**Lab 07**

Question 02

Linear Autoencoders (AEs) and Principal Component Analysis (PCA) are closely related because they both aim to reduce the dimensionality of data while preserving as much information as possible. However, they achieve this in different ways, especially in how they are implemented and the assumptions they make.

**Key Points of Comparison Between Linear AE and PCA:**

1. **Linear Transformation**:
   * In a **linear autoencoder**, if no activation functions are used, the encoder and decoder consist of simple linear transformations. The network essentially learns a linear mapping from the input space to a lower-dimensional space and then reconstructs the input from this compressed representation.
   * In **PCA**, a similar linear transformation is performed. PCA projects data onto a set of orthogonal vectors (principal components) that maximize the variance in the data. These components represent directions of maximum variance in the data.
2. **Objective**:
   * A **linear autoencoder** aims to minimize the reconstruction error, typically using a loss function like **Mean Squared Error (MSE)** between the original input and the reconstructed output.
   * **PCA** explicitly seeks to find a set of orthogonal axes (principal components) that capture the most variance in the data. This can also be viewed as minimizing the reconstruction error, but in terms of variance captured.
3. **Dimensionality Reduction**:
   * Both methods reduce the dimensionality of the data. In a **linear AE**, the encoder maps the input to a lower-dimensional latent space. PCA reduces dimensionality by projecting the data onto a lower-dimensional subspace formed by its principal components.
   * In both cases, the latent space in a linear AE can represent the same subspace found by PCA, if the autoencoder has no activation functions and is trained optimally.
4. **Reconstruction**:
   * In a **linear AE**, the decoder reconstructs the input by transforming the latent representation back to the original space. The aim is to recreate the input as accurately as possible.
   * **PCA** reconstructs data by combining the principal components with their corresponding coefficients (the projection of the data onto these components). This process results in an approximation of the original data based on the reduced representation.
5. **Relationship**:
   * In theory, if a **linear AE** (with no activation functions) is trained correctly, it will learn the same subspace as PCA. In fact, it is mathematically equivalent to PCA if:
     + The **linear AE** has the same number of neurons in the bottleneck layer (the compressed representation) as the number of principal components you want to extract.
     + The training objective is set to minimize the squared reconstruction error, which aligns with PCA's objective of capturing maximum variance.
   * The key difference lies in the method of optimization: **PCA** uses an eigenvalue decomposition or Singular Value Decomposition (SVD) to analytically solve for the principal components, while **linear AE** uses backpropagation and gradient descent to minimize the reconstruction loss.

Question 04

**Reasons for Performance Differences:**

1. **Spatial Information in CNN Autoencoder**:
   * **CNN Autoencoder**: Convolutional layers are designed to capture spatial hierarchies and local patterns in data, which are especially useful for images. In the case of the Fashion MNIST dataset, which contains images of clothing items, the spatial relationships (like edges, textures, and patterns) are crucial for understanding the structure of the images. The CNN layers are better equipped to capture these spatial features than dense layers.
   * **Dense Autoencoder**: In contrast, dense layers treat every pixel independently, ignoring the local spatial structure of the data. As a result, dense autoencoders often fail to capture the intricate relationships between neighboring pixels, leading to poorer reconstruction of images.

**Reason for Improvement**: CNN-based autoencoders perform better on image data because they preserve the spatial structure of the data, leading to better reconstructions with lower loss values.

1. **Parameter Sharing and Efficiency**:
   * **CNN Autoencoder**: Convolutional layers share weights across different spatial locations in the image. This weight sharing allows CNNs to learn more efficient representations with fewer parameters, leading to faster convergence and more generalizable models.
   * **Dense Autoencoder**: Dense layers require separate weights for every connection between neurons, which can lead to a larger number of parameters and more difficulty in learning efficient representations.

**Reason for Improvement**: CNNs are more parameter-efficient for image data, allowing the model to learn faster and generalize better to unseen data.

1. **Overfitting and Generalization**:
   * **CNN Autoencoder**: Due to the shared weights and the ability to capture local features, CNNs tend to generalize better, even with fewer training examples. They are less prone to overfitting compared to dense networks when trained on image data.
   * **Dense Autoencoder**: Dense networks, with their larger number of parameters and lack of spatial context, can overfit to the training data more easily, especially if the training data is limited or noisy.

**Reason for Improvement**: CNNs provide better generalization, reducing overfitting and making them more robust to unseen data compared to dense networks.

1. **Hierarchical Feature Learning**:
   * **CNN Autoencoder**: CNNs learn hierarchical features—starting from simple edges and textures in the lower layers and gradually building up to more complex shapes and patterns in the deeper layers. This hierarchical feature learning is crucial for image data, where understanding high-level structures is important for reconstruction.
   * **Dense Autoencoder**: Dense layers do not learn hierarchical features in the same way. Instead, they tend to learn global relationships, which may not be as effective for capturing the fine details of an image.

**Reason for Improvement**: CNNs' ability to learn hierarchical features leads to better performance in tasks involving image data.

Question 06

The Image De-noising Autoencoder (AE) outperforms the Vanilla CNN AE in several key ways:

* Better Generalization: The de-noising AE is trained for de-noising of images and therefore more suitable for noisy and or imperfect data as compared to the vanilla CNN AE that is only trained to reconstruct clean data.
* Robustness to Noise: That is why de-noising AEs are equipped in order to work with noisy inputs, and in fact the reconstructions with such inputs are closer to true data while the noisy input affects the vanilla CNN AE.
* Improved Feature Learning: De-noising auto-encoder learns higher order, abstract representation to distinguish noise from data and reconstruct the data with more efficacy than a normal CNN AE which may focus at pixel level.
* Regularization: The task of de-noising serves as regularizer that reduces over-reliance on noise content and make de-noising AE perform better than vanilla CNN AE.
* Thus, the de-noising AEs do well because they learn to delete noise, generalize better, and provide better reconstructions especially in noisy surroundings.

Question 07

* **Latent Space**:
* AE: Encodes input to a fixed point in latent space.
* VAE: Encodes input to a probability distribution (mean and variance), encouraging smoothness in the latent space.
* **Objective**:
* AE: Minimizes reconstruction loss only.
* VAE: Minimizes both reconstruction loss and KL Divergence, regularizing the latent space to follow a Gaussian distribution.
* **Generative Capability**:
* AE: Not generative, reconstructs specific inputs.
* VAE: Generative, can sample new data from the latent space.
* **Stochasticity**:
* AE: Deterministic encoding-decoding process.
* VAE: Stochastic, sampling from distributions during decoding.
* **Regularization**:
* AE: No latent space regularization.
* VAE: Regularizes the latent space for better generative properties.