**Student ID: IT21279966**

**Batch: Y1.S1.WE.SE.01.01**

**Name: Gunawardana S.D.L**

A graph with numbers and lines

Description automatically generatedNum of epochs = 10,

A graph with orange lines

Description automatically generatedNum of epochs = 30,

|  |  |  |
| --- | --- | --- |
| **Epoch** | **Test Data Loss** | **Validation Data Loss** |
| 01 | 0.0085 | 0.0088 |
| 05 | 0.0086 | 0.0088 |
| 10 | 0.0086 | 0.0087 |
| 15 | 0.0085 | 0.0087 |
| 20 | 0.0086 | 0.0087 |
| 25 | 0.0085 | 0.0087 |
| 30 | 0.0085 | 0.0086 |

1. Explain the relationship between linear AE and principal component analysis (PCA).

Both linear autoencoders and PCA are linear techniques used for dimensionality reduction. They both aim to find a lower dimensional representation of the data that captures the most important features.

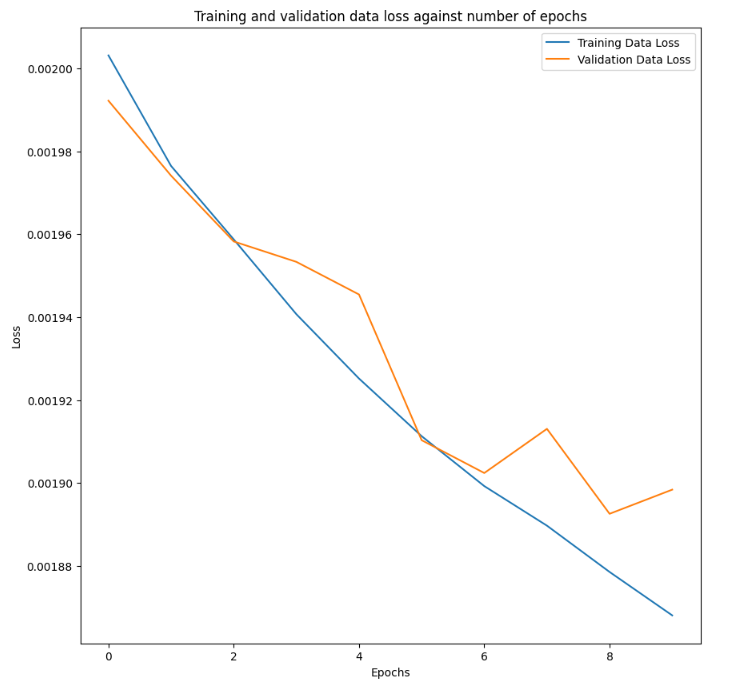
The key difference between the two is that PCA is a linear transformation that projects the data onto a new feature space, whereas a linear autoencoder is a neural network that learns to compress and reconstruct the data.

Similarities,

* Both are linear techniques used for dimensionality reduction.
* Both aim to find a lower-dimensional representation of the data that captures the most important features.
* Both can be used for data compression and feature extraction.

Differences,

* PCA is a linear transformation, whereas a linear autoencoder is a neural network.
* PCA is a non-parametric technique, whereas a linear autoencoder is a parametric technique.
* PCA is typically faster and more efficient to compute than a linear autoencoder.

Num of epochs = 10,

A line graph with blue and orange lines

Description automatically generatedNum of epochs = 30,

|  |  |  |
| --- | --- | --- |
| **Epoch** | **Test Data Loss** | **Validation Data Loss** |
| 01 | 0.0019 | 0.0019 |
| 05 | 0.0018 | 0.0018 |
| 10 | 0.0018 | 0.0018 |
| 15 | 0.0018 | 0.0018 |
| 20 | 0.0017 | 0.0018 |
| 25 | 0.0017 | 0.0017 |
| 30 | 0.0017 | 0.0018 |

|  |  |
| --- | --- |
| **Model 01 – Dense Layers** | **Model 02 – CNN Layers** |
| The fully connected autoencoder flattens the image data, losing the spatial relationships between pixels. This can make it less effective for tasks where spatial patterns are important. | Convolutional layers preserve spatial information by applying filters that capture local patterns. This is crucial for image data, leading to more meaningful and efficient feature extraction. |
| uses fully connected layers, which require a large number of parameters to process the flattened data, making it harder to generalize and more prone to overfitting. | uses convolutional layers, which are more parameter efficient since they use shared weights and small receptive fields, allowing the model to capture essential features with fewer parameters. |
| has higher test and validation losses, with values stabilizing around 0.0085 | achieves much lower loss values, stabilizing around 0.0017. The lower loss suggests that Model 02 can reconstruct the input data with significantly less error. |
| Not much suitable for image-based tasks | more suitable for image-based tasks |

Model 02 (CNN) generalizes better across both training and validation data, which is evident from the closer alignment of training and validation loss curves. This indicates better performance in avoiding overfitting compared to Model 01.

Epochs = 10, Noise\_factor = 0.2

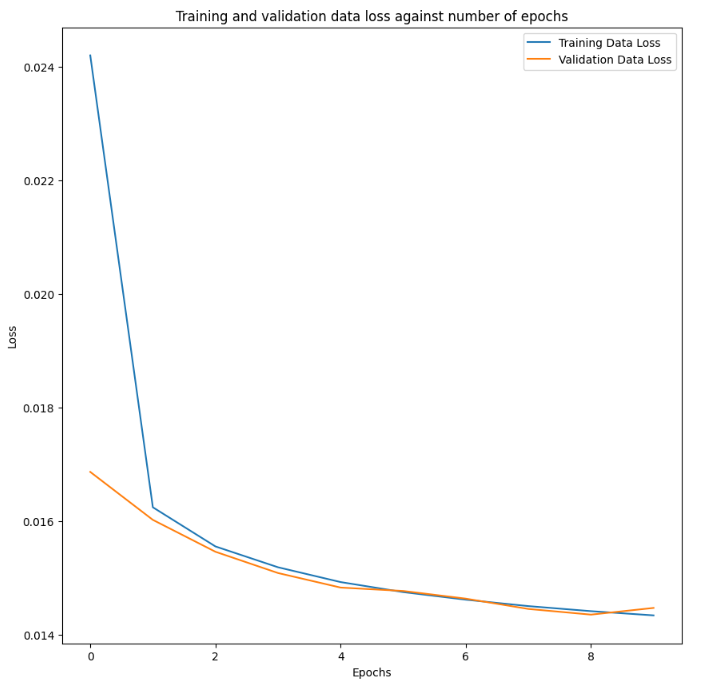
A graph with a line

Description automatically generatedA collage of different clothing

Description automatically generated

A collage of images of clothing

Description automatically generatedEpochs = 10, Noise\_factor = 0.4



Epochs = 10, Noise\_factor = 0.1

A collage of different clothing

Description automatically generated

A graph of a graph

Description automatically generated

Epochs = 10, Noise\_factor =0.05,  
  
A collage of different types of clothing

Description automatically generated

A graph of a graph

Description automatically generated

Epochs = 10, Noise\_factor =0.01,

A collage of different types of clothes

Description automatically generated

A graph of a number of epoxy

Description automatically generated

Epochs = 30, Noise\_factor =0.1,

A collage of different types of clothes

Description automatically generated

A graph of a graph

Description automatically generated

**Noise factor**

A lower noise factor value will introduce less noise, resulting in cleaner images and possibly less reconstruction effort for the autoencoder. However, it might not be challenging enough for the network to learn meaningful representations for denoising.

A higher noise factor value might introduce too much noise, leading to a significant loss in image quality, which the autoencoder might struggle to denoise effectively.

The Image De-noising autoencoder shows a significant improvement over the Vanilla CNN autoencoder, especially in the early epochs. While the Vanilla CNN model's loss reduces only slightly over time, the Image De-noising model quickly achieves lower test and validation losses, stabilizing at around epoch 10.

This improvement can be attributed to the de-noising autoencoder's ability to learn more robust representations, as it is trained to remove noise from the data. This additional task helps the model focus on important features, leading to better generalization and performance compared to the simpler reconstruction task of the Vanilla CNN autoencoder.

|  |  |
| --- | --- |
| **Autoencoders** | **Variational Autoencoders (VAEs)** |
| Encoder maps input to fixed latent space | Encoder maps input to probabilistic latent space (mean and variance) |
| Latent space is fixed and deterministic | Latent space is probabilistic and stochastic |
| Decoder reconstructs input from latent space | Decoder reconstructs input from sampled latent space |
| Limited to reconstructing input data | Can generate new samples from learned distribution |
| Difficult to interpret latent space | Latent space is more interpretable due to probabilistic nature |