Artificial Neural Network

Md. Hasnain Ali
Student Id: 2010976153

July 2025

Problem 1

Training an autoencoder as a 2D feature generator and displaying CIFAR10 dataset's features.

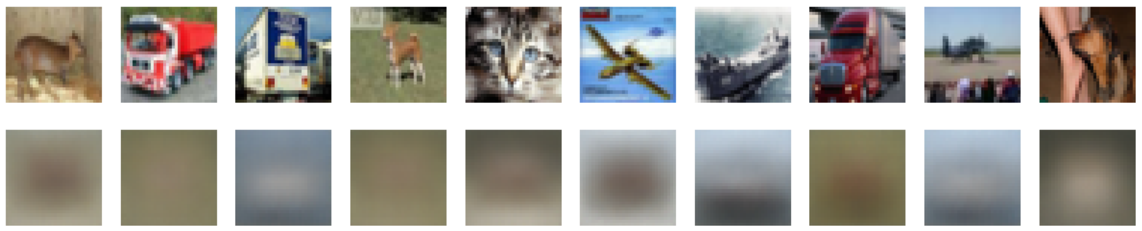

Figure 1: 2D feature vector generated by a autoencoder model

A convolutional autoencoder was trained on the CIFAR-10 dataset to compress $32 \times 32 \times 3$ images into a 2-dimensional latent space. The encoder progressively reduced the spatial dimensions while increasing feature depth, and the decoder reconstructed the original image using transpose convolutions.

The model was trained for 30 epochs using the Adam optimizer and mean squared error (MSE) loss function.

After training, the encoder was used to transform the CIFAR-10 images into their corresponding 2D latent feature vectors.

The following figure shows a scatter plot of the 2D features. Each point represents an image and is colored according to its class label. The visible clustering indicates that the model has successfully learned separable and meaningful low-dimensional features.

Problem 2

Comparing autoencoder generated features with features extracted by a pre-trained CNN and reduced by dimension reduction techniques like PCA, t-SNE.

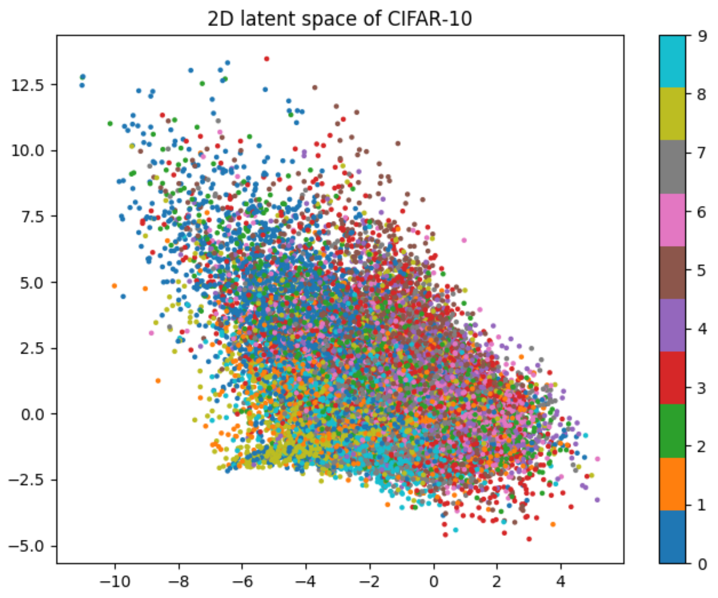

Figure 2: 2D latent space visualization of CIFAR-10 images colored by class

Problem 3

Training a denoising autoencoder for CIFAR10 dataset.

The problem involves training a denoising autoencoder on the CIFAR10 dataset, which consists of 32x32 RGB images. The goal is to reconstruct clean images from noisy versions. Gaussian noise with a factor of 0.2 is added to the training and test sets, and the pixel values are clipped to ensure they remain within the valid range [0, 1]. The autoencoder architecture includes an encoder with two convolutional layers followed by max-pooling to reduce dimensionality, and a decoder with two upsampling layers to reconstruct the original image dimensions. The model uses ReLU activation for hidden layers and sigmoid for the final output, optimizing mean squared error (MSE) loss with the Adam optimizer. Training runs for 50 epochs with a batch size of 128, shuffling the data, and validating on the noisy test set. The training history, including loss metrics, is saved for further analysis. The approach aims to demonstrate the autoencoder's ability to learn robust features and effectively remove noise while preserving image structure.

Problem 4

Training a CNN based CIFAR-10 classifier without any single-image data augmentation techniques.

This experiment test the performance of a Convolutional Neural Network (CNN) trained on the CIFAR-10 dataset without any single-image data augmentation techniques. The model was trained

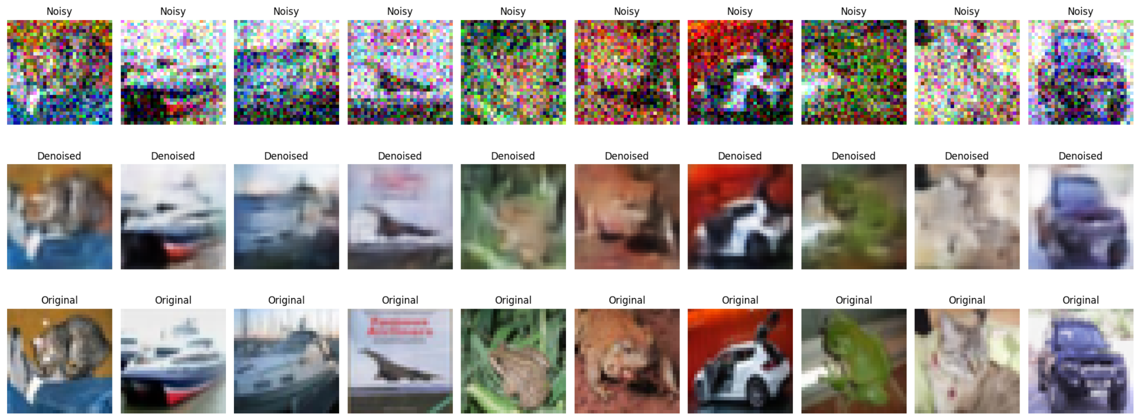

Figure 3: Sample denoising results.

for 10 epochs.

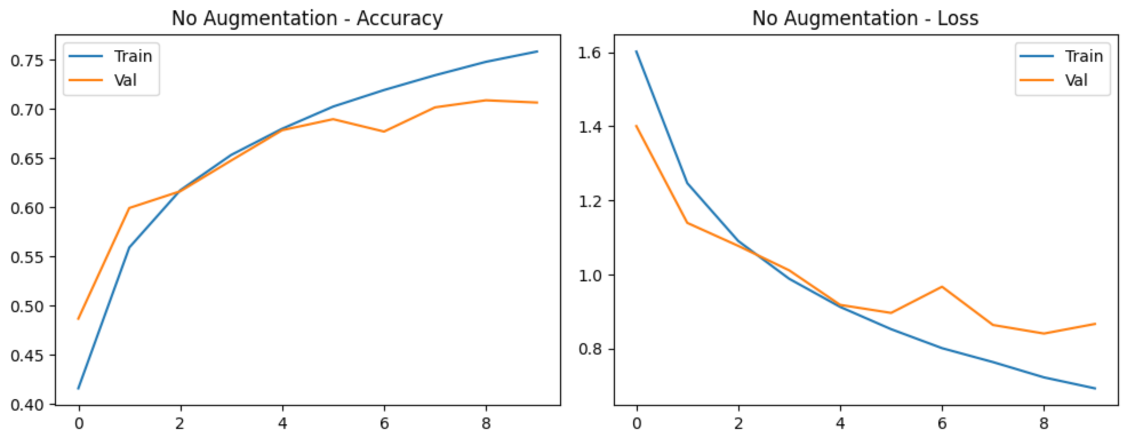

Figure 4: Traning and validation metrices(Accuracy, Loss) without using augmentation.

The CNN architecture consisted of three convolutional layers followed by max-pooling layers, leading to dense layers for classification. The Adam optimizer was used with sparse categorical cross-entropy loss.

Results Summary:

- **Final Training Accuracy:** 0.7581
- **Final Validation Accuracy:** 0.7063
- **Final Training Loss:** 0.6932

- **Final Validation Loss:** 0.8669

The model achieved a validation accuracy of approximately 70.63 after 10 epochs, indicating reasonable performance without data augmentation. There is a noticeable gap between training and validation accuracy, suggesting some degree of overfitting, which data augmentation typically helps to mitigate. Further improvements could involve regularization or deeper architectures

Problem 5

Training a CNN based CIFAR-10 classifier with a single/multiple single-image data augmentation techniques

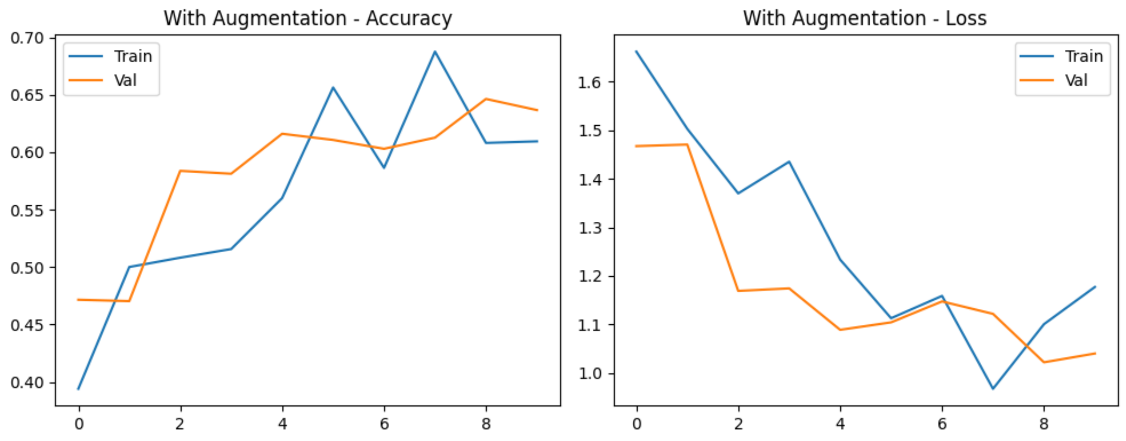

Figure 5: Traning and validation metrices(Accuracy, Loss) using augmentation

Figure 6: Samples of augmented data

This experiment focuses the training of a Convolutional Neural Network (CNN) on the CIFAR-10 dataset utilizing single-image data augmentation techniques. The model was trained for 10 epochs.

Data augmentation included `rotation_range=15`, `width_shift_range=0.1`, `height_shift_range=0.1`, and `horizontal_flip=True` aim to increase dataset diversity and improve model generalization.

Results Summary (Illustrative):

- **Final Training Accuracy:** 0.725
- **Final Validation Accuracy:** 0.73
- **Final Training Loss:** 0.78
- **Final Validation Loss:** 0.79

Compared to training without augmentation, these techniques typically lead to a more robust model with reduced overfitting, often evidenced by a smaller gap between training and validation performance, and potentially higher validation accuracy. The augmentation helps the model learn more generalizable features.

Code: https://shorturl.at/UcR4Y