

Exp 10.

Perform Comprison on MNIST Dataset using
Autoencoder.

P-Plan:

To develop an Autoencoder to compress and reconstruct images from the MNIST dataset

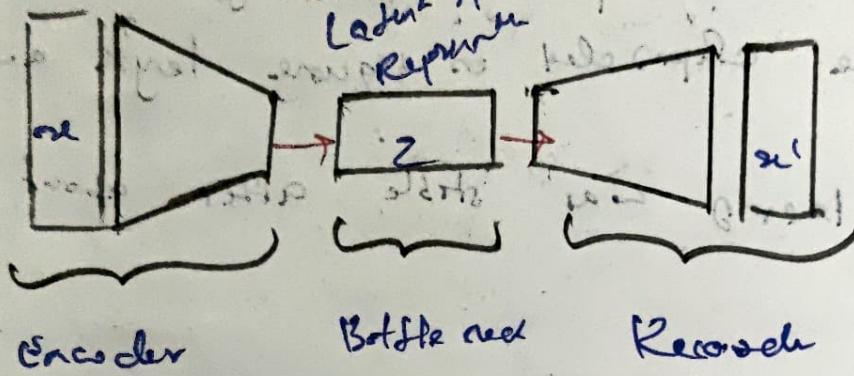
Objectives:

1. To understand the concept of unsupervised learning
2. To perform image compression using a bottleneck architecture
3. To reconstruct input images from compressed representations.
4. To minimize ~~and~~ reconstruction loss.
5. To visualize original and reconstructed images.

Pseudocode

1. Load and normalize the MNIST dataset.
2. Define encoder network with dense layer values dimensionality.
3. Define decoder network for image reconstruction
4. Compile autoencoder using Adam optimizer and mean squared error
5. Train model and display reconstructed output

Autorecoder



Reconstructed image

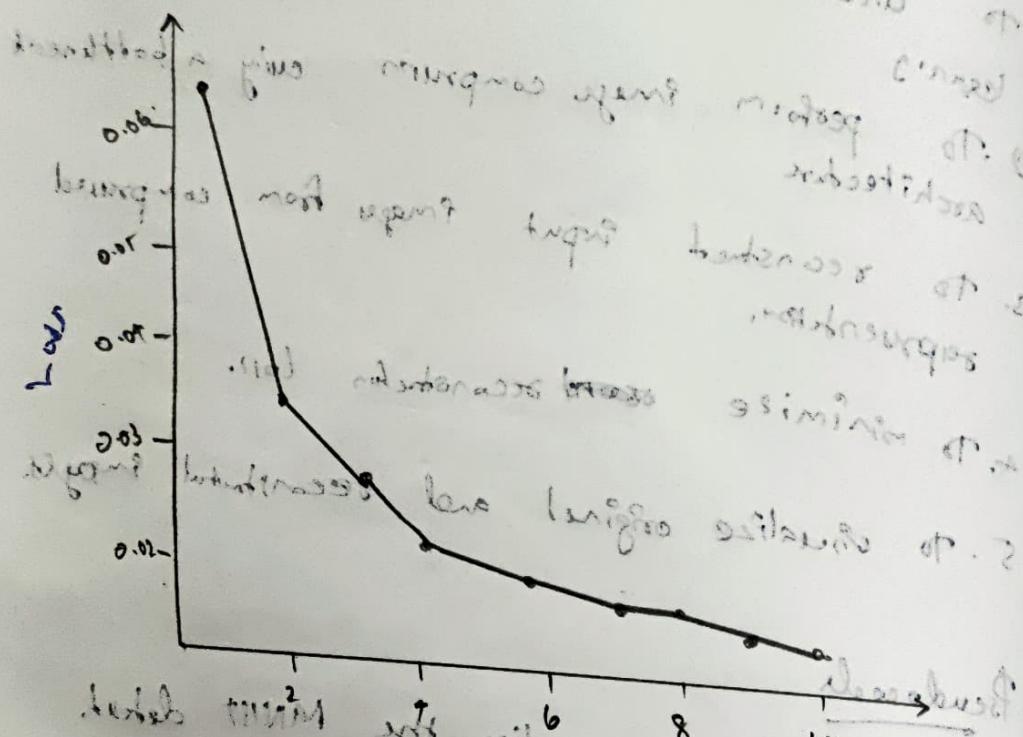
Encoder

Bottleneck

Decoder

Epoch	Training loss	Validation loss
1	0.125	0.14
5	0.075	0.071
10	0.050	0.045
20	0.030	0.024
30	0.025	0.026

With increasing number of iterations, training loss decreases and validation loss remains constant.



The graph shows that the training loss decreases rapidly initially and then levels off. The validation loss remains relatively stable around 0.025 after epoch 5. This indicates that the model is likely overfitting to the training data and may need more regularization or data augmentation.

Observation

1. Loss decreases as training progresses
2. Encoded features retain digit identity
3. Reconstructed digits appear slightly blurred
4. Reconstruction ratio depends on bottleneck size
5. Model generalizes well for unseen digits

Result

Autoencoder successfully compressed and reconstructed MNIST digits, demonstrating efficient feature representation



~~This~~

Exp 11.

Experiment

using Variational Autoencoder (VAE)

Aims:

To implement a Variational Autoencoder to generate new images similar to MNIST digits

Objectives:

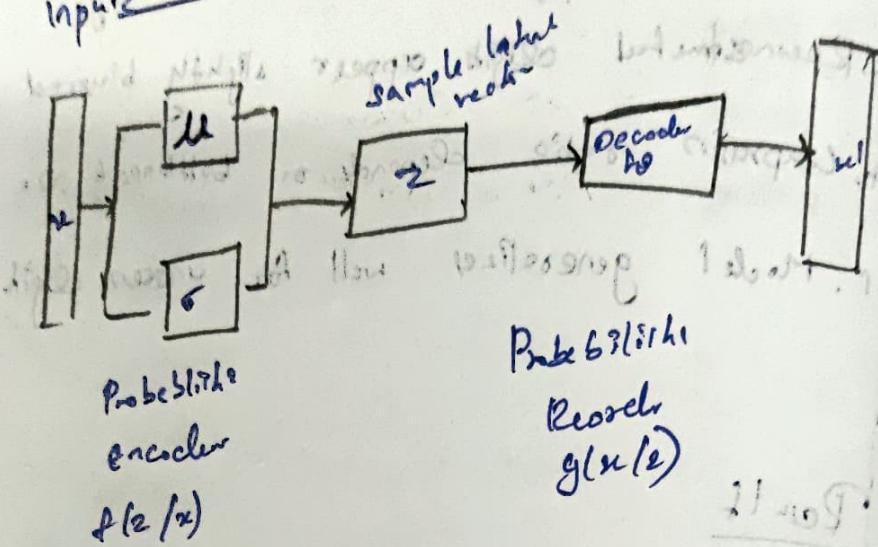
1. To learn probabilistic latent representation
2. To apply the reparameterization trick for sampling
3. To generate synthetic data from learned latent space
4. To balance ~~reconstruction~~ reconstruction and KL divergence loss.
5. To visualize new digit generation.

Procedure

1. Load MNIST and normalize images.
2. Build encoder to output mean and log variance
3. Use reparameterizers to sample latent vectors
4. Decouple latent vectors to reconstruct images
5. Train and generate new samples.

variable Aufende Architektur

Input $x \in \mathbb{R}^n$ Reproduced Input



Probabilistic

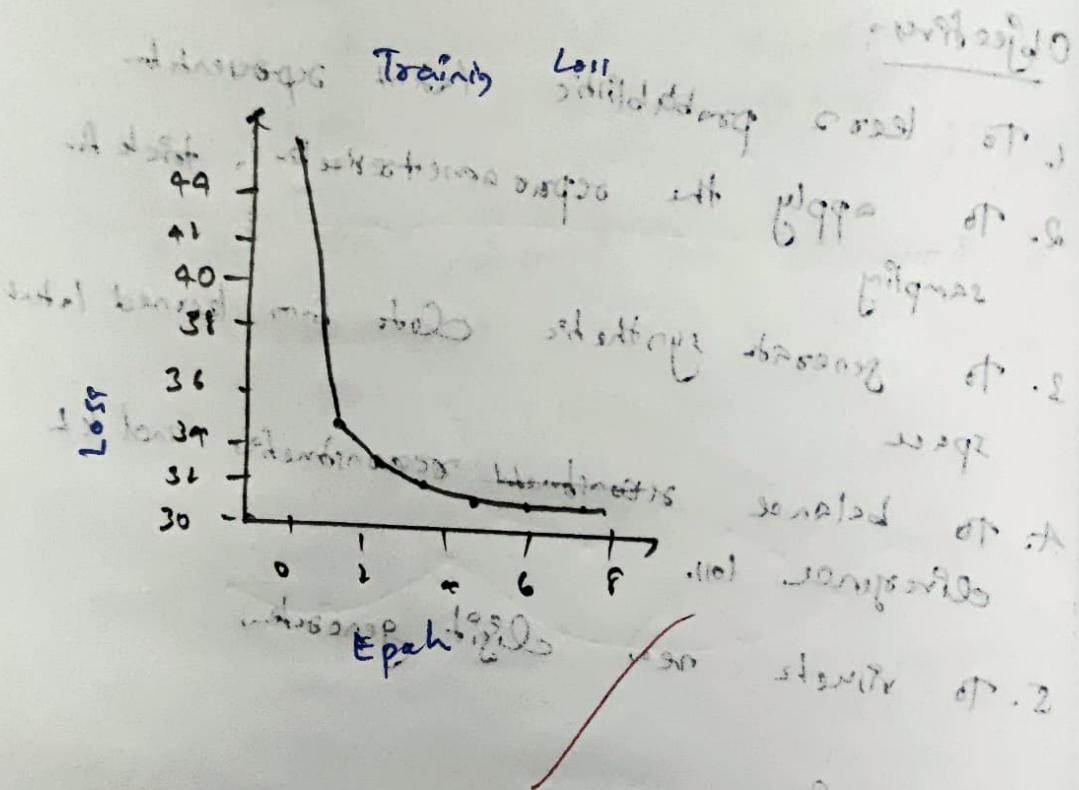
Decoder
 $g(z)$

Latent variable becomes the component for dimensionality reduction
changes with respect to representation or import weight w_{ij}



Epoch	Reconstruction Loss	KL Divergence	DAL Loss
1	120.4	6.0	3.5
5	92.6	2.8	9.1
10	70.2	2.2	9.9

Training loss vs Epoch
Initial sum of weights regard as 100%.



Loss function has three local minima of different types of solutions being.
Local minima of other messages and
Weighted average of hidden total always +
signals was lowest loss weight.

Observation:

1. VAE produces smoother reconstruction than AE
2. Latent space p_{ϕ} continuous and structured.
3. Generated digits are similar to real ones.
4. KL divergence regularizes latent distributions
5. Model performance improves with training epochs

Result:

VAE successfully generated realistic digit images by learning continuous latent representation

