Neural Networks Homework 4 Report

Dataset and Image Preprocessing

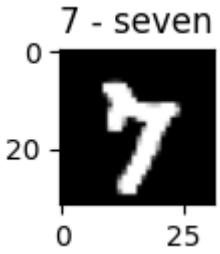
We handled the loading and verification of datasets. The datasets mentioned are MNIST and CIFAR-10, which are standard benchmarks in the field of machine learning and computer vision.

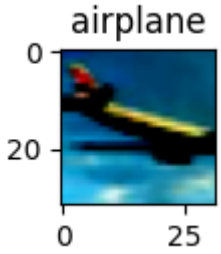
The shapes suggest that the images are resized to a common format of `[128, 3, 32, 32]`, where `128` is the batch size, `3` represents the RGB color channels, and `32x32` is the image resolution. We have a function 'SampleAndMeanDataset' which takes two dataset objects as input and has a unique implementation of an item getter which returns a sample from the first dataset, another from the second, and their calculated mean in each channel, stacked together to form an RGB image that is the mean of those two.
Since those two datasets are MNIST and CIFAR10, we have to also resize the MNIST images to match CIFAR-10 dimensions (32x32). Then, we can calculate the mean and return the two images, their labels and the mean image altogether.
Also, we have four different datasetsL MNIST(train and test), CIFAR10(train and test). From which we built our test and train (mean) datasets.
We chose 128 as the batch size and built dataloaders for both testing and training.

Visualization

The notebook includes code for visualizing images from the datasets. Visualizing the images helps in ensuring that the data preprocessing steps are correctly implemented.

A sample was taken from the (train) dataset and shown as below:

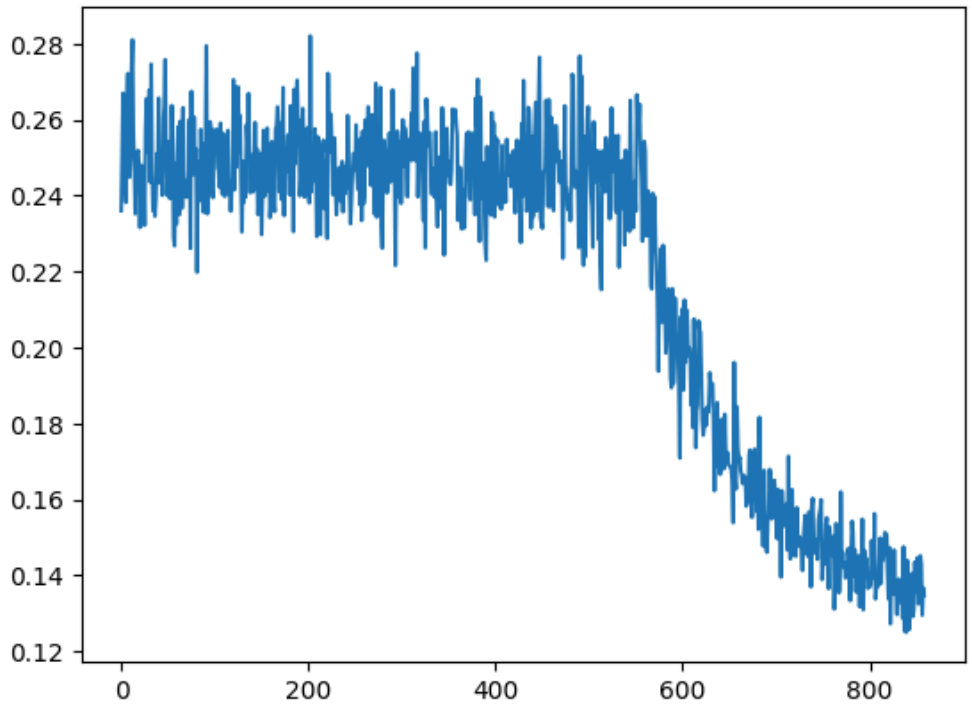




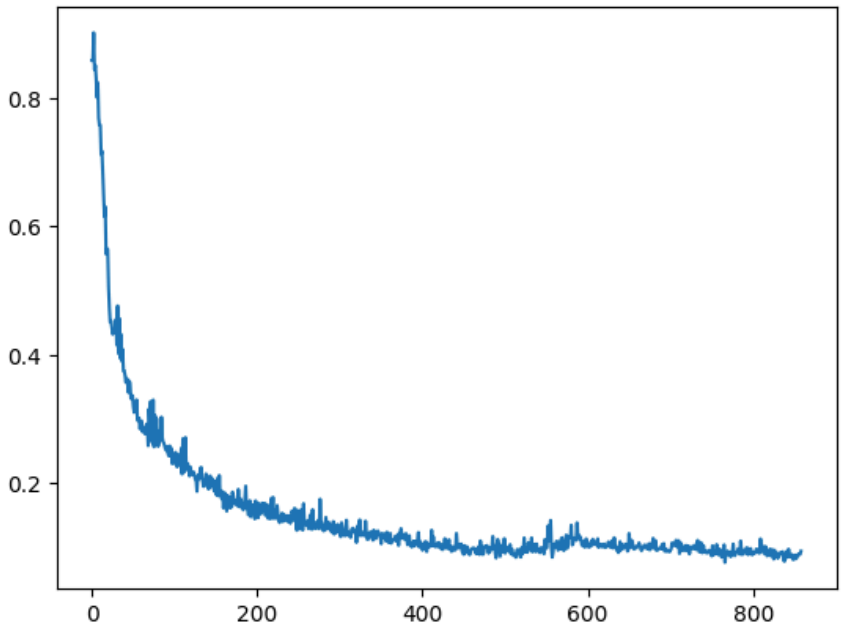
Model

Our model is an autoencoder. It is fine-tuned on the pretrained resnet18 network as the encoder and has two decoder paths for output. One for reconstructing the MNIST image and another for the CIFAR10 one.
The resnet18's job is to quickly find an excellent representation of the input in its latent space and feed it to the decoders.
The decoders have the same structure. Both have 4 layers, use ReLU. The decoder consists of multiple nn.ConvTranspose2d layers, which perform transposed convolution, also known as deconvolution, on the input data. The decoder starts with a transposed convolutional layer that takes in input data with 2 channels and outputs 64 channels using a kernel size of 3. Each transposed convolutional layer is followed by an activation function, nn.ReLU(). As we progress through the layers, the number of output channels increases, and the kernel size adapts to capture more complex patterns. The stride argument controls the stride of the convolution, and padding ensures that the output has the desired spatial dimensions. Finally, the model ends with a transposed convolutional layer that outputs 3 channels, corresponding to the three color channels in an RGB image. The size of the output is (128, 3, 32,32) which means it will reconstruct RGB images for each of the samples in the input batch.

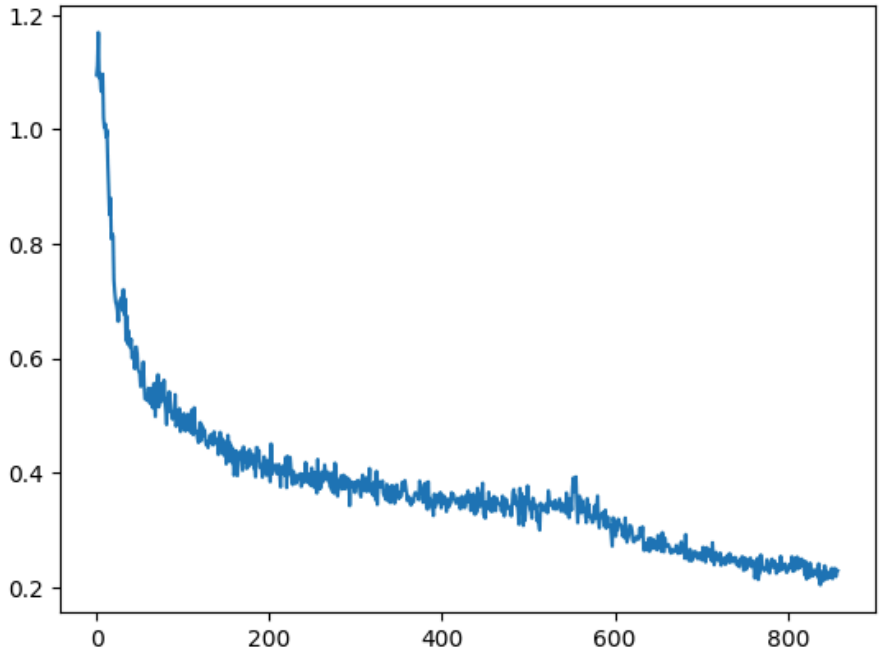
We used MSE to assess the difference of the reconstructed images with the original images. Then, each of the two decoders give their own loss and the sum of those is backproped into the network.
We used ADAM with a learning rate of 10^(-3). We also added some momentum with beta parameters. The betas parameter in the Adam optimizer refers to the exponential decay rates for the first and second-order momentum estimates. Higher values for beta1 and beta2 give more weight to past gradients, while lower values focus more on recent gradients.
We then train the model for only one epoch
Starting from 1.094, the total loss decreases to 0.228 after seeing the whole train data once.
This figure shows the loss for CIFAR10 decoder:



Loss for MNIST decoder:



Total loss over around 400 batches:



Evaluation

We then evaluated the model on the test data.

The notebook imports two key metrics for assessing image quality.

- PSNR: This metric quantifies the reconstruction quality of images. It is commonly used in image compression and denoising tasks.
- Structural Similarity Index (SSIM): This metric evaluates the perceptual similarity between two images. It is particularly useful in tasks where maintaining structural information is crucial, such as image generation and enhancement.
- SSIM considers changes in structural information, making it more aligned with human visual perception compared to PSNR.

Both metrics are essential for evaluating the performance of neural networks in image-related tasks. PSNR provides a mathematical measure of quality, while SSIM aligns more closely with human visual perception.

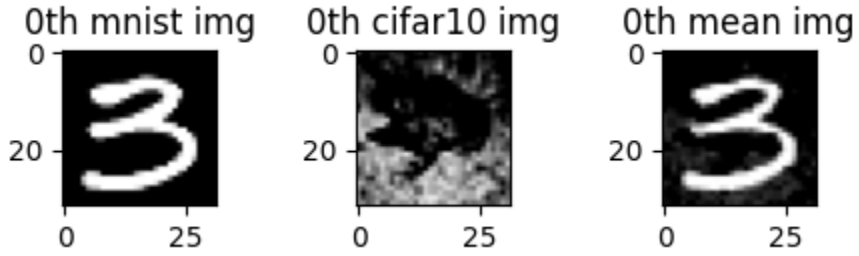
Both were measured for 128 random samples and their mean was reported at the end:
Mean PSNR for cifar10: 62.79011552553733
Mean SSIM for cifar10: 0.9962222993689486
Mean PSNR for mnist: 65.22051003145367
Mean SSIM for mnist: 0.99826334636813

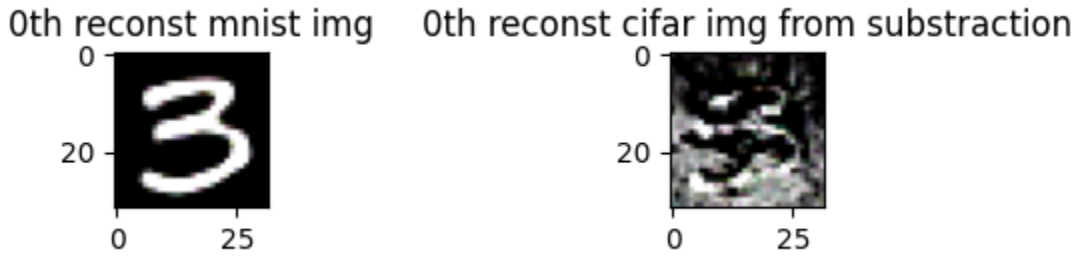
- Mean PSNR: Higher PSNR values indicate better image quality, as they signify a higher ratio of the peak signal (image) to the noise. In our case, the cifar10 set of images has a lower PSNR (62.79) compared to the mnist set (65.22). This suggests that the second set of images has better overall quality.

- Mean SSIM: SSIM values range from -1 to 1, with 1 indicating perfect similarity between images. Both sets of images have high SSIM

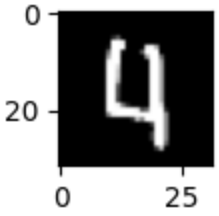
scores (0.9962 and 0.9983), indicating that the images are almost identical. However, the MNIST set has a slightly higher SSIM score, suggesting a better structural similarity with the ground truth images compared to the first set.

In conclusion, the provided scores indicate that the MNIST set of images has slightly better overall quality and structural similarity with the original images compared to the cifar10 set. However, both sets have very high SSIM values, implying that they are almost identical.
However, upon visualizing the reconstructed images, we found out that the MNIST images were very accurate while the reconstructed cifar10 images were not, they were highly blurry and barely recognizable, even tho they had kept the overall color mappings of the original image. That made me think that a better way to get the CIFAR10 image would be to let the network correctly reconstruct the first image, then just use this formula to get the second image:
(2*mean - first_image) = second_image
The visualizations for this step were done on 20 random samples as below:

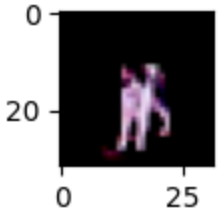




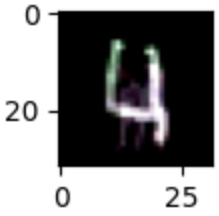
1th mnist img



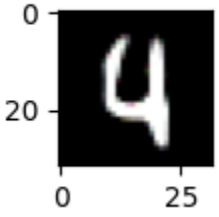
1th cifar10 img



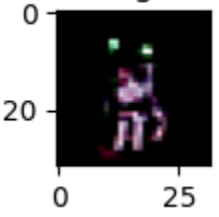
1th mean img



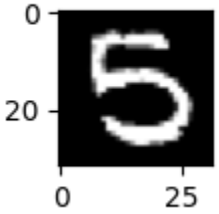
1th reconst mnist img



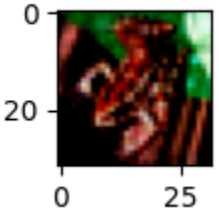
1th reconst cifar img from substraction



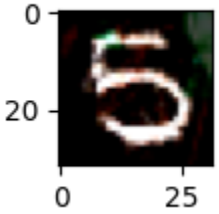
11th mnist img



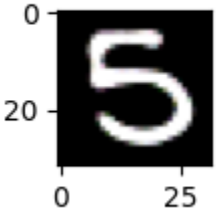
11th cifar10 img



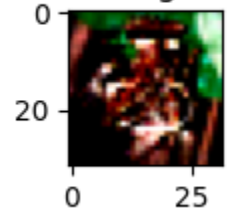
11th mean img



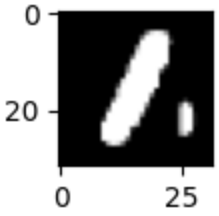
11th reconst mnist img



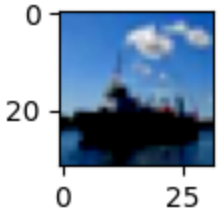
11th reconst cifar img from substraction



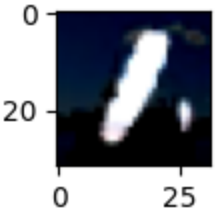
15th mnist img



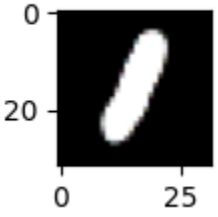
15th cifar10 img



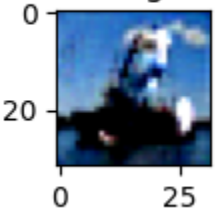
15th mean img



15th reconst mnist img

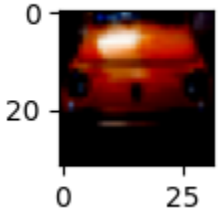


15th reconst cifar img from substraction

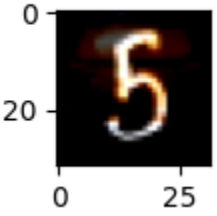


17th mnist img

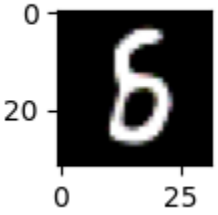
17th cifar10 img



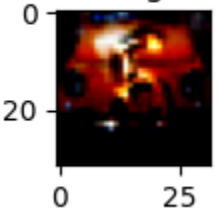
17th mean img

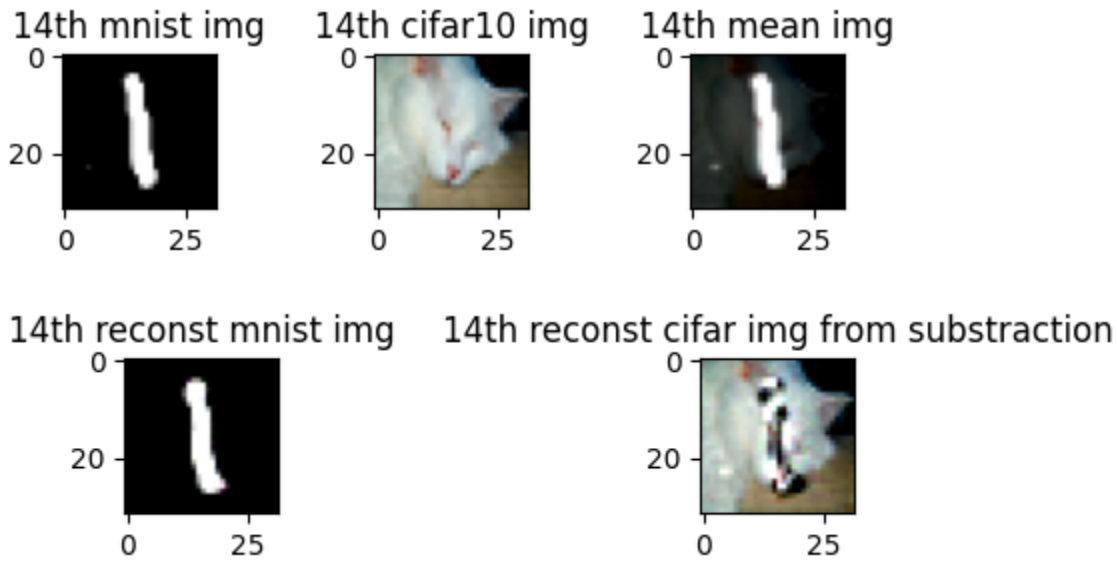


17th reconst mnist img



17th reconst cifar img from substraction





Conclusion

The loss plots were very unstable indicating that whether our samples were too small or the learning rate should have been different. Also, the network learned the handwritten numbers better which can be due to the simpler nature of the data. Also, this data has less details and just the overall structure of the numbers can make us understand the context of the images, compared to the detailed and more complex nature of cifar10 images that necessarily need the details for the human eye to recognize its context. So, i think that is the reason that even though the PSNR and SSIM were good, I still saw the MNIST reconstruction as better but CIFAR10 reconstruction to be bad and decided to use the mentioned formula for getting the CIFAR10 image. Anyhow, the formula is still a valid way to construct the two images because even if we don't have the original data, just by the learned MNIST representation that the AE is able to reconstruct, we can access the second image too.