Image Retrieval Using UNet-based Architecture on CIFAR-10

Abtin Mahyar

1. Overview and Problem Statement

In this project, I created a UNet-based Auto-encoder capable of producing original images by giving their averaged image as its input. the tasks involved are the following:

1. Selecting 1,000 images form original dataset stratified to their label. Make train and test set from these images.
2. Make a proper input-output pipeline in order to train the model.
3. Come-up with an architecture which could extract the most important features form the input images, and decode those features into their original images.
4. Modify the loss function, optimizer and tune the learning rate.
5. Calculate the performance of the final model on train and test set and visualize them.

2. Data Preparation

The process of aggregating and making the data ready for fitting into the models consists of following steps:

1. First of all, the dataset was downloaded from this Github repository which provided CIFAR-10 as images separated in different folders as their labels in jpg format. The data was available in two formats of train and test sets. There were 50,000 records in the training set, and 10,000 records in test set. There were 10 different classes as target labels for the records. The size of each input image is 32 * 32 pixels.
2. No especial transform function was performed on the data, since there was no need to augmentation because of size of the dataset, and the quality of dataset. Only one transformation was used in order to convert raw inputs to Torch tensor for ease of computation in future works.
3. 900 instances of training data were selected for training the model. 100 instances of test data were selected for testing the model.
4. Train, validation and test datasets were passed into their own data loader in order to make the data into batches of size 32. For training phase, I used shuffling for each epoch.
5. For preparing the data for fitting into the model, I divided the dataset into the two groups based on their labels with no overlapping instances. Then, I made a custom dataset which randomly chooses on images in each group and calculate their average image. Finally, it returns the averaged image along with original images.

3. Methodology

First, I made an auto-encoder model with shallow layers, consisting for 3 convolution layers in encoder and decoder. Also, the model has two decoders for making each original image as outputs respectively. But after training the model, results were not clear and they were blurry and similar to each other. Then, I concluded that since the network is shallow, it can not extract the most important features and can not distinguish the original images from each other; therefore, I decided to use UNet architecture as the final model which can definitely extract most useful features from the input.

The final model is the UNet-based model which has two decoders in it in order to produce the original images based on their averaged image. I used Adam as my optimizer with different learning rate. It concluded that 1e-3 is the best value for the learning rate for training with this dataset. I used MSE error as the loss function with average reduction which measures how much pixels in images are far from each other and average the values on total number of pixels in the image. Also, I modify the loss function in a way that, difference of output image of first decoder and first original output which were randomly chosen of first group images will be add up with this difference between output image for second decoder and second original output using MSE loss function and optimizer tries to reduce this value. As a result, first decoder will learn to produce images from first group of data from the representation achieved by the encoder. Same will happen for the second decoder and it will learn to produce the second types of images based on the latent space as its input.

3. Results

Model was trained for 2,000 epochs and achieved 0.0063 MSE on training data which means that each predicted pixel is extremely similar to its corresponding pixel in the original image. Also, model achieved 0.062 MSE on the test data.

Following images are some random samples from training data. Original images are plotted alongside with prediction images. As it can be concluded, model can predict original images very well.