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.

**Similarities:**

* **Dimensionality Reduction:** Both techniques aim to reduce the number of features in a dataset while capturing most of the important information.
* **Linear Transformations:** When the encoder and decoder in an AE use linear activation functions (no activation at all), the model becomes equivalent to PCA. Both methods learn a linear transformation that projects data onto a lower-dimensional subspace.
* **Unsupervised Learning:** Both PCA and a linear AE are unsupervised learning techniques. They don't require labeled data for training.

**Differences:**

* **Optimization:** PCA finds the principal components by maximizing variance, while a linear AE optimizes the reconstruction loss (minimizing the difference between the input and reconstructed data).
* **Interpretability:** PCA's principal components are directly interpretable as linear combinations of the original features. In a linear AE, the latent representation (encoded data) might not be as easily interpretable.
* **Flexibility:** PCA is limited to linear transformations, while AEs with non-linear activation functions can capture more complex relationships in the data.

**In essence, a linear AE is a neural network implementation of PCA.** Both achieve similar goals with slightly different approaches.

1. 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.
2. Observe the model performance improvements between the above two models and give reasons for the observed improvements.

The second model, which uses a Convolutional Neural Network (CNN) architecture, is expected to demonstrate superior performance compared to the first model, which is a simple Autoencoder. Here are the reasons for this improvement:

**1. Feature Extraction**

**2. Hierarchical Feature Learning**

**3. Translation Invariance**

**4. Reduced Overfitting**

The CNN-based model leverages the strengths of CNN architectures, leading to improved feature extraction, hierarchical representation, translation invariance, and reduced overfitting. These factors contribute to its superior performance in image reconstruction tasks compared to the simple Autoencoder model.

**Key differences between the two models:**

**Architecture**: The first model is a simple Autoencoder with fully connected layers, while the second model is a CNN with convolutional and pooling layers.

**Feature Extraction**: The CNN model can automatically learn and extract relevant features from the image data, while the Autoencoder relies on hand-crafted features or learned features from the fully connected layers.

**Spatial Relationships**: The CNN model can capture spatial relationships and local features within the images due to its convolutional layers, while the Autoencoder might struggle to capture these relationships.

**Computational Efficiency**: CNNs can be computationally more efficient than Autoencoders, especially for large images, due to the use of convolutional operations and shared weights.

1. 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.)
2. Observe the model performance improvements between the Image De-noising AE and the Vanilla CNN AE.
   * Explain the reasons for the observed improvements.

**Analyzing Performance Improvements: Image De-noising AE vs. Vanilla CNN AE**

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

**Key Observations:**

**Task-Specific Architecture**

**Noise Handling**

**Feature Extraction**

**Reconstruction Focus**

**Potential Reasons for Improvements:**

**Task-Specific Architecture:** The Image De-noising AE's architecture, with its focus on denoising, is likely more suitable for this task compared to the Vanilla CNN. The specific design of the encoder and decoder layers might be better at handling noise and reconstructing the original image.

**Noise Handling:** The Image De-noising AE is specifically designed to handle noisy input data. It might learn to separate the noise from the underlying signal more effectively than the Vanilla CNN, leading to better denoising performance.

**Feature Extraction:** While both models use convolutional layers for feature extraction, the Image De-noising AE's architecture might be more effective at extracting features that are relevant for denoising. This could be due to the specific design of the convolutional layers or the training process.

**Reconstruction Focus:** The Image De-noising AE's decoder is designed to reconstruct the original image from the encoded representation. This focus on reconstruction can lead to better performance in denoising tasks, as the model is specifically trained to produce output images that are similar to the original clean images.

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

**Autoencoder (AE) vs. Variational Autoencoder (VAE)**

Autoencoders (AEs) and Variational Autoencoders (VAEs) are both unsupervised learning techniques used for dimensionality reduction and generative modeling. While they share the same basic structure of an encoder and a decoder, there are significant differences in their underlying principles and objectives.

**Autoencoder (AE)**

**Objective**: To learn a compressed representation of the input data that can be used to reconstruct the original data.

**Encoding**: The encoder maps the input data to a latent space.

**Decoding**: The decoder maps the latent representation back to the original input space.

**Loss Function**: Typically uses a reconstruction loss, such as mean squared error (MSE) or binary cross-entropy, to minimize the difference between the reconstructed and original data.

**Latent Space**: The latent space is not explicitly constrained, and the encoder can learn any representation.

**Variational Autoencoder (VAE)**

**Objective**: To learn a probabilistic model of the data distribution.

**Encoding**: The encoder maps the input data to a latent space, but it also outputs a mean and variance for each dimension of the latent representation.

**Decoding**: The decoder samples from the latent distribution and maps the sampled latent representation back to the original input space.

**Loss Function**: Uses a combination of reconstruction loss and a regularization term called the Kullback-Leibler (KL) divergence. The KL divergence ensures that the latent distribution is close to a prior distribution (often a standard normal distribution).

Latent Space: The latent space is constrained to follow a specific distribution (typically a Gaussian distribution). This constraint encourages the VAE to learn a meaningful latent representation.

**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.