quantitative results align well with the qualitative analysis, demonstrating the superiority U-Net has in retaining finer spatial information and performing better under noise conditions. A potential reason for the higher MSE values could simply be lack of training and depth. For this experiment, we trained each model with a batch size of 64 and 20 epochs. Along with adding additional upsampling and downsampling layers, an increase in batch size or training epochs can absolutely increase the accuracy with higher noise levels. If this were the case, based on these results, the difference in performance between the Autoencoder and U-Net would be much more severe.

# 5 Code

GitHub Repository