DL lab 7 -Autoencoders

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

Relationship between Linear Autoencoder (AE) and Principal Component Analysis (PCA)

Linear Autoencoder (AE) and Principal Component Analysis (PCA) are both dimensionality reduction techniques that share similarities, especially when the autoencoder is used without activation functions. In this scenario, the AE becomes linear, and the relationship between the two techniques becomes clearer.

# Linear Autoencoder (AE)

An Autoencoder (AE) is a type of neural network that is trained to learn an identity function, mapping the input to the output. It consists of two main parts:

1. \*\*Encoder\*\*: Compresses the input into a lower-dimensional representation (latent space).

2. \*\*Decoder\*\*: Reconstructs the original input from this compressed representation.

When no activation functions are used in the encoder and decoder layers, the AE is linear, and the transformations performed are linear mappings of the input space.

# Principal Component Analysis (PCA)

PCA is a statistical method that is used to reduce the dimensionality of data while preserving as much variance as possible. It transforms the data into a new set of orthogonal axes (called principal components), where each successive component captures the maximum possible variance under the constraint that it is orthogonal to the previous components. This technique is based on linear transformations and provides a simple and efficient way to compress data.

# Relationship between Linear AE and PCA

When an Autoencoder is used without non-linear activation functions, it essentially performs a linear transformation similar to PCA. Both methods aim to find a lower-dimensional representation of the input data, and their behavior is closely related. Specifically, a linear AE:

1. \*\*Learns a linear mapping\*\* from the input to the latent space (encoder) and from the latent space back to the input (decoder).

2. \*\*Captures the same subspace\*\* as PCA, since both techniques reduce the dimensionality by projecting the input data into a lower-dimensional space.

3. \*\*Finds principal components\*\* implicitly. The weights of the linear encoder approximate the principal components of the input data, and the reconstruction of the input from the latent space is analogous to the reconstruction provided by PCA.

However, there are some differences:

1. \*\*Training Method\*\*: PCA is computed directly using matrix operations (e.g., eigendecomposition), while a linear AE is trained via backpropagation.

1. 2. \*\*Objective\*\*: PCA minimizes the reconstruction error by finding the principal components, while an AE minimizes
2. 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.
   * Write the code implementation to plot the train and validation loss against number of epochs.
3. Observe the model performance improvements between the above two models and give reasons for the observed improvements.
4. 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.
   * Write the code implementation to plot the train and validation loss against number of epochs.
   * 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.)
5. Observe the model performance improvements between the Image De-noising AE and the Vanilla CNN AE.
   * Explain the reasons for the observed improvements.

 **Image De-noising AE** improves performance because it is explicitly trained to handle noisy data, learning robust and generalized features that can handle real-world imperfections.

 **Vanilla CNN AE**, while effective for clean data, may struggle with noisy or unseen inputs because it does not undergo the same regularization and noise-handling training process.

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

AE

This encode reconstructed given input

VAE

Generate new data reason is generate new data thwre two vector zidma and relue

**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.