Documentation - Generative Artificial Intelligence Algorithms

1. Principal Component Analysis (PCA)

a. Data Loading and Initial Exploration

- Loaded the MNIST dataset from torchvision.datasets using Python.
- Created functions to visualize the handwritten digits.

b. Covariance Matrix

- Computed the covariance matrix, `covMatrix`, for the standardized dataset using `np.cov()`.
- Provides insights into relationships and variances between features.

c. Eigenvalues and Eigenvectors

- Computed eigenvalues and eigenvectors using the covariance matrix: `eigenvalues, eigenvectors = np.linalg.eig(np.cov())`.
- Eigenvalues represent variance captured by each principal component, while eigenvectors represent their direction.
- Sorted eigenvalues and eigenvectors in descending order for dimensionality reduction.

d. Significance of Eigenvalues

- Eigenvalues quantify variance captured by each principal component.
- Larger eigenvalues indicate more information retained from original data.

e. Sorting Eigenpairs

- Sorted eigenvalue-eigenvector pairs based on eigenvalues in descending order.
- Ensures principal components are arranged by significance in capturing variance.

f. Dimensionality Reduction

- Retained principal components with higher eigenvalues for effective dimensionality reduction while preserving most of the dataset's variance.
- Equivalent to finding the Latent Representation of the original image in a lower-dimensional space.

g. Number of Principal Components

- Determines dimensionality of the Latent Space for finding Latent representation.

h. Reconstruction

- Obtained original image from its Latent Representation using the transpose of the Top-K Eigenvector Matrix for dimensionality reduction.
- `original = latent @ V.T + X_mean`

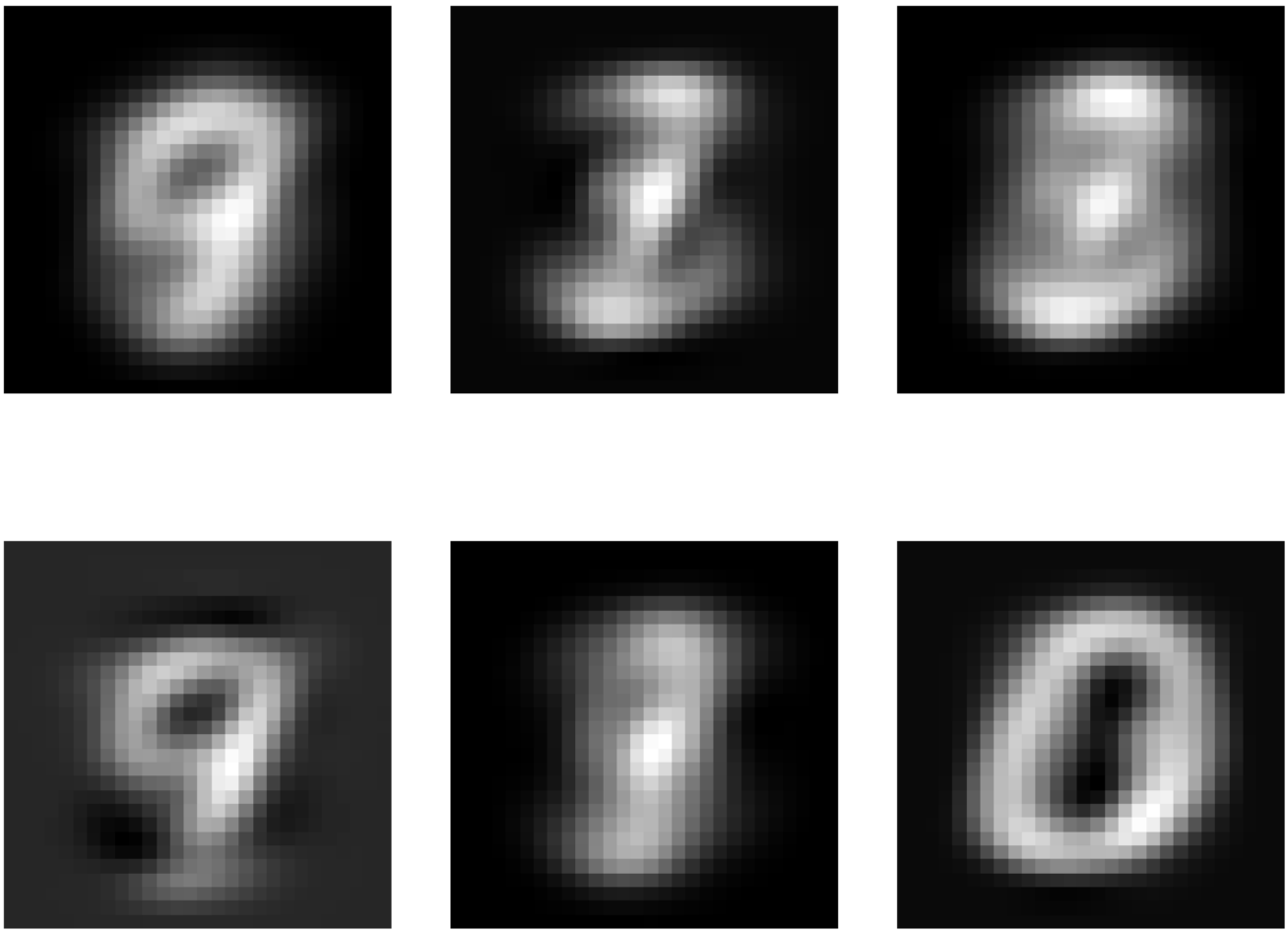
i. Visualization

- Sampled 6 images from the MNIST dataset, plotted them, converted them to their latent representation, and reconstructed them.

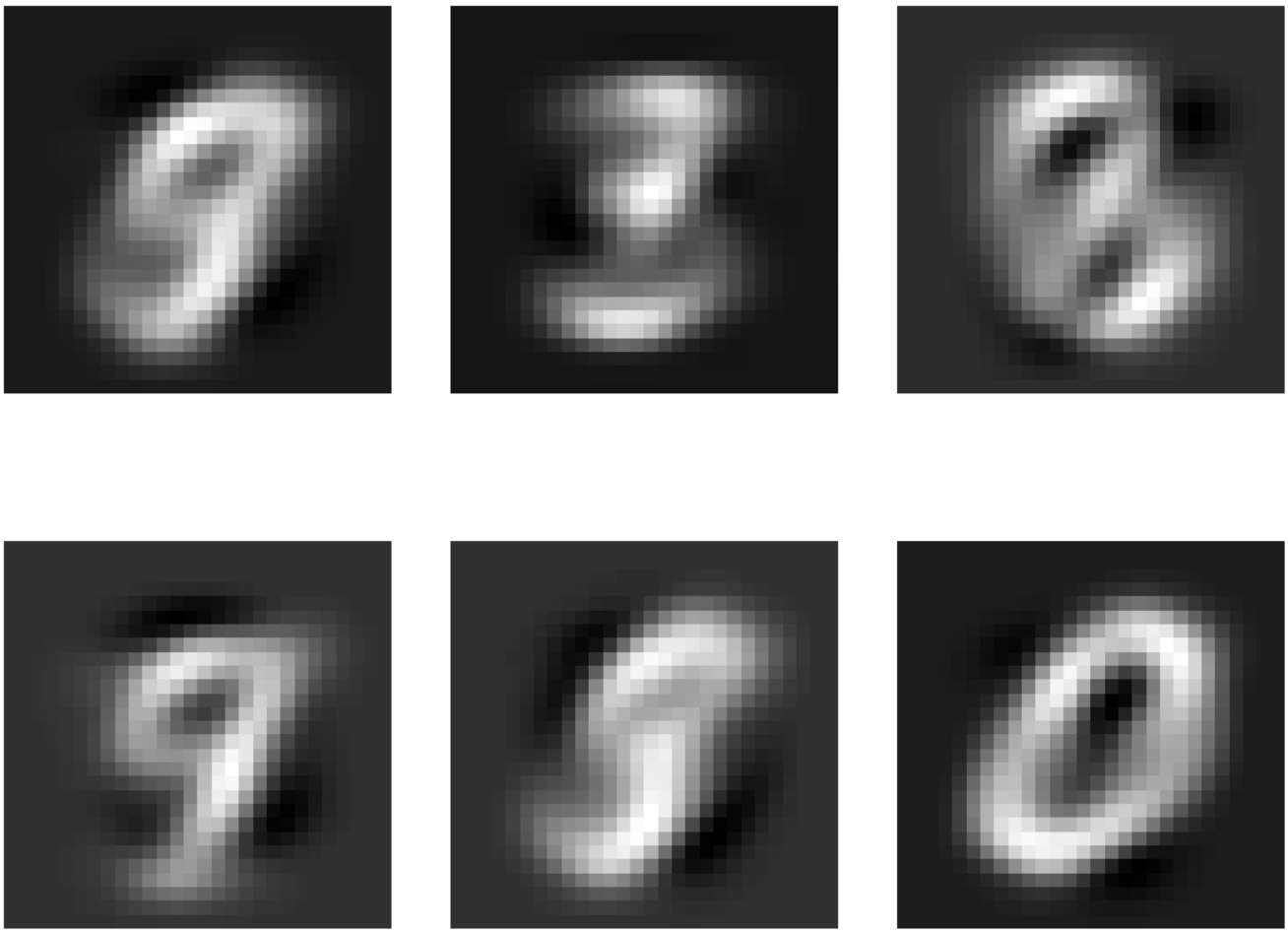
Original Images

Reconstructed Images

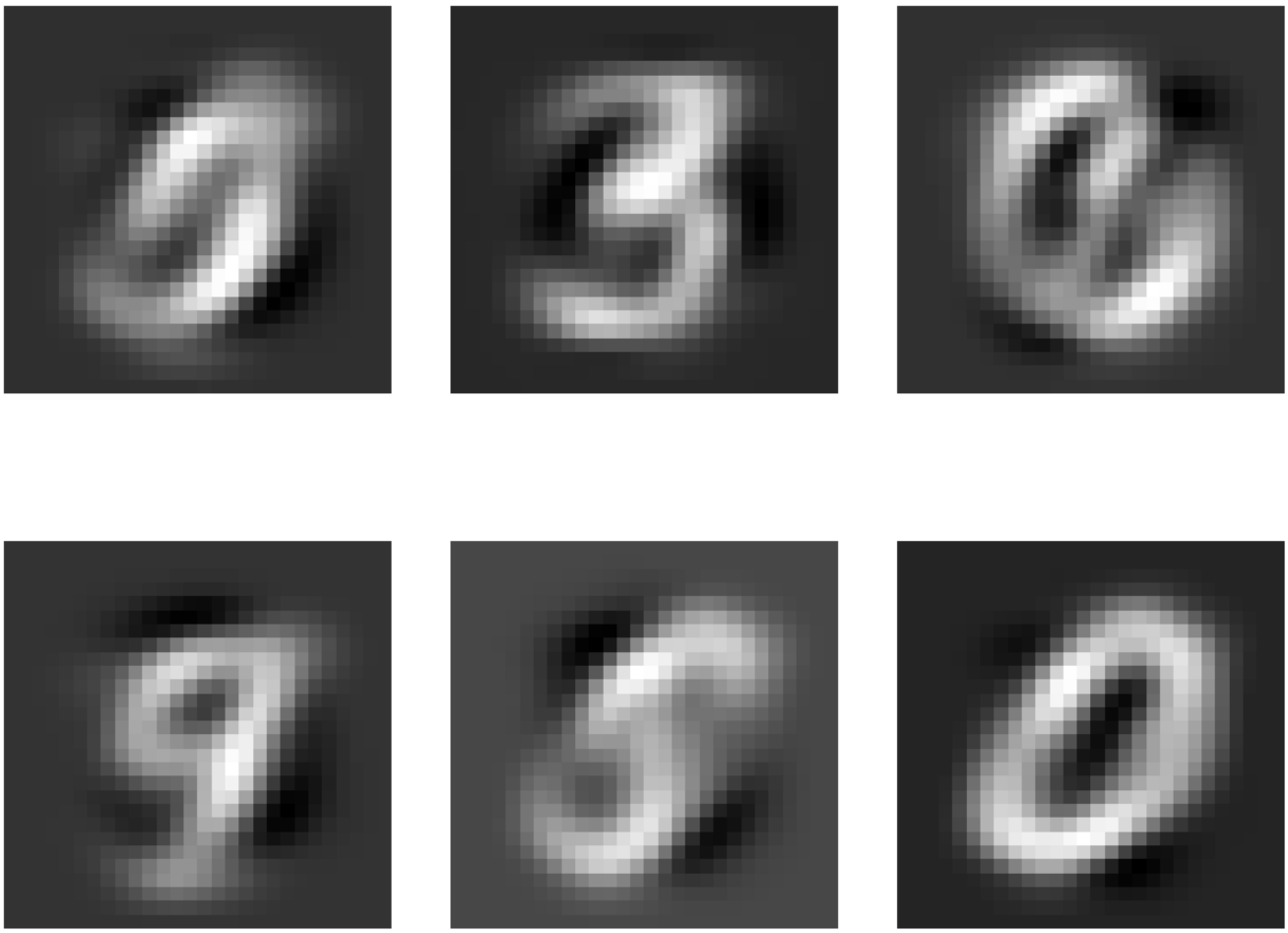
2 Dimensional Latent Representation



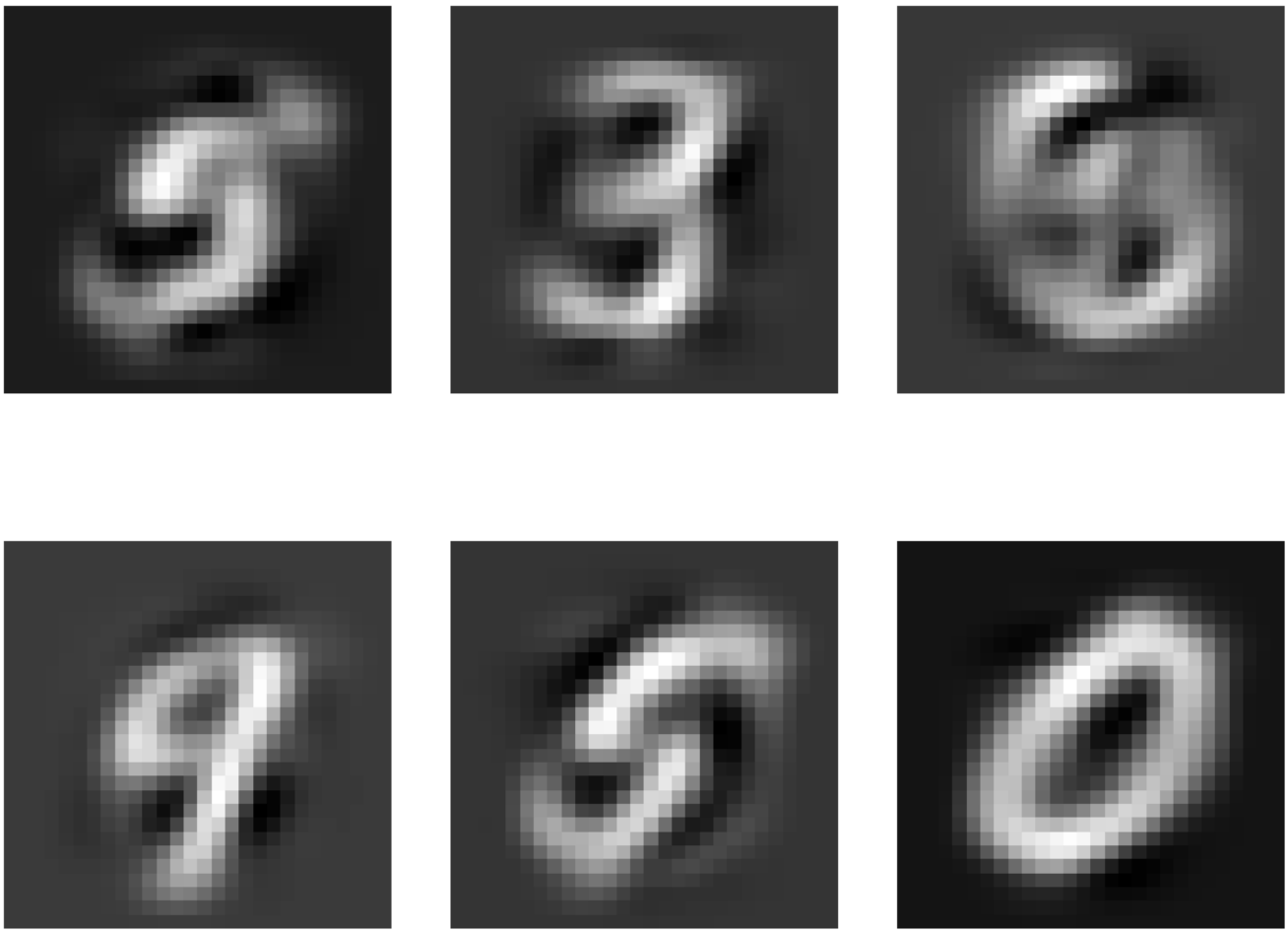
4 Dimensional Latent Representation



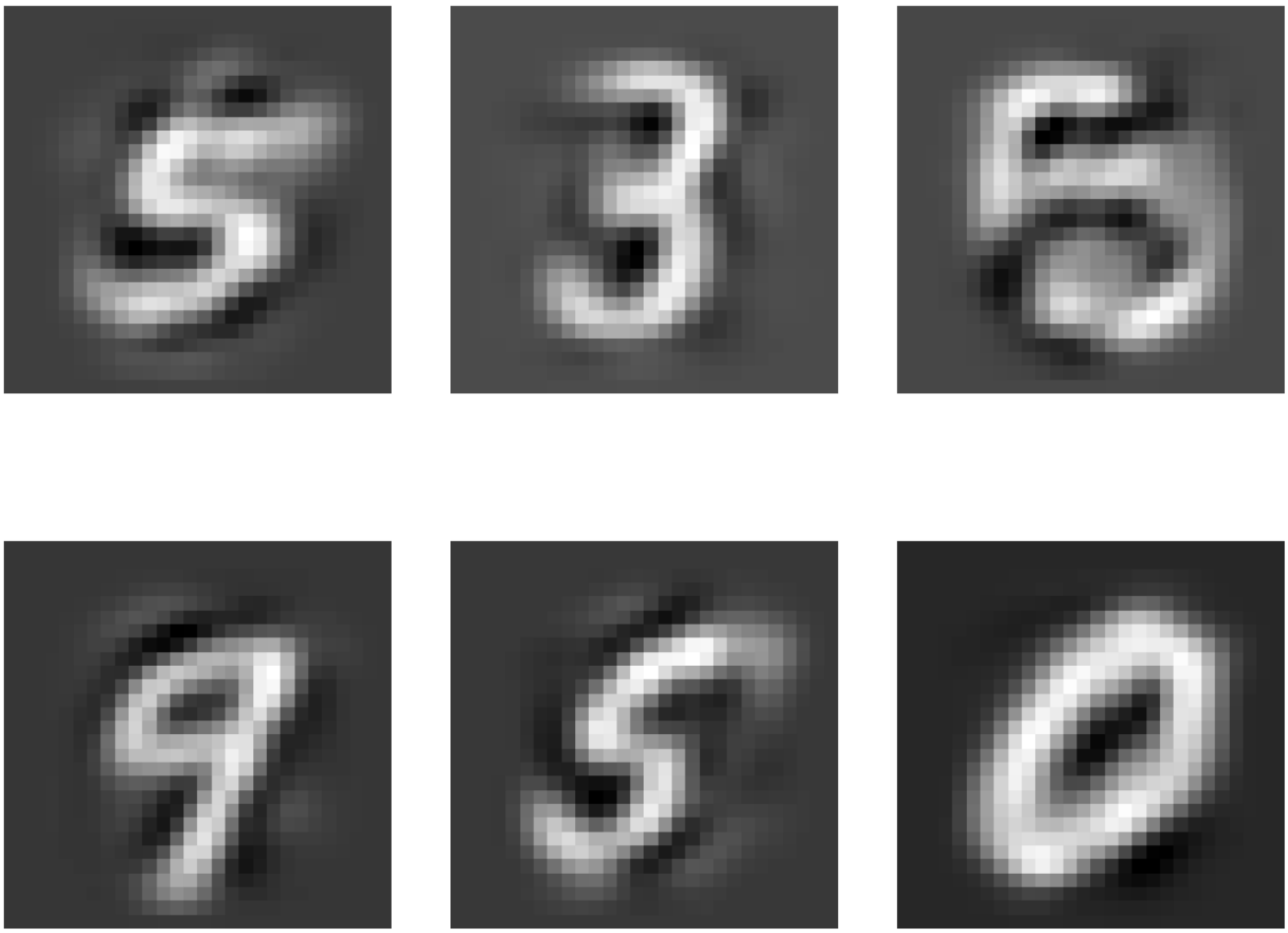
8 Dimensional Latent Representation



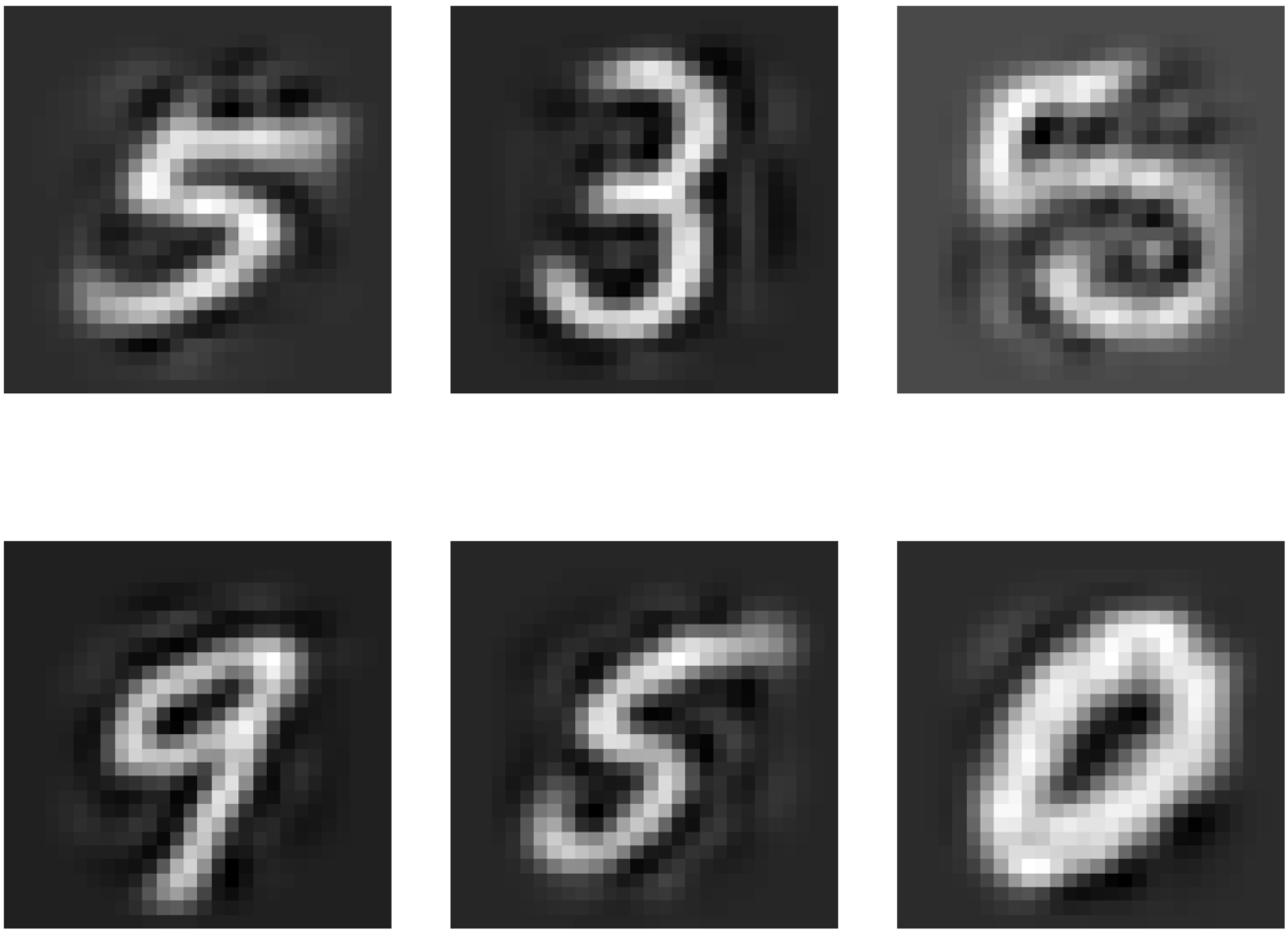
16 Dimensional Latent Representation



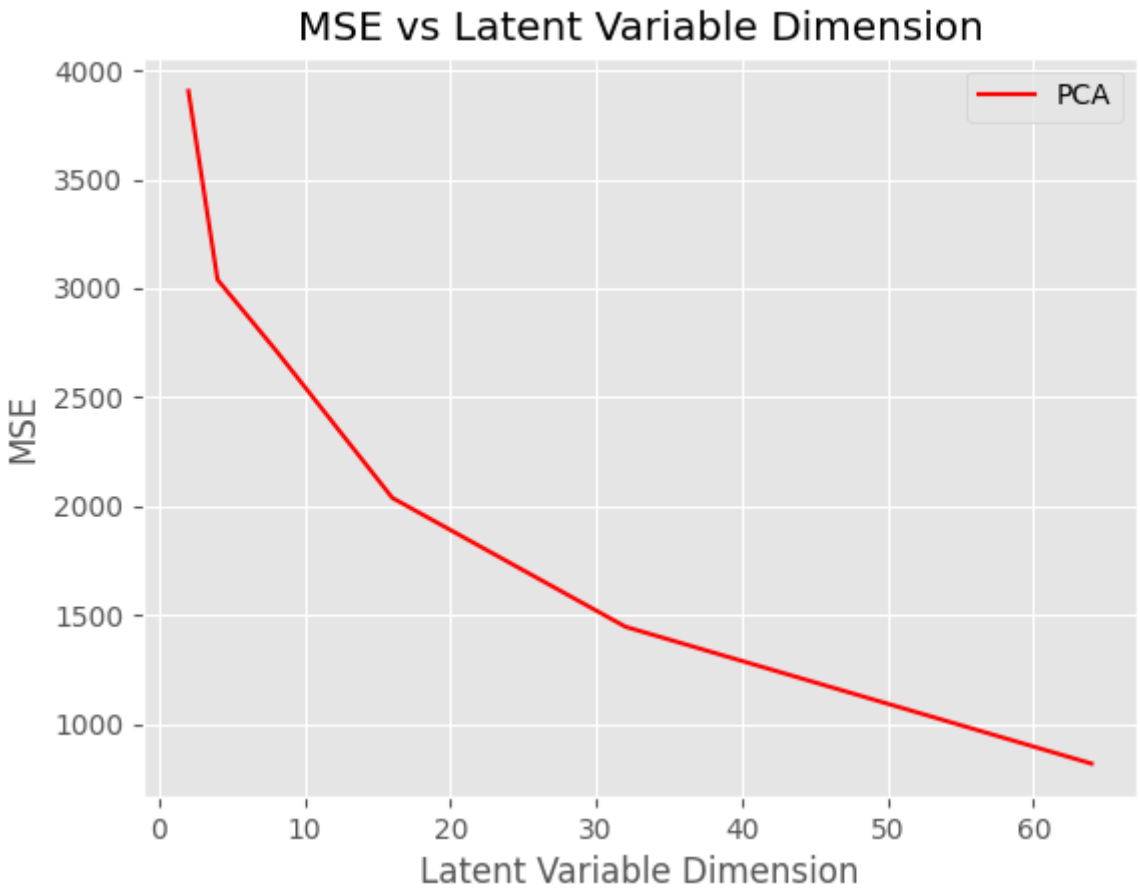
32 Dimensional Latent Representation



64 Dimensional Latent Representation



Reconstruction MSE vs Latent Dimension



2. Probabilistic Principal Component Analysis (PPCA)

a. Data Loading and Initial Exploration

- Loaded the MNIST dataset from torchvision.datasets using Python.
- Created functions to visualize the handwritten digits.

b. Covariance Matrix

- Computed the covariance matrix, `covMatrix`, for the standardized dataset using `np.cov()`.
- Provides insights into relationships and variances between different features.

c. Eigenvalues and Eigenvectors

- Computed eigenvalues and eigenvectors using the covariance matrix: `eigenvalues, eigenvectors = np.linalg.eig(np.cov())`.
- Sorted eigenvalue-eigenvector pairs based on eigenvalues in descending order.

d. Sorting Eigenpairs

- Ensured principal components are arranged by their significance in capturing variance.

e. Dimensionality Reduction

- Modeled latent representations to follow a Standard Normal Distribution.

- Calculated conditional probability of an image given a latent variable as a Gaussian Distribution linearly projected around the latent variable.

f. The optimal W

- Calculated analytically as: `Wml = Um @ np.sqrt(Lm - (sigma**2) * np.identity(z_dim)) @ R`, where Um are chosen K eigenvectors, Lm are corresponding eigenvalues, and R is an orthogonal rotation matrix.

g. Latent Variable Calculation

- Obtained latent representation of an image.
- The reconstructed image follows a Gaussian Distribution described by mean and covariance.
- `z ~ N(mean, C)`
- `mean = np.linalg.inv(M) @ Wml.T @ ((t - X_mean).T).flatten()`
- where `M` and `C` are calculated as:
- `M = Wml.T @ Wml + (sigma**2) *np.identity(z_dim)`
- `C = (sigma**2) *np.linalg.inv(M)`

h. Reconstruction

- Obtained original image from its latent representation using the Wml Matrix for dimensionality reduction.
- `reconstructed = latent @ Wml.T + X_mean`

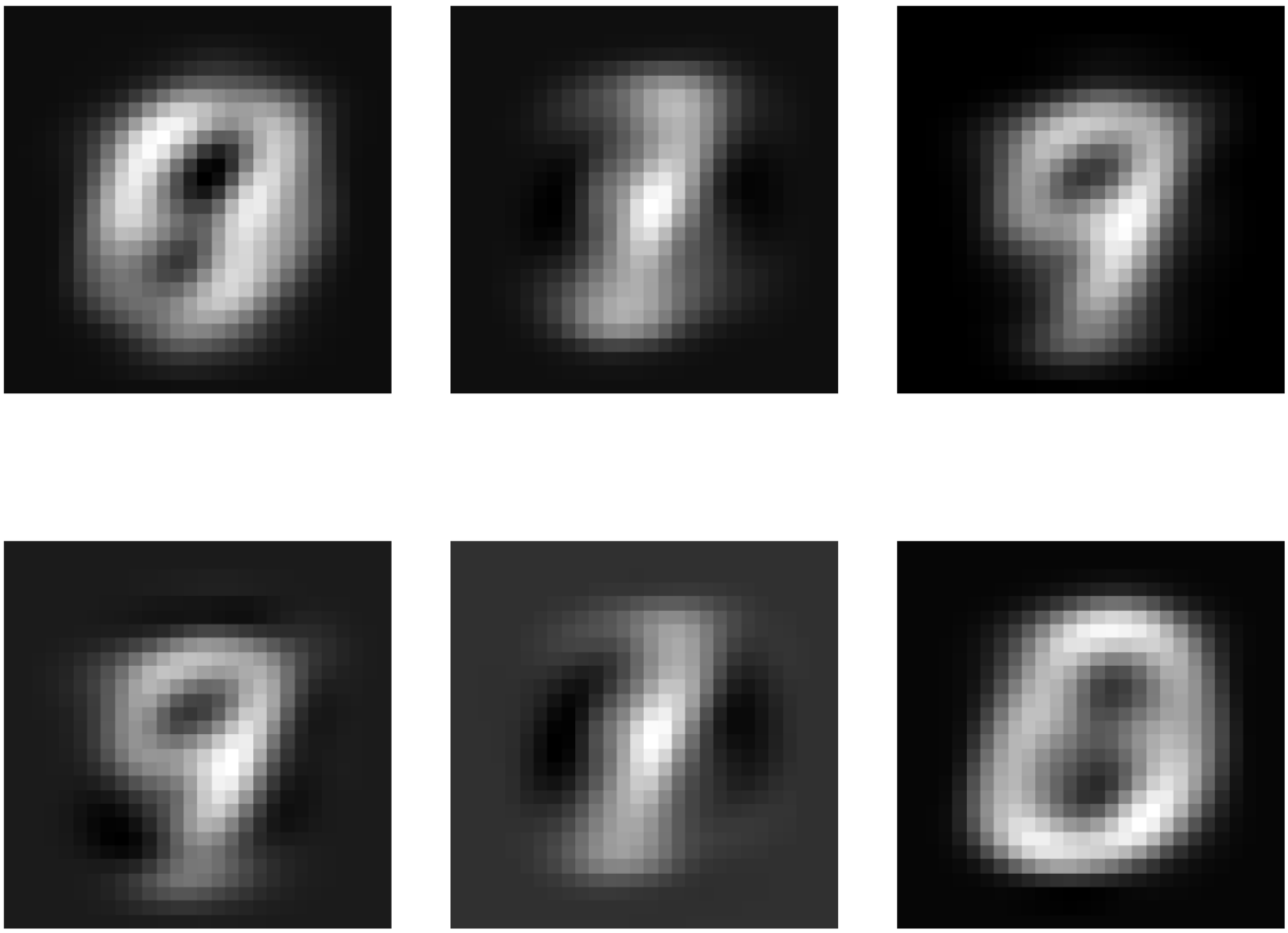
i. Visualization

- Sampled 6 images from the MNIST dataset, plotted them, converted them to their latent representation, and reconstructed them.

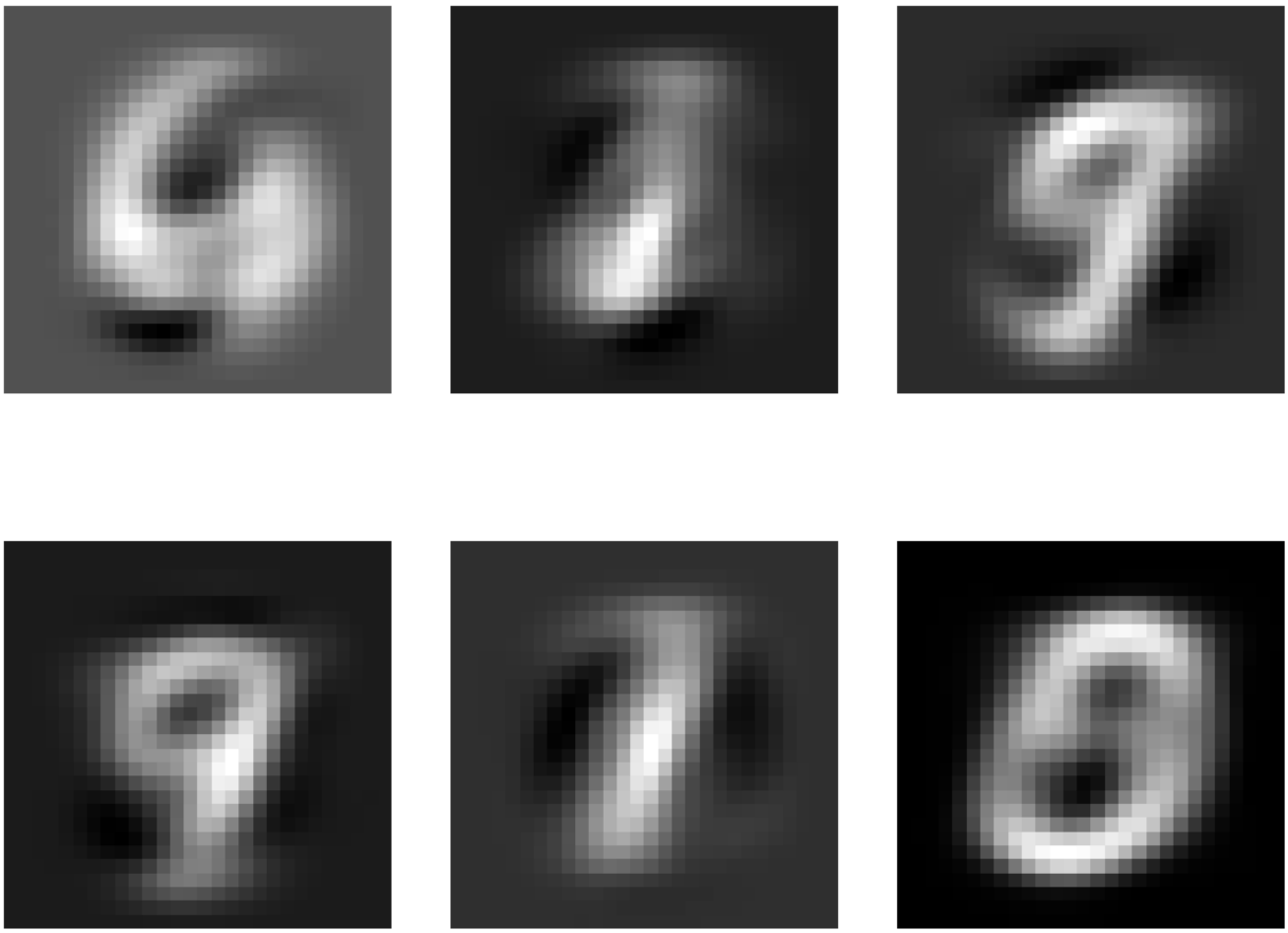
Original Images

Reconstructed Images

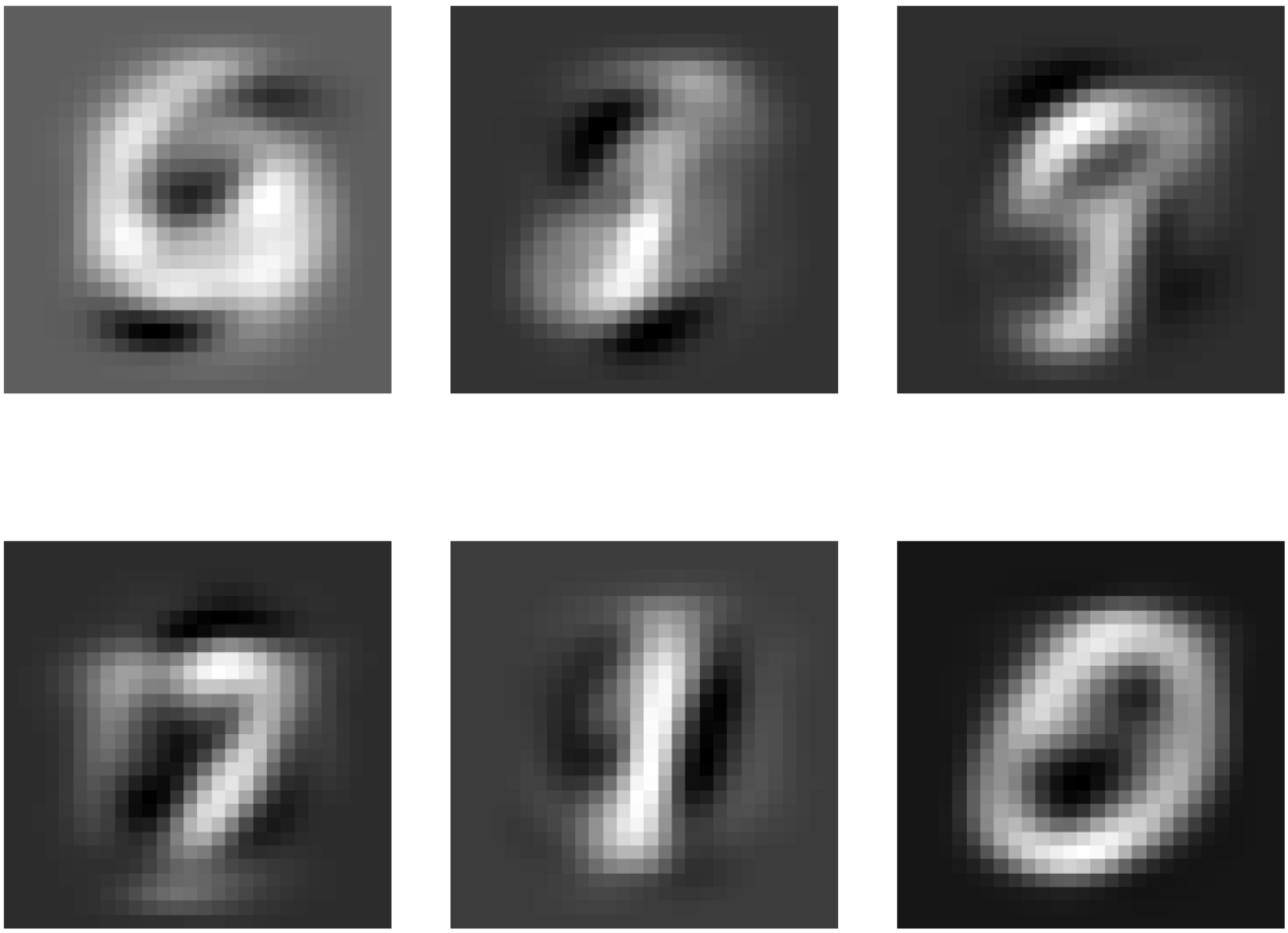
2 Dimensional Latent Representation



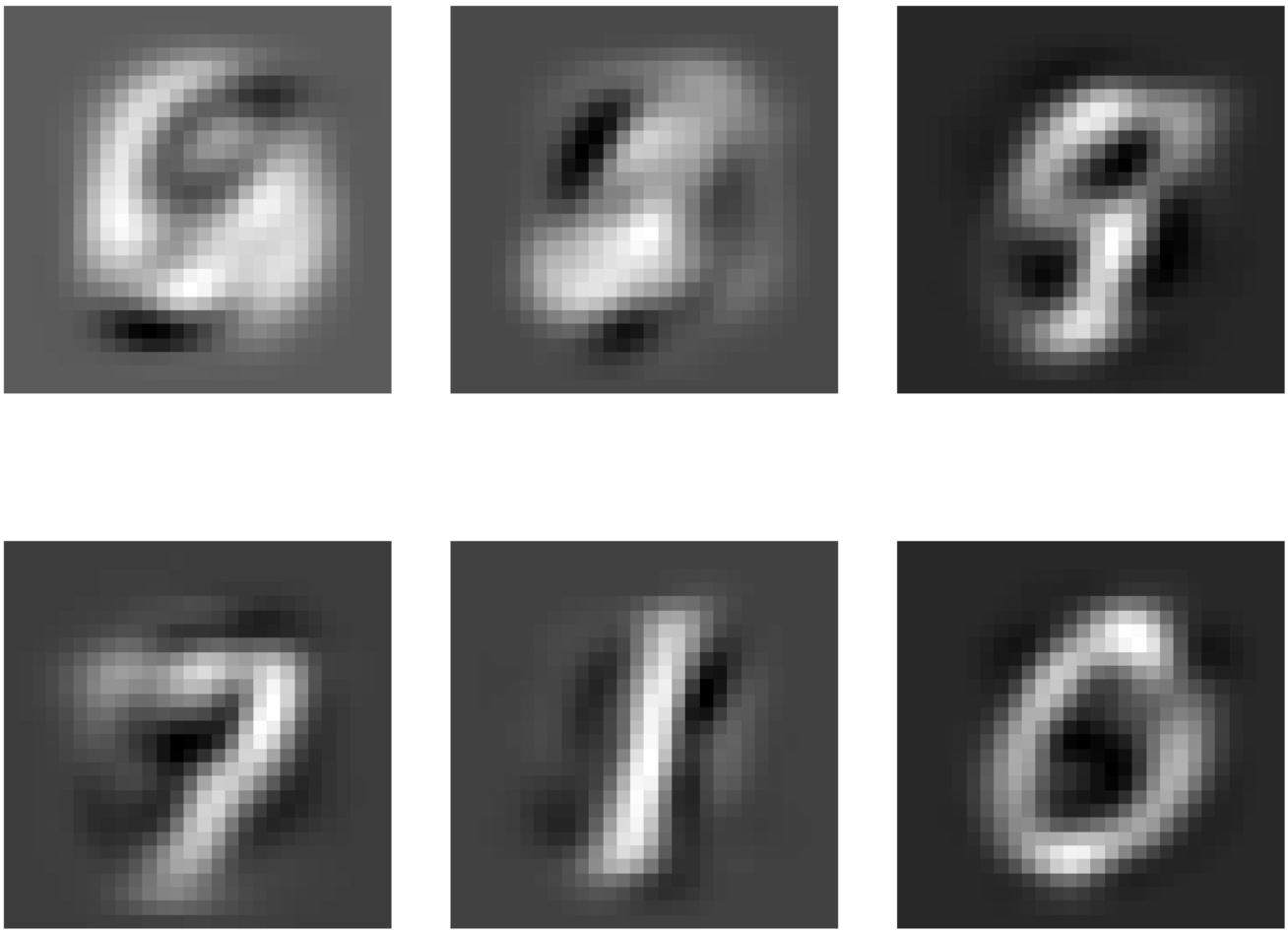
4 Dimensional Latent Representation



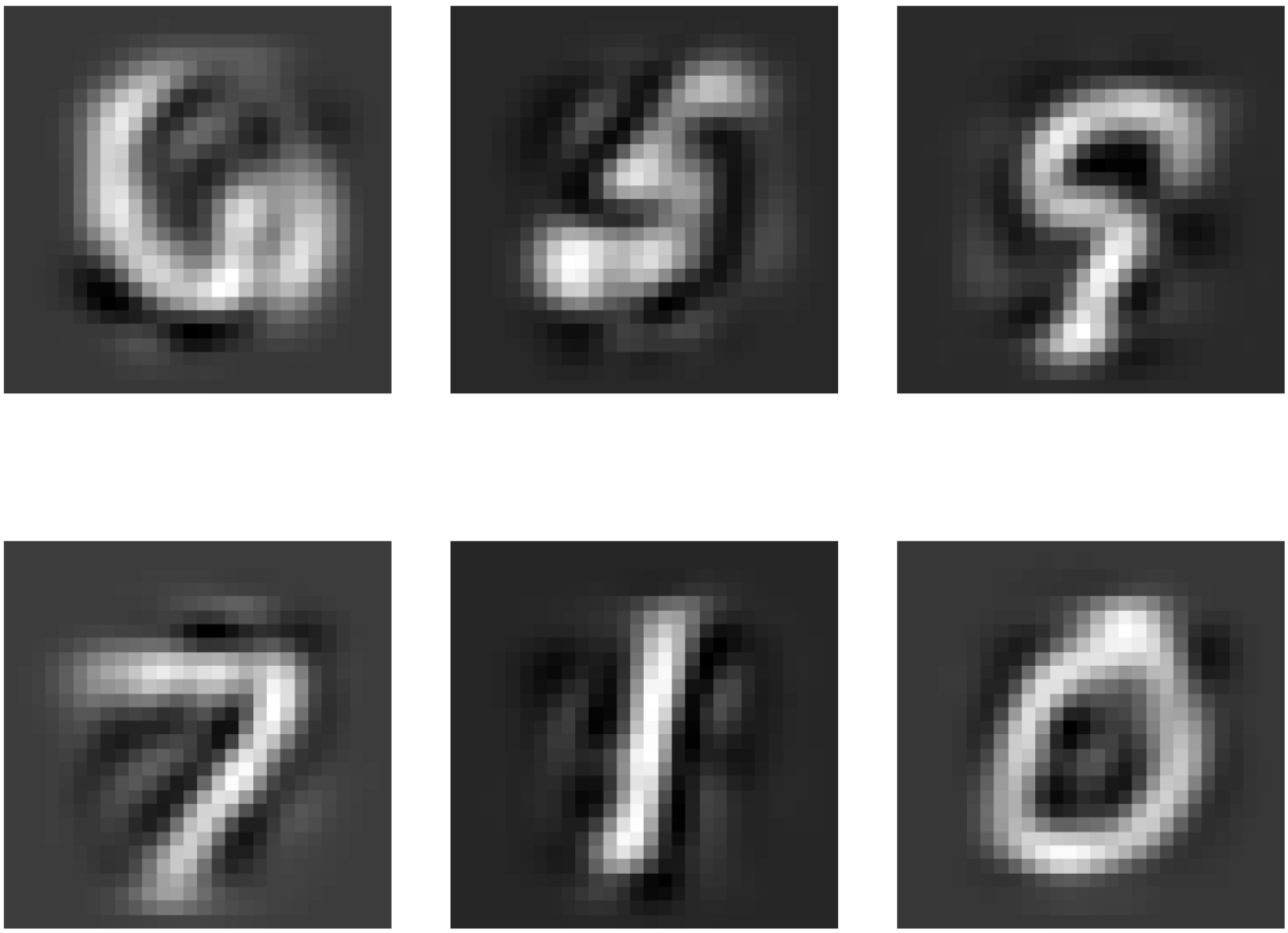
8 Dimensional Latent Representation



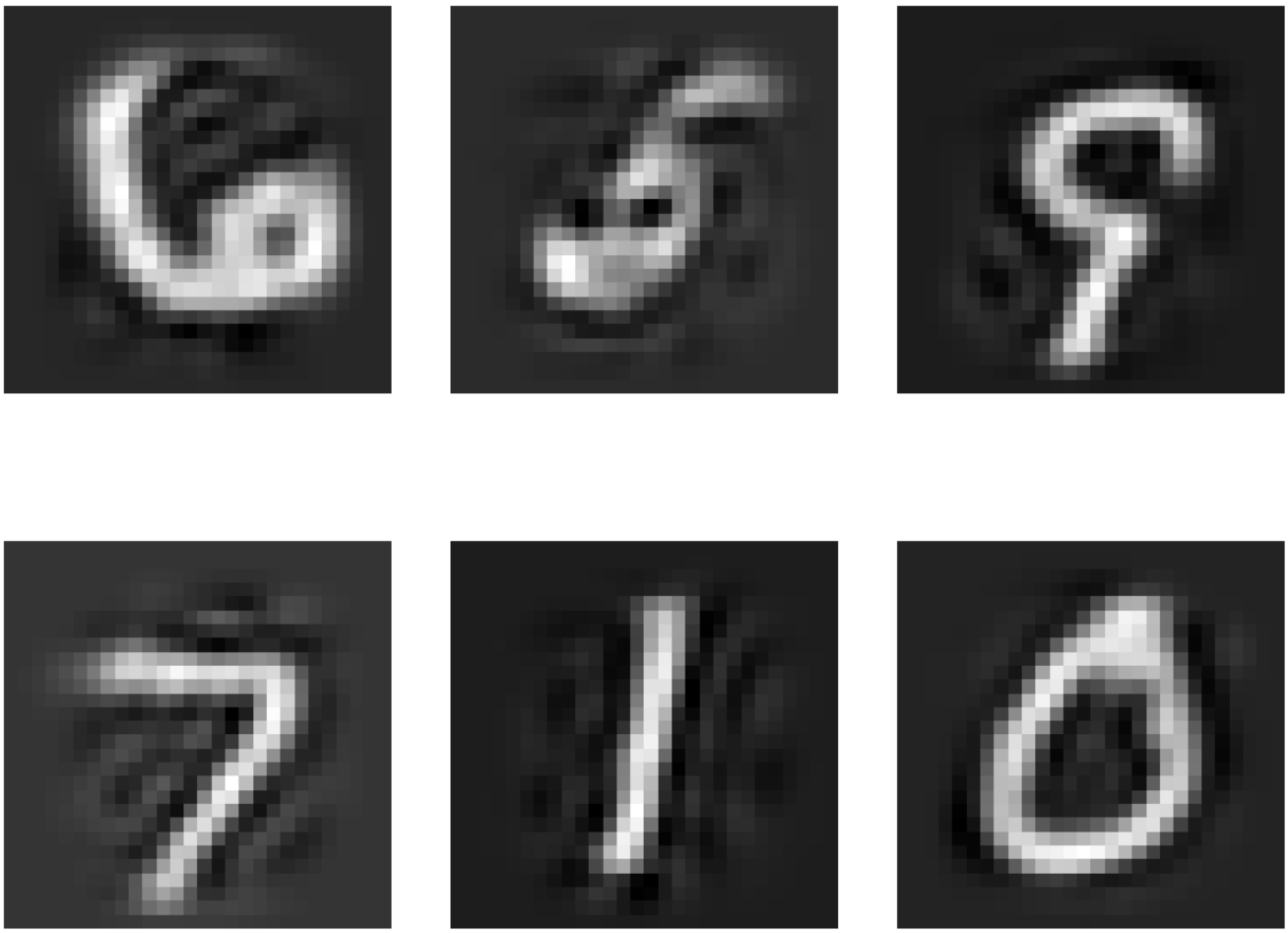
16 Dimensional Latent Representation



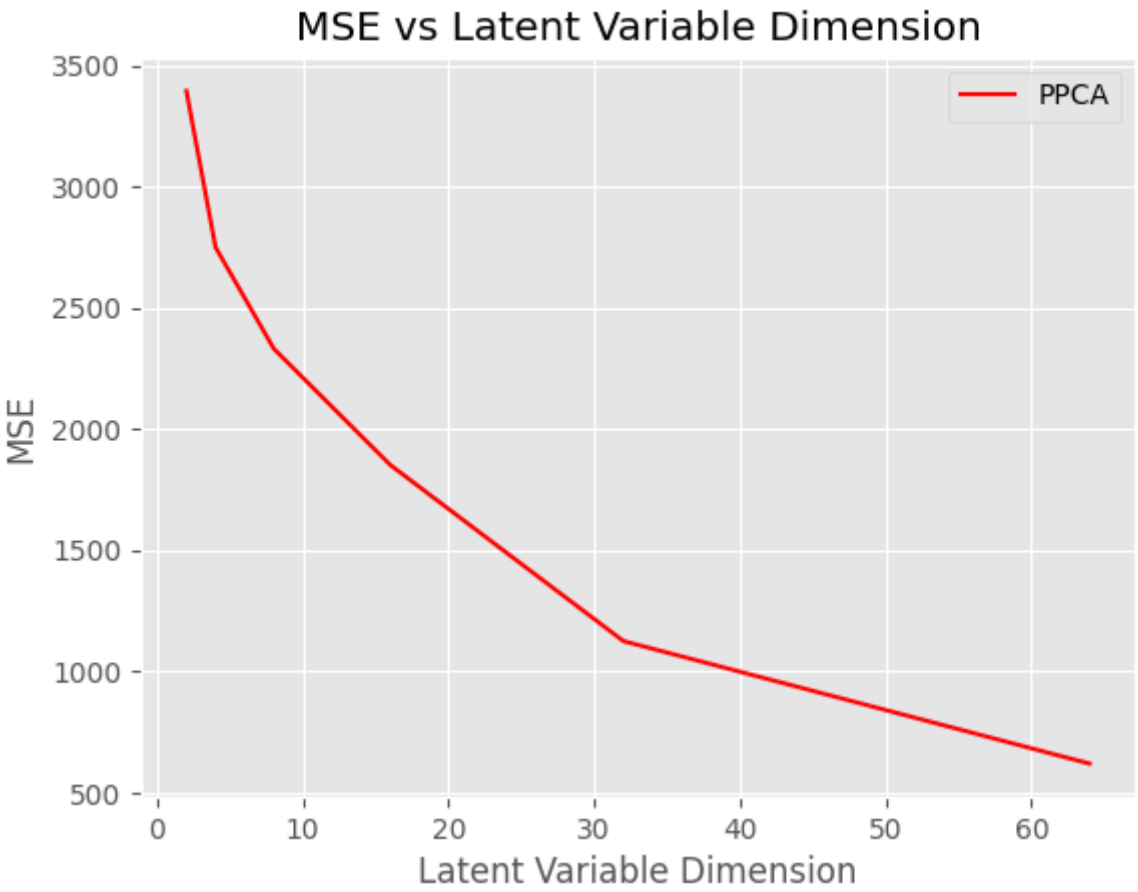
32 Dimensional Latent Representation



64 Dimensional Latent Representation



Reconstruction MSE vs Latent Dimension



3. Variational Auto-Encoders (VAE)

a. Data Loading and Initial Exploration

- Loaded the MNIST dataset from torchvision.datasets using Python.
- Created functions to visualize the handwritten digits.

b. Latent Variable Calculation

- Obtained latent representation of an image using an encoder Neural Network.
- Generated some latent variables from its distribution.

c. Reconstruction

- Passed the latent variable through the decoder Neural Network to obtain an image in the original dimensions.

d. Loss Function

- Minimized the MSE between Reconstructed and Original Image (reconstruction error).
- Minimized the KL Divergence between the Gaussian Distribution obtained on encoding and the Standard Normal Distribution in D dimensions.

e. Encoder Architecture

- Used Multi Layer Perceptrons (MLP) with LeakyReLU activation for projection.

- Projected from the original 784 to hidden_dim=100, and then to latent_dim using two different layers to get Mean and LogVar respectively.

f. Decoder Architecture

- Used Multi Layer Perceptrons (MLP) with LeakyReLU activation for projection.
- Projected from latent_dim to hidden_dim=100, then to original_dim=784 using Sigmoid activation.

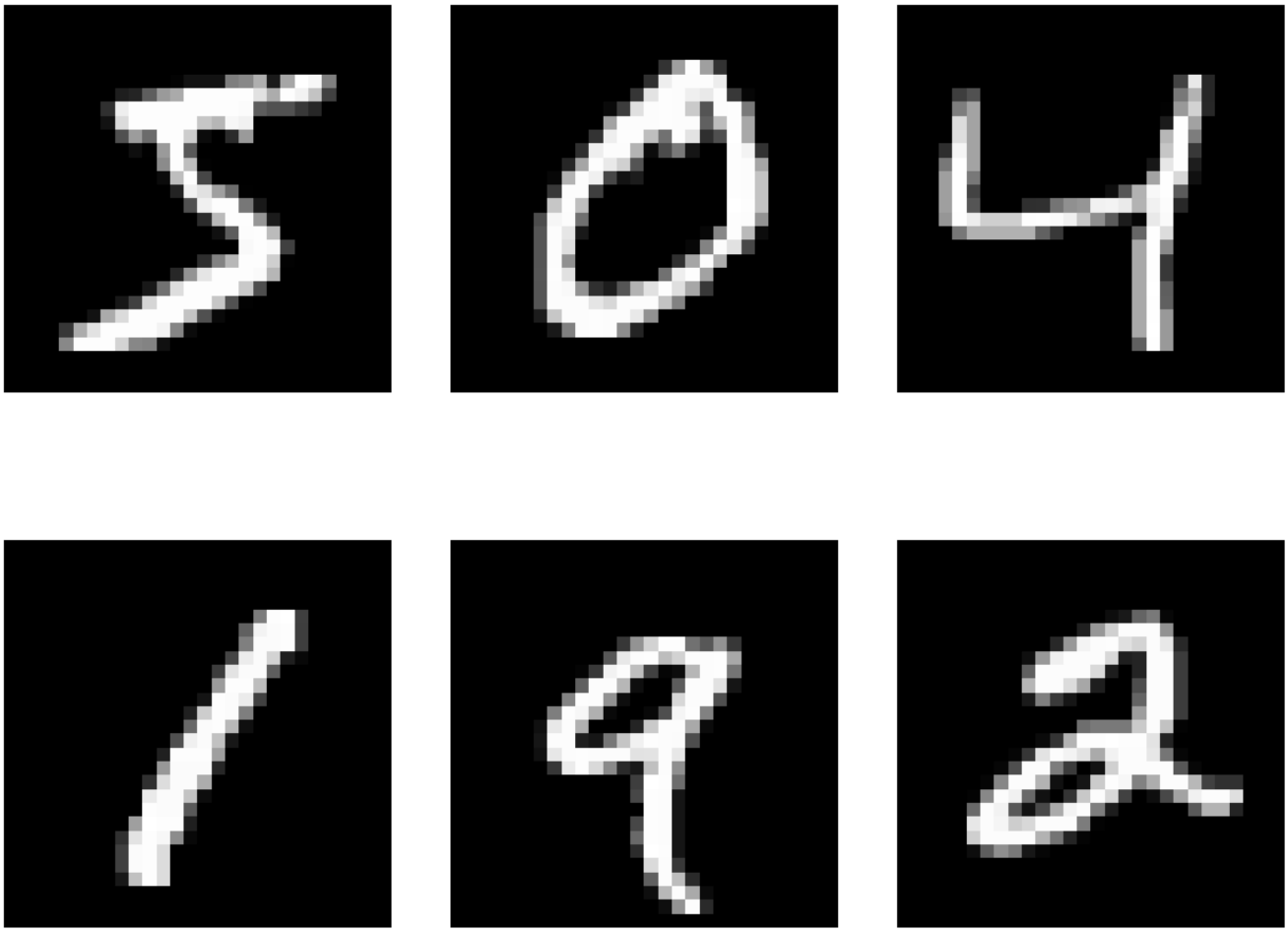
g. Reparametrization Trick

- Sampled from Gaussian Distribution obtained from encoder forward propagation using the reparametrization trick.
- Forward propagated sampled vector through decoder to obtain reconstructed image in original dimensions.

h. Visualization

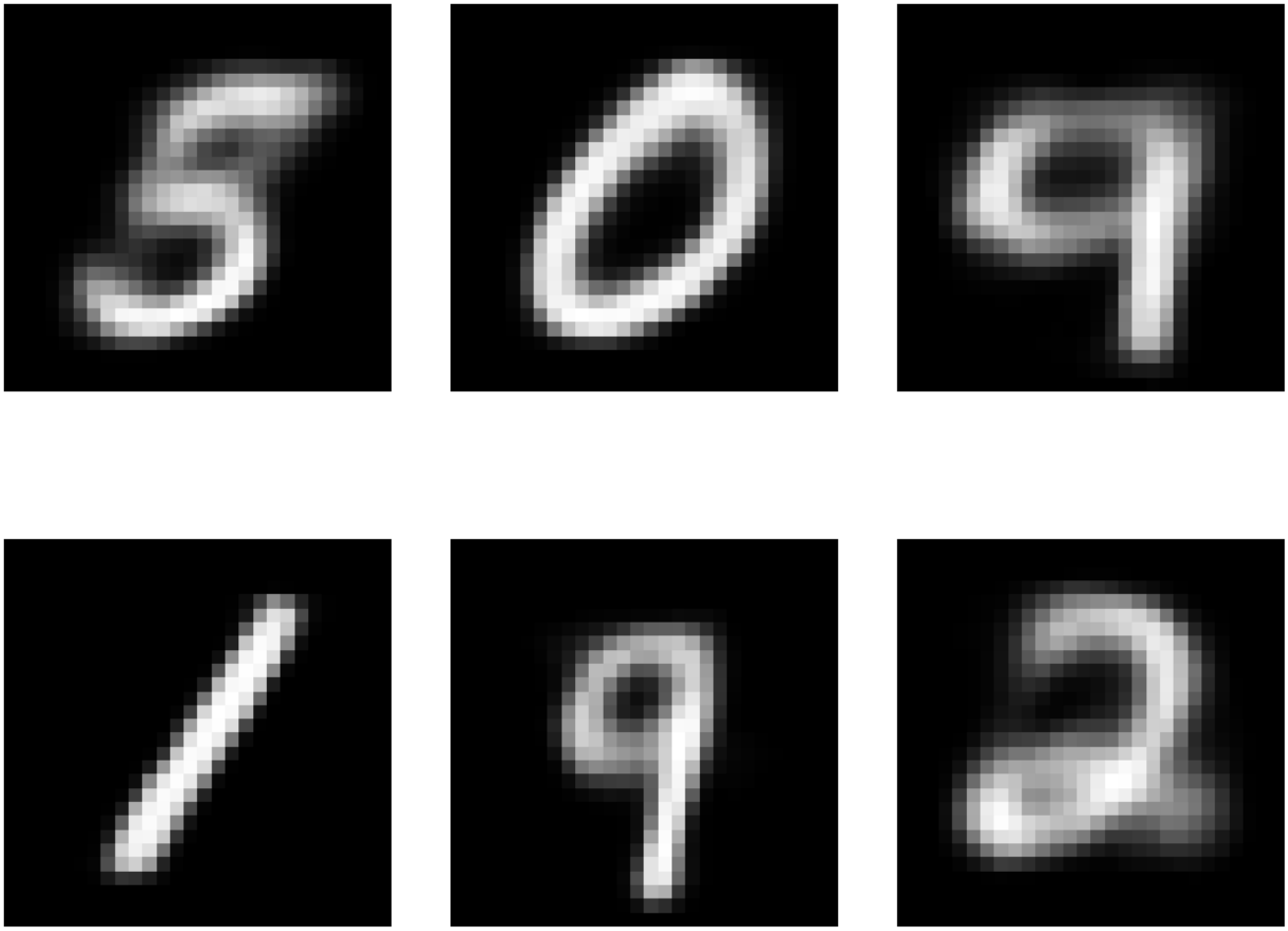
- Sampled 6 images from the MNIST dataset, plotted them, converted them to their latent representation, and reconstructed them.

Original Images

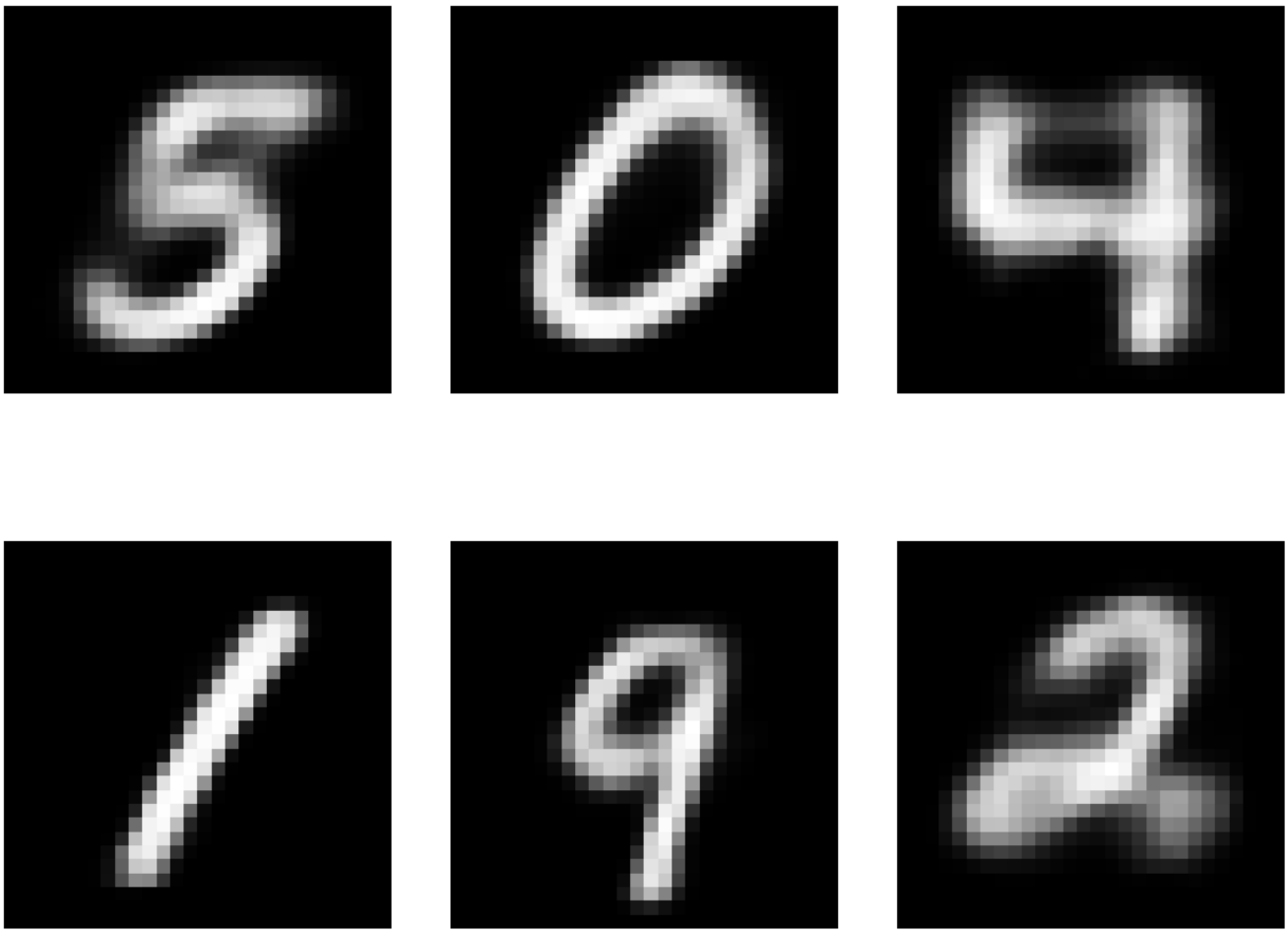


Reconstructed Images

2 Dimensional Latent Representation



4 Dimensional Latent Representation

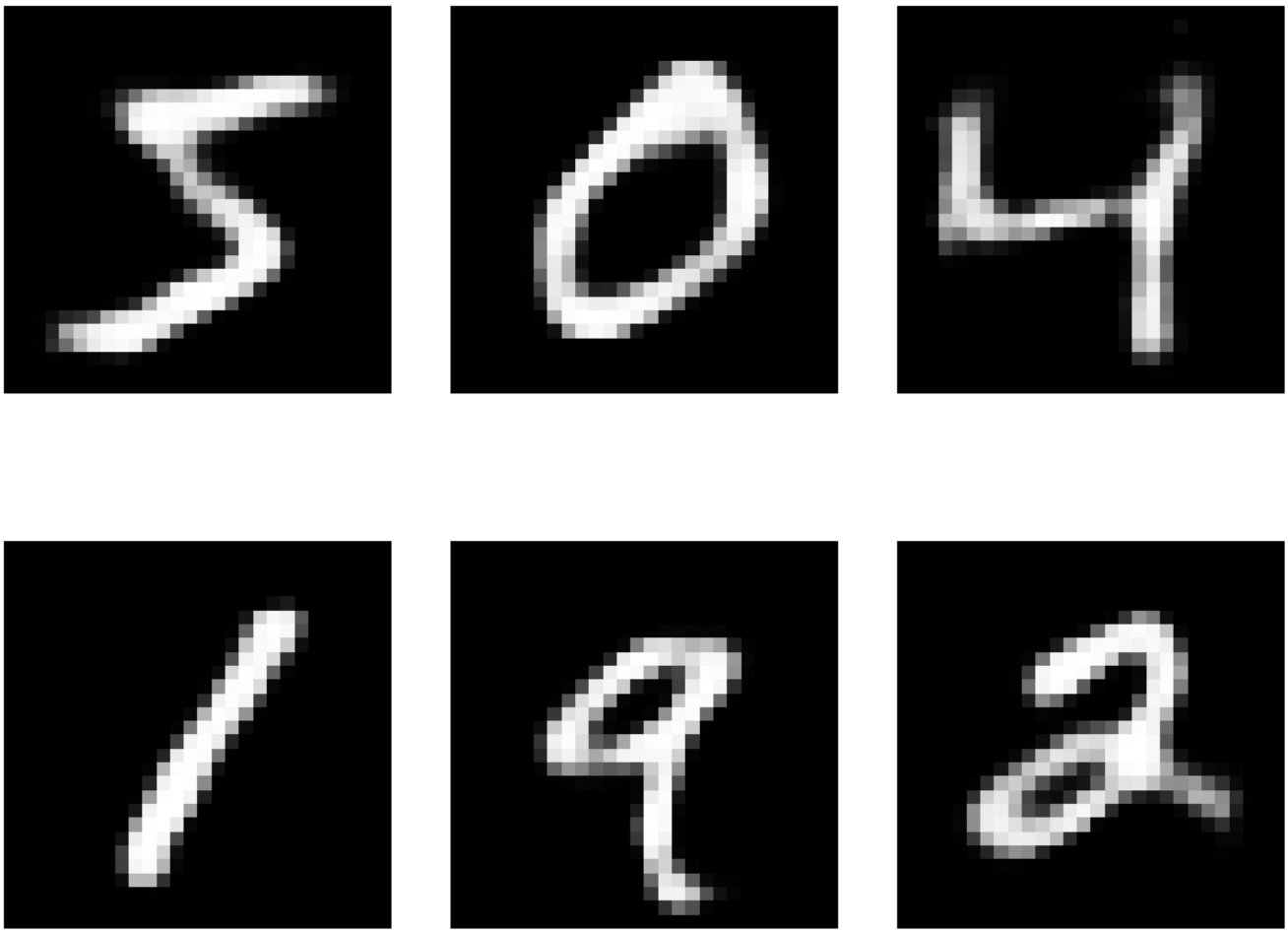


8 Dimensional Latent Representation

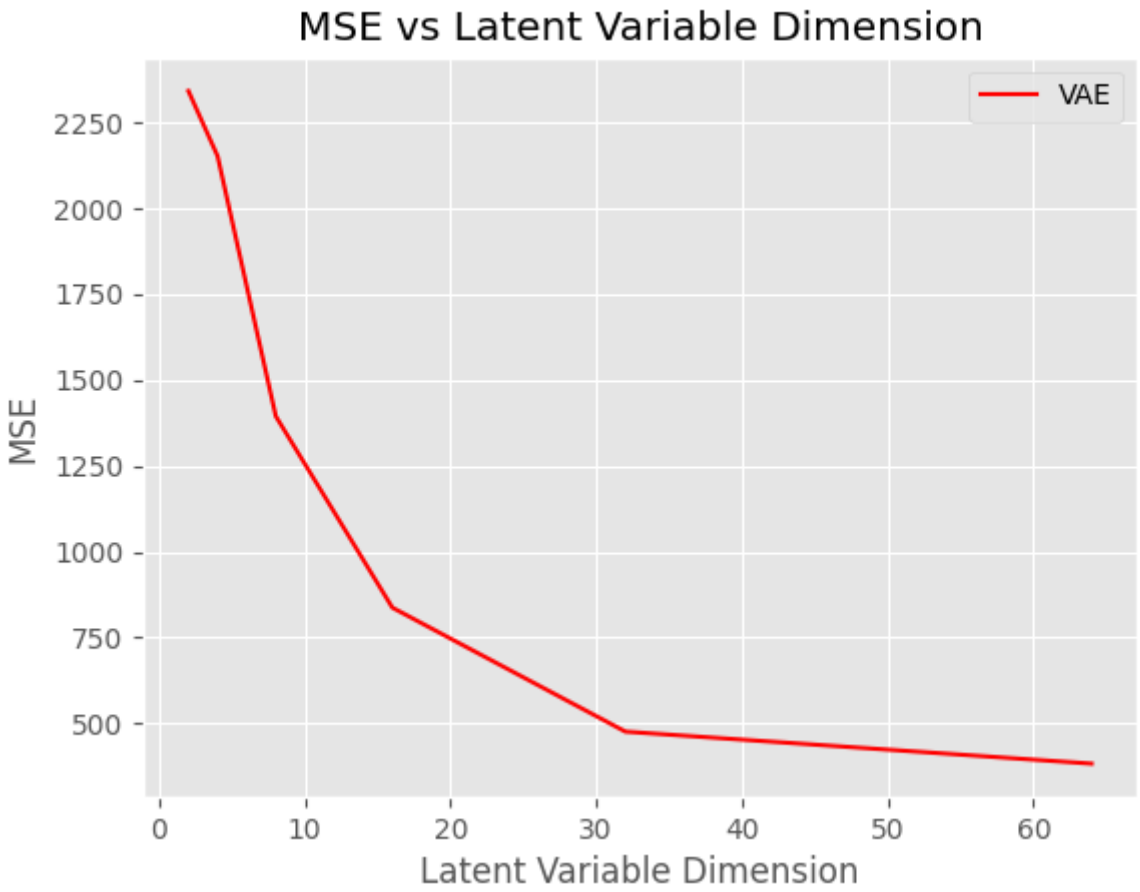
16 Dimensional Latent Representation

32 Dimensional Latent Representation

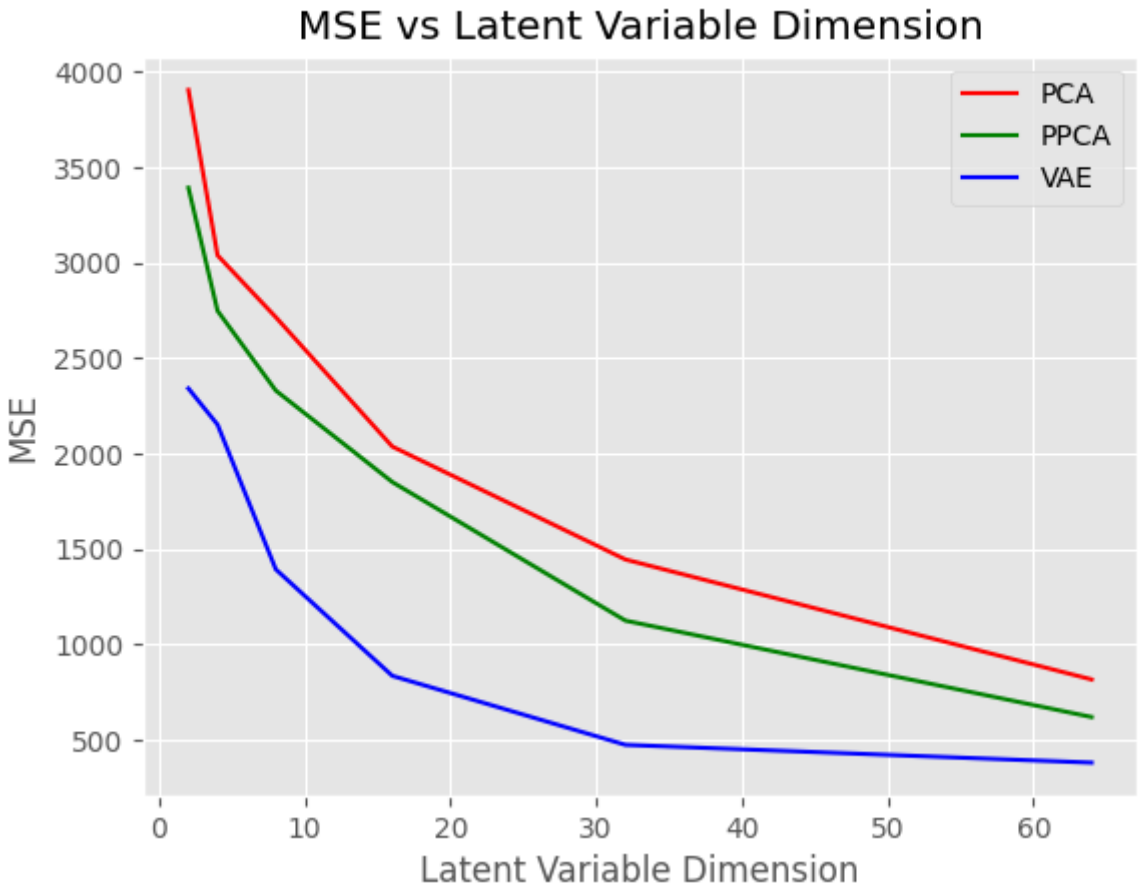
64 Dimensional Latent Representation



Reconstruction MSE vs Latent Dimension



Comparing the Reconstruction Error of the 3 Models



Conclusion

- Probabilistic Models enable understanding the underlying distribution of data for generating more samples and finding the probability of a particular sample occurring in the distribution.
- PCA provides a lower-dimensional representation with reasonable accuracy.
- PPCA offers probabilistic representation, insight into data distribution, and latent variables of an image.
- VAE is a sophisticated approach enabling good reconstruction accuracies and generating convincing images on random sampling from the Standard Normal Distribution in Latent Space because it minimizes the KL Divergence of the Latent Distributions and SND in the loss function.