Assignment 1
CSP7030: DLOps
M23CSA014 - Manish V

M23CSA014_DLOps_Assignment1_Q1.ipynb
M23CSA014_DLOps_Assignment1_Q2_a.ipynb
M23CSA014_DLOps_Assignment1_Q2_b.ipynb
M23CSA014_DLOps_Assignment1_Q2_c.ipynb

Q1)

Data Pre-processing:
The code starts by importing the necessary libraries for data preprocessing, model building, evaluation, and visualization. Libraries such as PyTorch, scikit-learn, Seaborn, and Matplotlib are imported.

The Iris dataset is loaded using scikit-learn's load_iris function. The dataset contains features (data) and target labels.

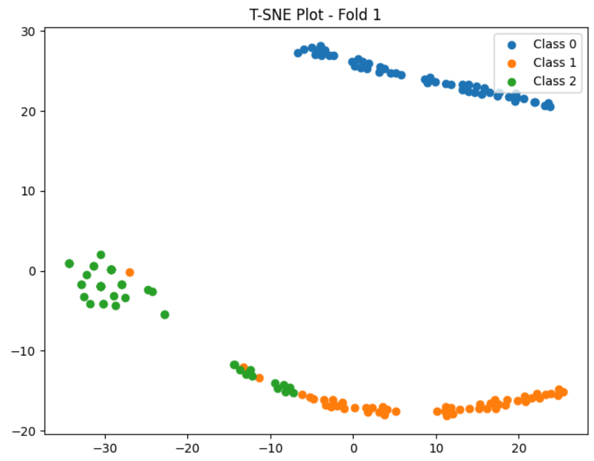
Model Architecture:
The model architecture is defined using a custom PyTorch module named Architecture. It's a simple feedforward neural network with two hidden layers and an output layer.
The input layer has 4 neurons, the first hidden layer has 5 neurons, the second hidden layer has 7 neurons, and the output layer has 3 neurons, corresponding to the three classes in the Iris dataset.
The ReLU activation function is used after each hidden layer, and the softmax activation function is used for the output layer to obtain class probabilities.

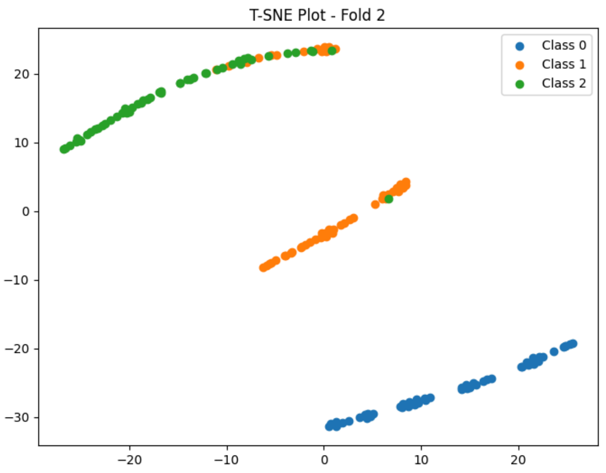
Training and Evaluation:
The code defines functions for training the model (train_model) and generating t-SNE plots (generate_tsne_plot).

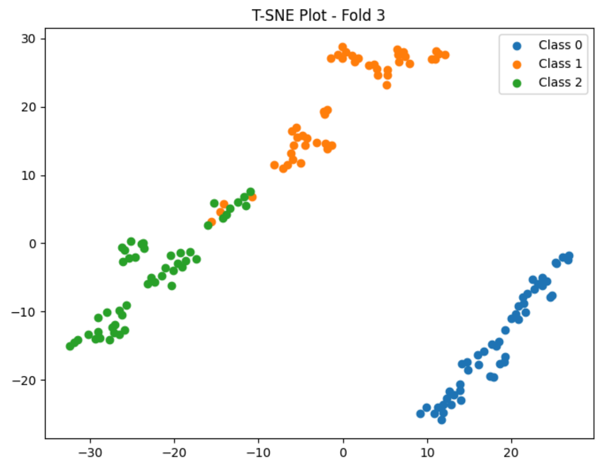
For training, the dataset is split into 10 folds using KFold cross-validation from scikit-learn. Each fold is used iteratively for training and validation. The model is trained for 25 epochs using the Adam optimizer and cross-entropy loss function. Training and validation losses, accuracy, precision, and recall scores are printed and logged using TensorBoard.

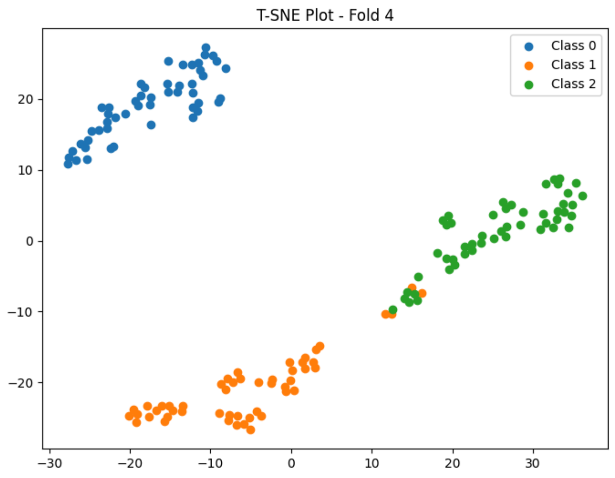
Analysis and Observations:
Observations and analysis can be derived from the printed and logged metrics during training. These include how the training and validation losses change over epochs, how accuracy,

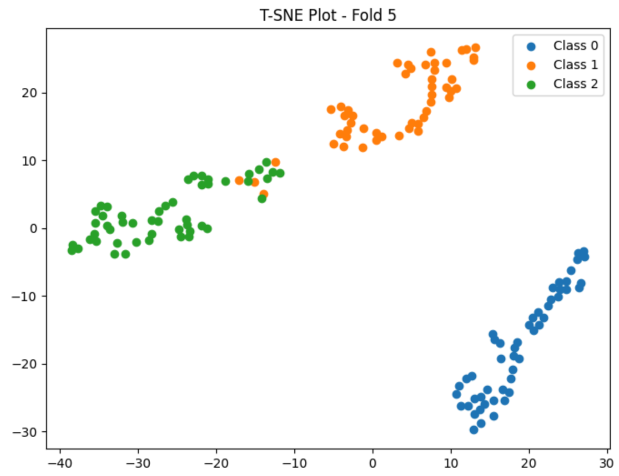
precision, and recall scores vary across folds, as does how well the model separates the classes in the t-SNE plots.

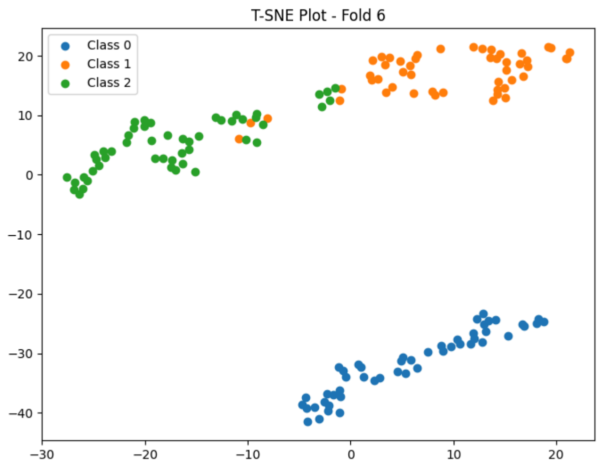


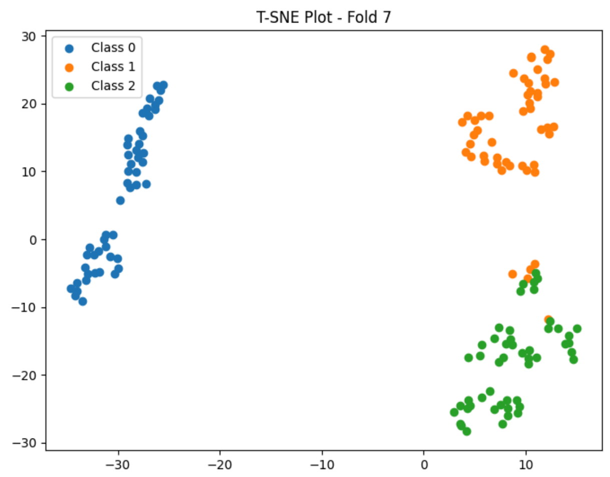


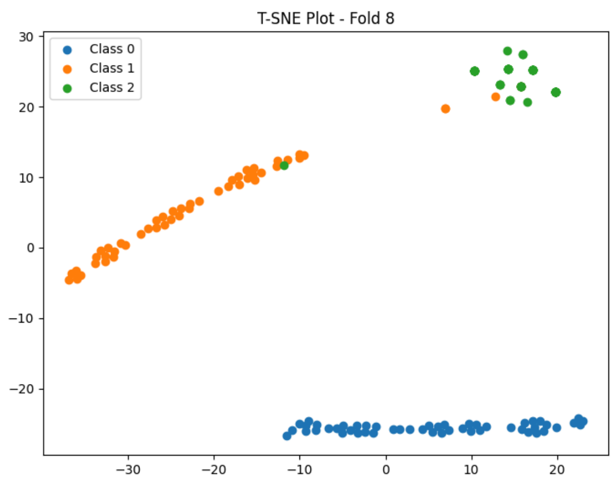


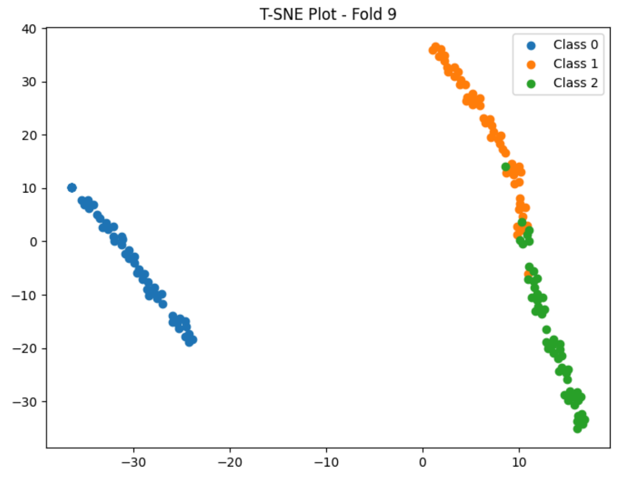


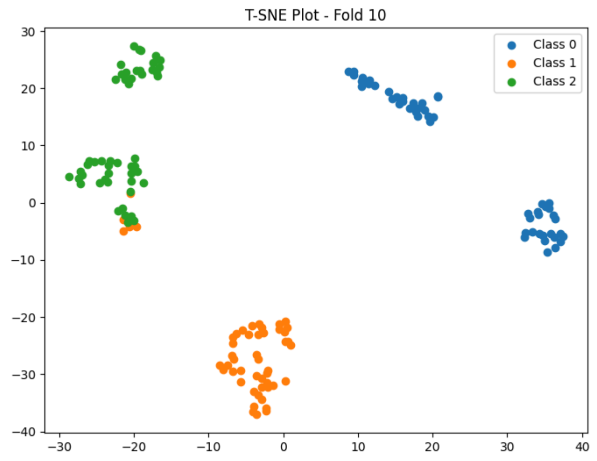












Results:
The results of the training and evaluation are printed during the cross-validation loop. These include training and validation losses, accuracy, precision, and recall scores for each fold and epoch.

Fold: 1, Epoch: 25, Train Loss: 0.8389, Val Loss: 0.8655, Val Accuracy: 60.0000, Val Precision: 0.3929, Val Recall: 0.6667

Fold: 2, Epoch: 25, Train Loss: 0.9293, Val Loss: 0.8846, Val Accuracy: 80.0000, Val Precision: 0.5758, Val Recall: 0.6667

Fold: 3, Epoch: 25, Train Loss: 0.7334, Val Loss: 0.6386, Val Accuracy: 100.0000, Val Precision: 1.0000, Val Recall: 1.0000

Fold: 4, Epoch: 25, Train Loss: 0.6447, Val Loss: 0.6376, Val Accuracy: 93.3333, Val Precision: 0.9524, Val Recall: 0.9333

Fold: 5, Epoch: 25, Train Loss: 0.7587, Val Loss: 0.7273, Val Accuracy: 100.0000, Val Precision: 1.0000, Val Recall: 1.0000

Fold: 6, Epoch: 25, Train Loss: 0.6758, Val Loss: 0.6702, Val Accuracy: 100.0000, Val Precision: 1.0000, Val Recall: 1.0000

Fold: 7, Epoch: 25, Train Loss: 0.6938, Val Loss: 0.7344, Val Accuracy: 93.3333, Val Precision: 0.9524, Val Recall: 0.9167

Fold: 8, Epoch: 25, Train Loss: 0.9261, Val Loss: 0.9895, Val Accuracy: 60.0000, Val Precision: 0.4000, Val Recall: 0.6667

Fold: 9, Epoch: 25, Train Loss: 0.9560, Val Loss: 0.9409, Val Accuracy: 66.6667, Val Precision: 0.5000, Val Recall: 0.6667

Fold: 10, Epoch: 25, Train Loss: 0.7067, Val Loss: 0.7199, Val Accuracy: 100.0000, Val Precision: 1.0000, Val Recall: 1.0000

Q2)

Data Pre-processing:

1. **Loading Data:**

The dataset is loaded using a custom dataset class, CustomDataset.
The training and validation datasets are loaded from the directory /content/drive/MyDrive/hymenoptera_data/train and /content/drive/MyDrive/hymenoptera_data/val, respectively.

2. **Data Transformation:**

Images are resized to (224, 224) using transforms.Resize.
Images are converted to tensors using transforms.ToTensor.

Observations:

1. **Training and Validation Data**:

The training dataset consists of images from the 'train' directory.
The validation dataset consists of images from the 'val' directory.
Both datasets are loaded using the CustomDataset class and transformed using the defined transformations.

Results:

1. **Model Architecture:**

- The model architecture consists of a convolutional autoencoder (CAE).
- The encoder comprises several convolutional layers, followed by batch normalization and leaky ReLU activation functions.
- Following batch normalization and ReLU activation functions, the decoder consists of transposed convolutional layers.

2. **Training Process:**
- The model is trained for 10 epochs using the Mean Squared Error (MSE) loss function and the Adam optimizer with a learning rate of 0.001.
- Training and validation losses are printed for each epoch.

3. **Performance Evaluation:**
- The model's performance is evaluated on the validation dataset.
- Metrics calculated include average MSE, average RMSE, average SSIM (structural similarity), and average PSNR (peak signal-to-noise ratio).
- Visualizations of original and reconstructed images are provided for qualitative assessment.

4. **Feature Representations:**
- Principal Component Analysis (PCA) and t-SNE (t-distributed stochastic neighbor embedding) are used to display the validation dataset's features.

Analysis:

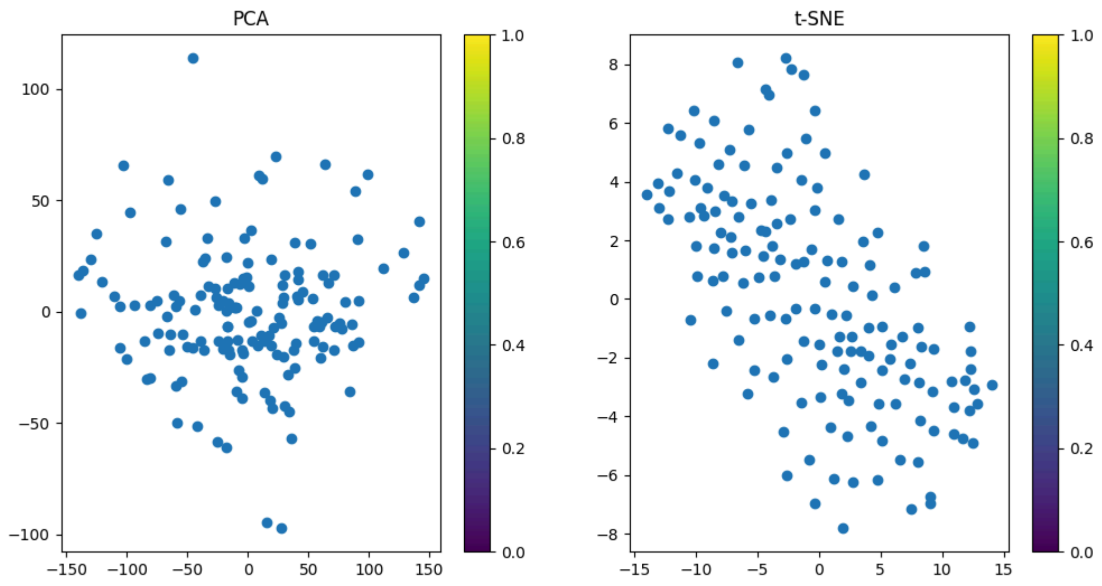
1. **Model Performance:**
- Low MSE and RMSE values indicate that the model produces satisfactory results in terms of reconstruction quality.
- SSIM and PSNR metrics provide insights into the similarity between original and reconstructed images, with higher values indicating better reconstruction quality.

2. **Feature Representations:**
- PCA and t-SNE visualizations demonstrate the distribution of latent features extracted by the CAE model.
- These visualizations offer insights into the separability and clustering of features in the latent space, which can aid in understanding the model's representational learning capabilities.

a) L*a*b* colorspace to the image in RGB Colorspace

Average MSE: 0.0513
Average RMSE: 0.2263

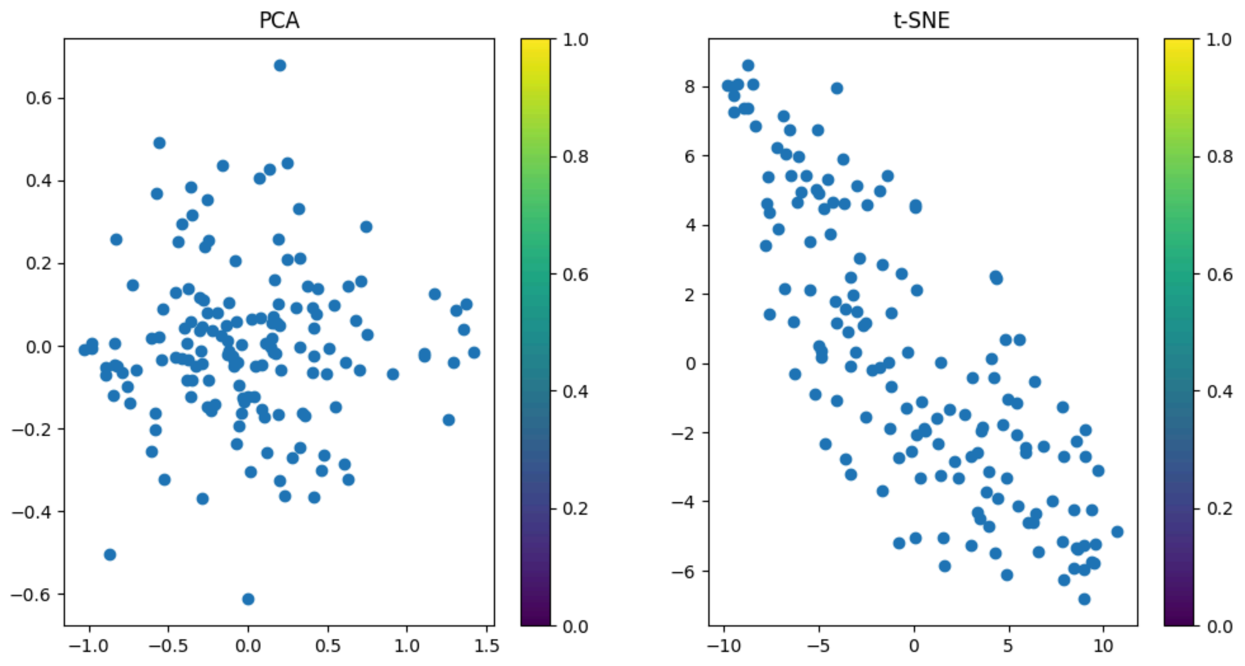
Average SSIM: 0.0013
Average PSNR: -31.33



b) RGB image to negative image

The below graphs are done when only 2 epochs are run in here. (For more epochs results, check the colab file.)

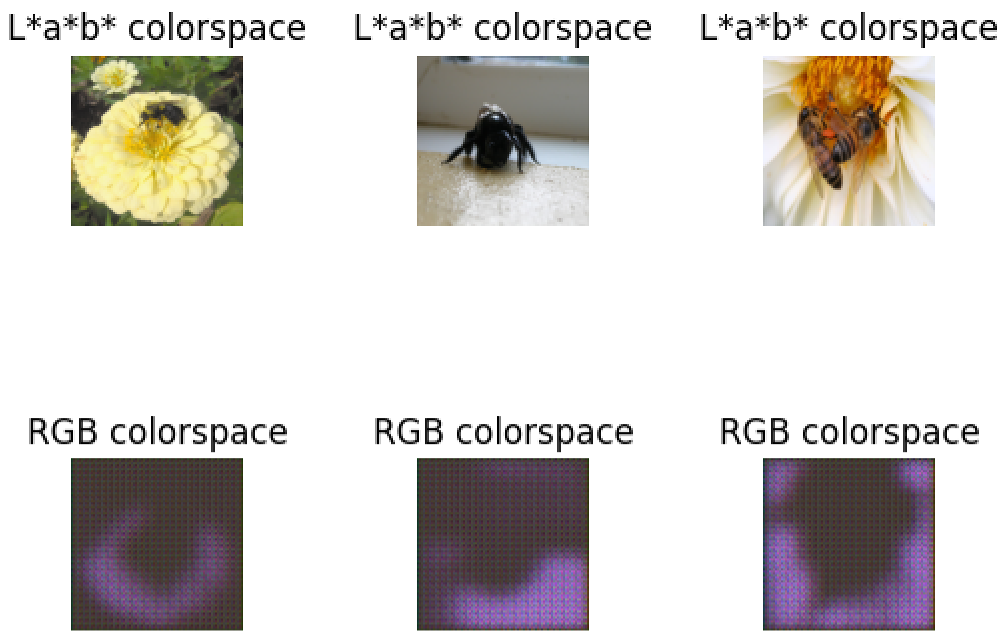
Average MSE: 0.4438
Average RMSE: 0.6662
Average SSIM: 0.0000
Average PSNR: -48.11

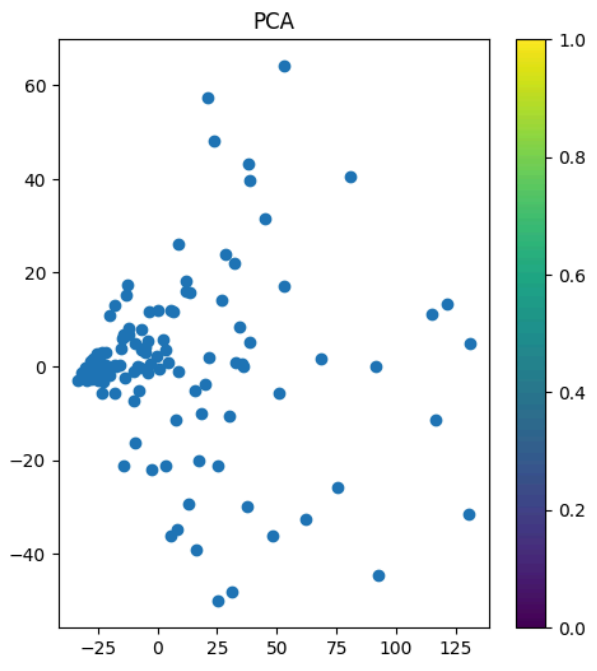


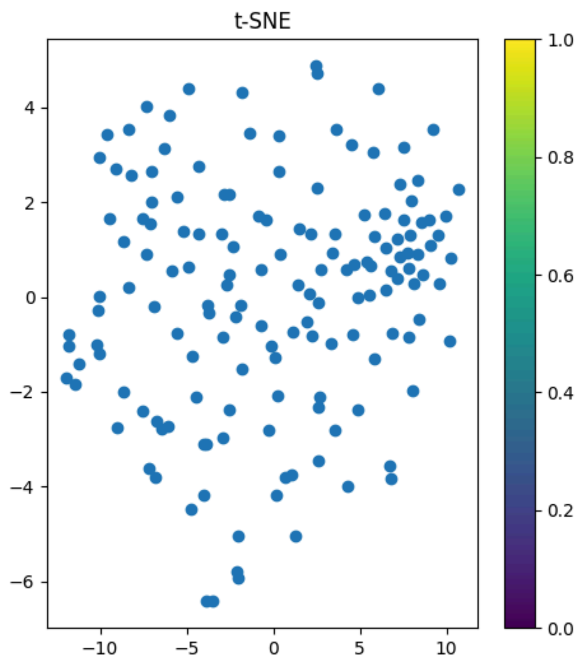
c) RGB image to RGB horizontally flipped image.

The below graphs are done when only 2 epochs are run in here. (For more epochs results, check the colab file.)

Average MSE: 0.1391
Average RMSE: 0.3729
Average SSIM: 0.0757
Average PSNR: 9.24


L*a*b* colorspace
L*a*b* colorspace
L*a*b* colorspace
RGB colorspace
RGB colorspace
RGB colorspace


PCA
60
40
20
0
−20
−40
−25
0
25
50
75
100
125
1.0
0.8
0.6
0.4
0.2
0.0


t-SNE
4
2
0
−2
−4
−6
−10
−5
0
5
10
1.0
0.8
0.6
0.4
0.2
0.0