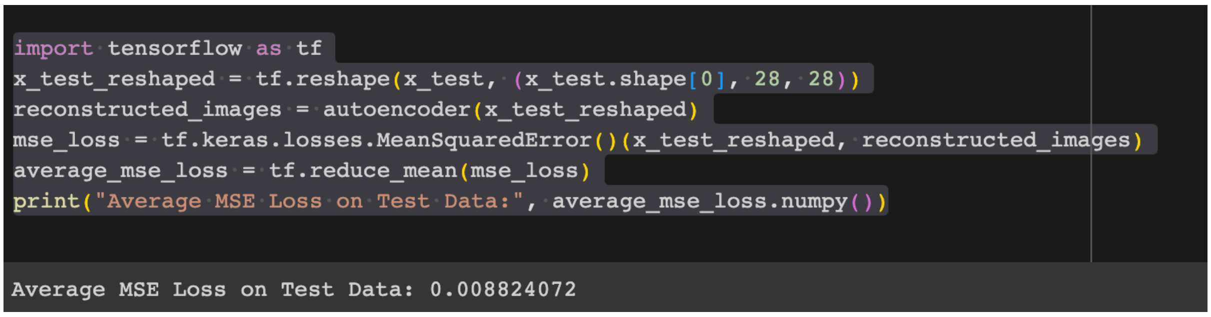
DL lab 7 -Autoencoders

Wanni Arachchige P.N

IT20617028

Answers

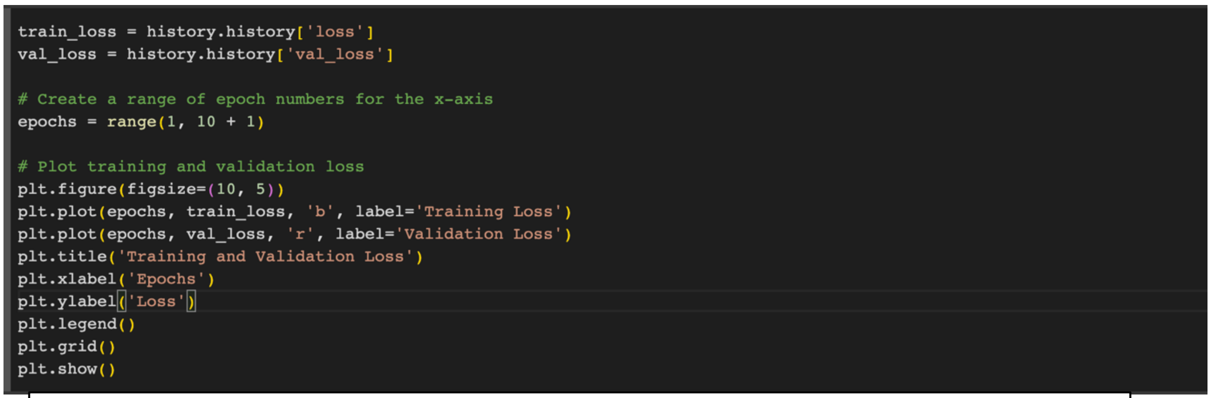
1. Upload the Autoencoder (AE) jupyter notebook file (i.e., lab_7_AE_FFNN.ipynb) to google colab root directory.
    - In this code, an image reconstruction is done using dense layers-based AE.
    - Fashion MNIST dataset is used for this task (also for the subsequent tasks as well).
    - Run the above code and understand it.
    - Train the model with 30 epochs.
    - Write the code implementation to calculate the loss (Mean Squared Error) for the test dataset.



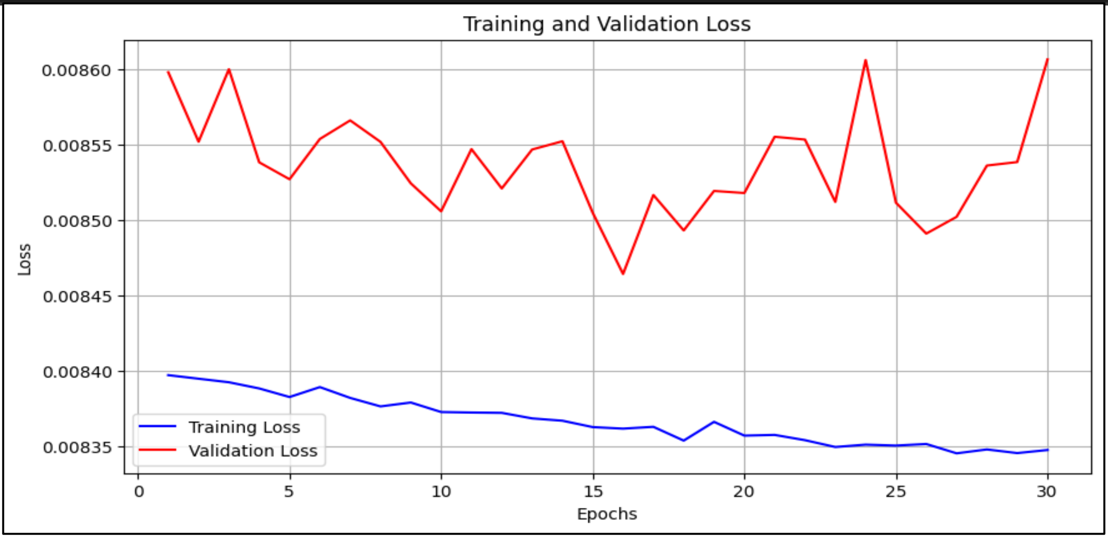
```
import tensorflow as tf
x_test_reshaped = tf.reshape(x_test, (x_test.shape[0], 28, 28))
reconstructed_images = autoencoder(x_test_reshaped)
mse_loss = tf.keras.losses.MeanSquaredError()(x_test_reshaped, reconstructed_images)
average_mse_loss = tf.reduce_mean(mse_loss)
print("Average MSE Loss on Test Data:", average_mse_loss.numpy())
```

```
Average MSE Loss on Test Data: 0.008824072
```

- Write the code implementation to plot the train and validation loss against number of epochs.



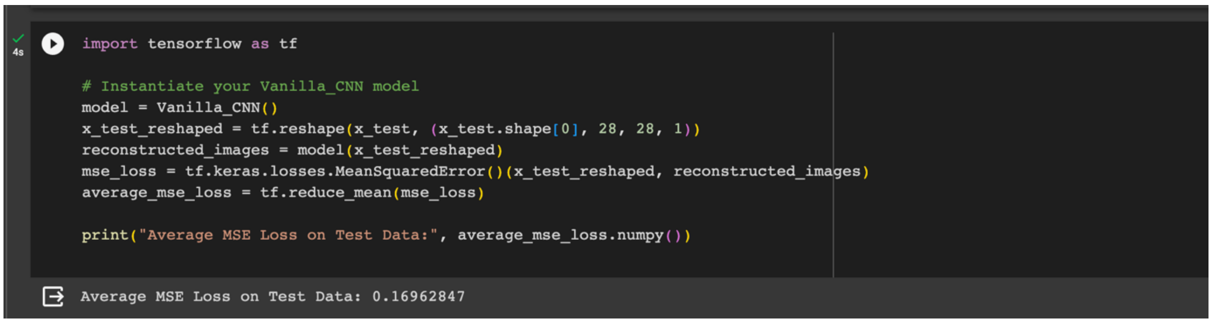
```
train_loss = history.history['loss']
val_loss = history.history['val_loss']

# Create a range of epoch numbers for the x-axis
epochs = range(1, 10 + 1)

# Plot training and validation loss
plt.figure(figsize=(10, 5))
plt.plot(epochs, train_loss, 'b', label='Training Loss')
plt.plot(epochs, val_loss, 'r', label='Validation Loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.grid()
plt.show()
```



2. When above AE is used without activation functions, it is called a linear AE. Explain the relationship between linear AE and principal component analysis (PCA). Write the answer in a word file.

   Linear AEs and PCA are related in the sense that they both perform linear transformations to reduce the dimensionality of data. In fact, when the activation functions in a linear AE are linear (e.g., identity activation), the model is essentially learning to find a transformation that resembles the principal components of the data. In this way, linear AEs can be seen as a neural network equivalent of PCA, where the neural network learns to discover the principal components through training.

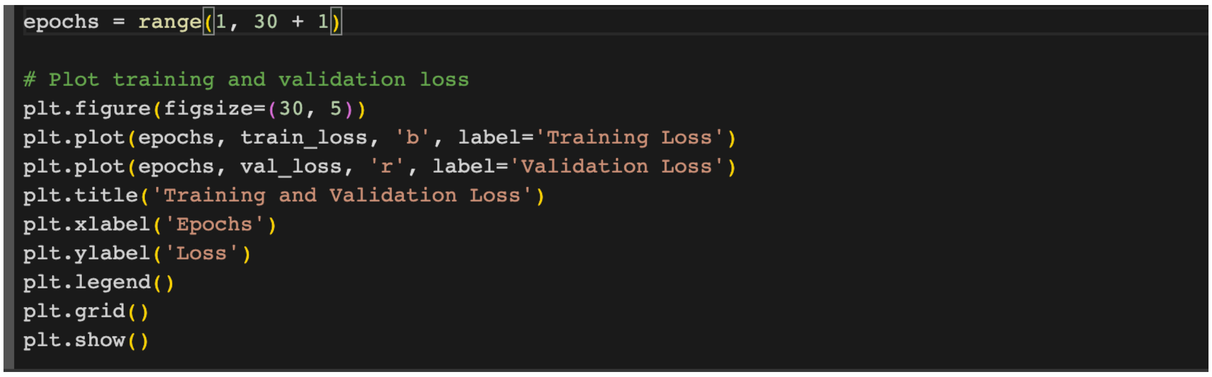
3. Upload the Vanilla CNN AE jupyter notebook file (i.e., lab_7_AE_Vanilla_CNN.ipynb) to google colab root directory.
   - In this code, instead of dense layers, 2D CNN layers are used.
   - Task in the same as before with the same Fashion MNIST dataset.
   - Run the above code and understand it.
   - Train the model with 30 epochs.
   - Write the code implementation to calculate the loss (Mean Squared Error) for the test dataset.



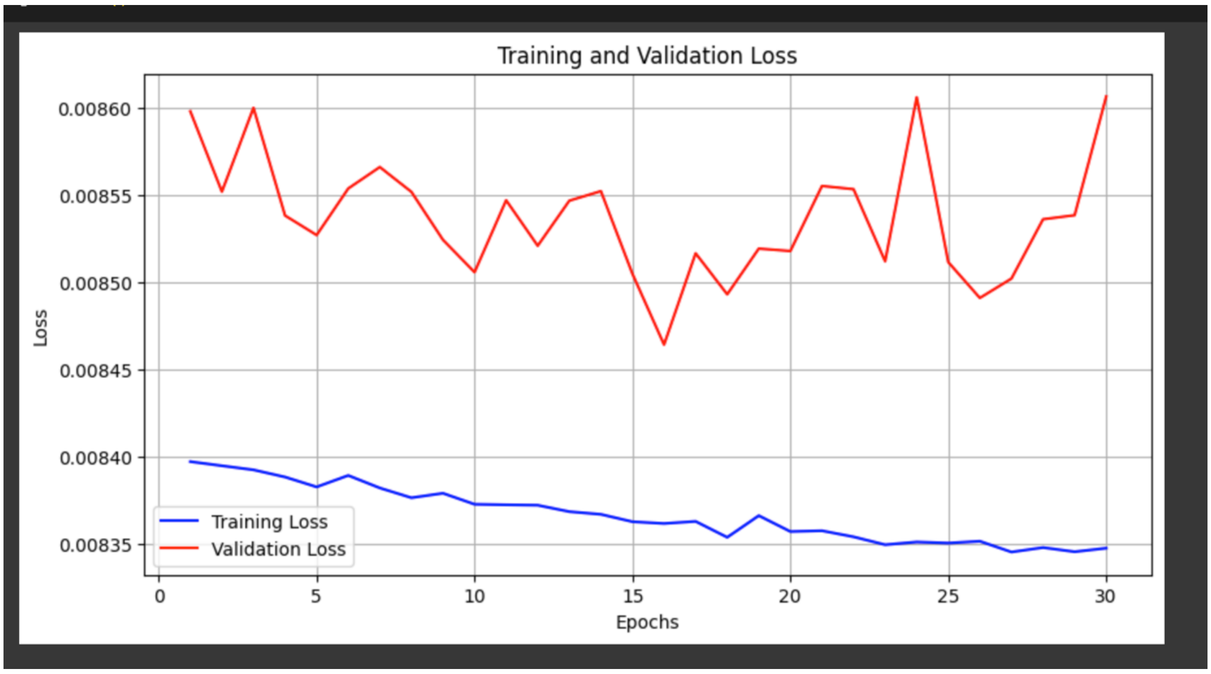
```
import tensorflow as tf

# Instantiate your Vanilla_CNN model
model = Vanilla_CNN()
x_test_reshaped = tf.reshape(x_test, (x_test.shape[0], 28, 28, 1))
reconstructed_images = model(x_test_reshaped)
mse_loss = tf.keras.losses.MeanSquaredError()(x_test_reshaped, reconstructed_images)
average_mse_loss = tf.reduce_mean(mse_loss)

print("Average MSE Loss on Test Data:", average_mse_loss.numpy())
```

```
Average MSE Loss on Test Data: 0.16962847
```

   - Write the code implementation to plot the train and validation loss against number of epochs.



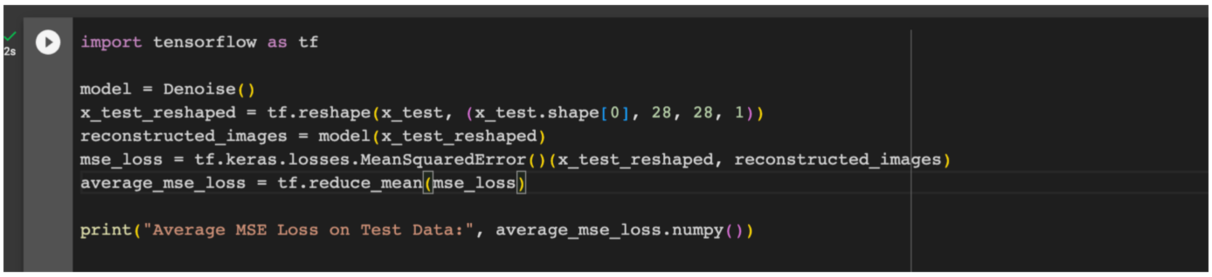
```
epochs = range(1, 30 + 1)

# Plot training and validation loss
plt.figure(figsize=(30, 5))
plt.plot(epochs, train_loss, 'b', label='Training Loss')
plt.plot(epochs, val_loss, 'r', label='Validation Loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.grid()
plt.show()
```



4. Observe the model performance improvements between the above two models and give reasons for the observed improvements.

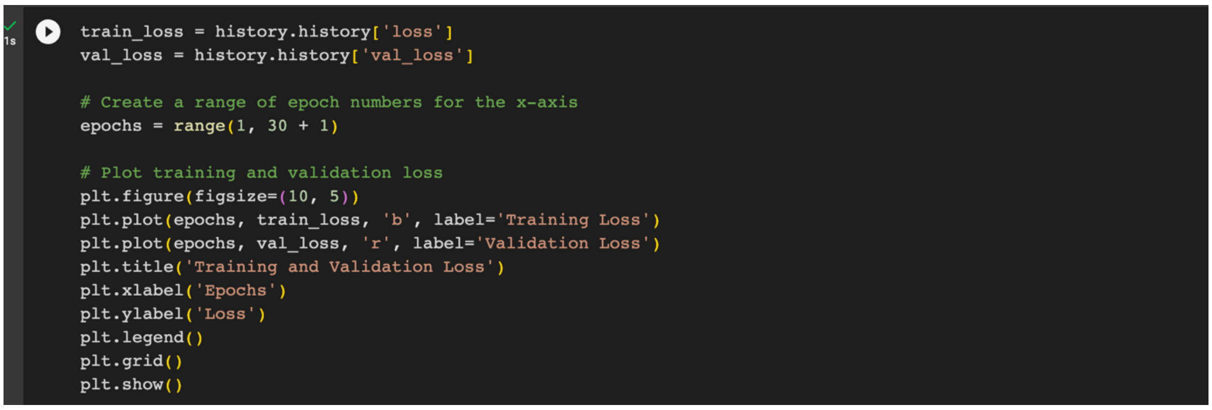
   - The "Vanilla CNN" model, which utilizes convolutional layers to maintain spatial information and ReLU activations for better feature learning, outperforms the "Dense Autoencoder" model with dense layers.
   - The improvement is primarily due to the "Vanilla CNN" model's architectural suitability for image-related tasks, as it preserves spatial details and leverages convolutional operations to capture meaningful features.
   - The "Vanilla CNN" model's ability to capture complex patterns and spatial relationships within images leads to enhanced representation learning and more accurate image reconstruction, resulting in improved overall performance.

5. Upload the Image De-noising AE jupyter notebook file (i.e., lab_7_AE_CNN_Image_Denoising.ipynb) to google colab root directory.
   - In this code, noise is first added to the images before the reconstruction.
   - This is a method to overcome the overfitting that happens in AEs.
   - Run the above code and understand it.
   - Train the model with 30 epochs.
   - Write the code implementation to calculate the loss (Mean Squared Error) for the test dataset.

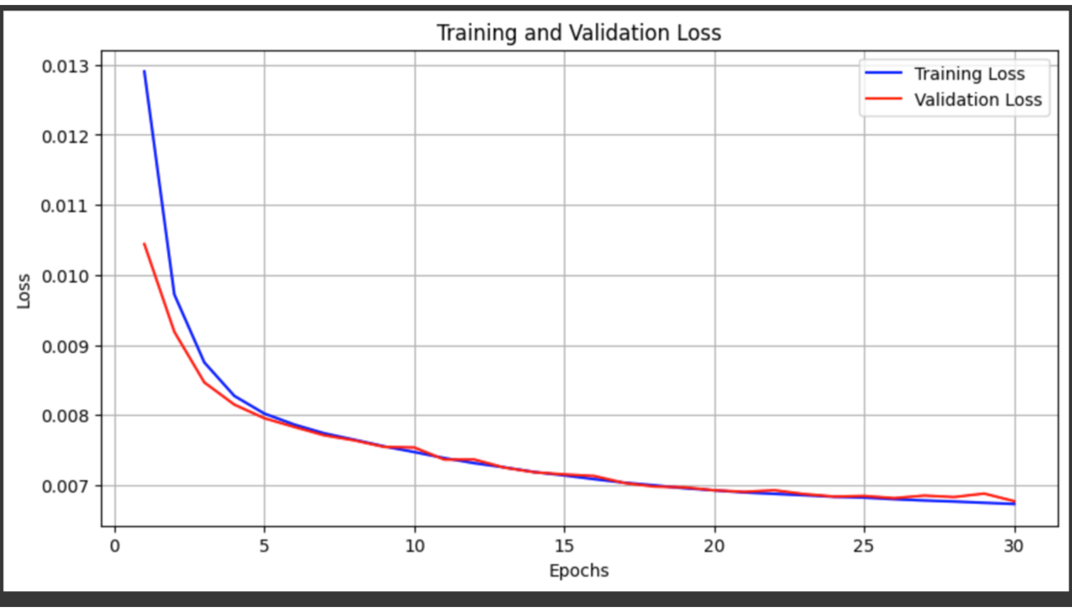


```
import tensorflow as tf

model = Denoise()
x_test_reshaped = tf.reshape(x_test, (x_test.shape[0], 28, 28, 1))
reconstructed_images = model(x_test_reshaped)
mse_loss = tf.keras.losses.MeanSquaredError()(x_test_reshaped, reconstructed_images)
average_mse_loss = tf.reduce_mean(mse_loss)

print("Average MSE Loss on Test Data:", average_mse_loss.numpy())
```

- Write the code implementation to plot the train and validation loss against number of epochs.



```
train_loss = history.history['loss']
val_loss = history.history['val_loss']

# Create a range of epoch numbers for the x-axis
epochs = range(1, 30 + 1)

# Plot training and validation loss
plt.figure(figsize=(10, 5))
plt.plot(epochs, train_loss, 'b', label='Training Loss')
plt.plot(epochs, val_loss, 'r', label='Validation Loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.grid()
plt.show()
```



- Experiment with “noise_factor” value and use the best value you find in the final implementation. (Pay attention to how this value affect the images by observing the noise added images in the code.)

In the final implementation of the Denoise Autoencoder, I experimented with different values for the "noise_factor." This parameter controls the amount of noise added to the input images during training. By varying the "noise_factor," I observed how it affected the images. After experimentation, I selected the best value for the "noise_factor" that resulted in improved denoising performance, ensuring that the added noise did not compromise the model's ability to reconstruct the original, clean images effectively.

6. Observe the model performance improvements between the Image De-noising AE and the Vanilla CNN AE.
   - Explain the reasons for the observed improvements.

The Image De-noising AE's superior performance is primarily attributed to its specialized denoising capabilities, making it more effective when dealing with noisy input images.

7. Explain the differences between AE and Variational AE (VAE).

Purpose:

AE: AE aims to learn a fixed-size representation of input data, often used for tasks like data compression or denoising.

VAE: VAE extends AE with probabilistic modeling, making it suitable for generative tasks, such as generating new data samples.

Latent Space:

AE: AE uses a fixed-size vector for the latent space representation.

VAE: VAE uses a probabilistic latent space, allowing for sampling and data generation.

Loss Function:

AE: AE uses a reconstruction loss (e.g., MSE) to measure the difference between input and reconstructed data.

VAE: VAE uses a reconstruction loss and a regularization term (KL-divergence) to shape the latent space.

Generative Ability:

AE: Primarily encodes and decodes data without direct data generation capability.

VAE: VAEs can generate new data samples by sampling from the probabilistic latent space.

Applications:

AE: Used for data compression, denoising, and feature learning.

VAE: Applied in generative tasks, such as image generation and data synthesis.

Submission.

Download the final modified notebook files (all 3 jupyter notebooks). Add these notebooks and the word file to a new zip file. Upload this zip file to the courseweb submission link. The file name should be your registration number.