RA2311047010040

# Index

மரம் வளர்ப்போம்
மண் வளம் காப்போம்

Superior®
Note Books

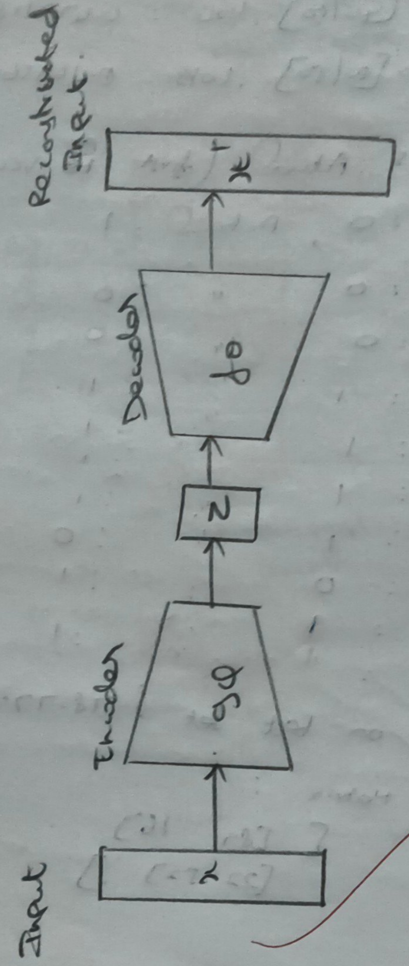
Name : Vishwa S

Subject : DL

Std. : .......... Sec. : .......... Roll No. ..........

School : ..........

| No. | Date | Title | Marks | Signature |
|---|---|---|---|---|
| 1 | 24/7/25 | Exploring the DL platforms. | | |
| 2 | 31/7/25 | Implement a classifier using open source dataset | | |
| 3 | 7/8/25 | Study the classifier with respect to statistical parameters. | | 18/8 |
| 4 | 14/8/25 | Build a simple feed forward neural network to Recognize Handwritten characters | | 22/8 |
| 5 | 22/8/25 | Study of Activation functions & Their role. | | 9/9/25 |
| 6 | 9/9/25 | Implement Gradient Descent & Backpropagation in Deep neural Network. | | |
| 7 | 16/9/25 | Build a CNN model to classify cat & dog image | | 23/9/25 |
| 8 Ex 2 | 9 30/9/25 | Build a RNN Network. | | 30/9 |
| 9 Ex 3 | 9/10/25 | Experiment using LSTM | | |
| 10 | 9/10/25 | AutoEncoder | | |
| 11 | 9/10/25 | VARIATIONAL AUTOENCODER | | |
| 12 | 17/10/25 | Deep convolutional GAN | | |




Architecture of Autoencoder

9/10/25

# Lab 10: Perform compression on MNIST Dataset using Autoencoder

**Aim:**
To implement an Autoencoder for compressing & reconstructing MNIST images, demonstrating dimensionality reduction & unsupervised learning.

**Objectives**

1) To understand the architecture & working of Autoencoders.
2) To perform image compression using the encoder network.
3) To reconstruct compressed images using the decoder network.
4) To visualize the quality of compression & reconstruction.

**Observation**

* Autoencoders learn compressed latent representations of input data without supervision.
* The encoder reduces images dimensions, while the decoder reconstructs it back to near-original form.
* As training progresses, reconstruction error decreases significantly.
* Compressed features occupy denser dimensions, demonstrating the model's ability to capture essential information.

## Pseudocode:

BEGIN

Step 1 : Import libraries (TensorFlow/Keras, Numpy, Matplotlib)

Step 2: Load MNIST dataset and normalize images (values between 0 and 1)

Step 3: Flatten images for input to the autoencoder.

Step 4: Define encoder model
- Input layer (784 neurons)
- Dense hidden layers with decreasing dimensions

Step 5: Define decoder model
- Dense hidden layers with increasing dimensions.
- Output layer (784 neurons, sigmoid activation)

Step 6: compile the autoencoder with optimizer

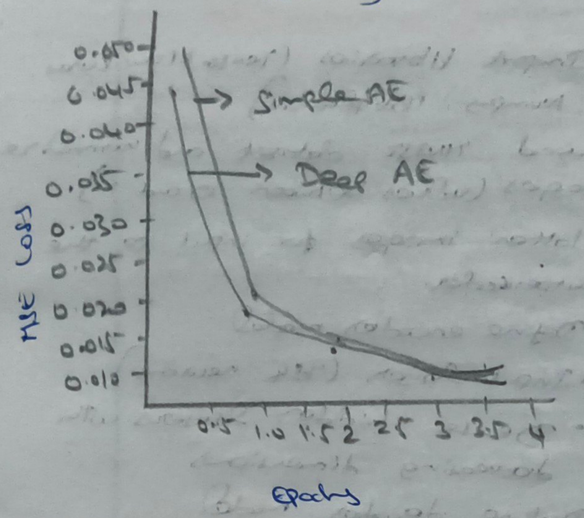
Step 7: Train the model using training data for N epochs.

Step 8: Encode and decode test images

Step 9: visualize original vs reconstructed images

END

Training Loss Comparison.




Simple AE Reconstruction (Top: original, Bottom: Reconstructed)

8 6 3 5 8

8 6 3 5 8

Deep AE Reconstruction (Top: original, Bottom: Reconstructed)

1 4 1 7 9

1 4 1 7 9

Output:

Simple AE

Epoch 1 Loss : 0.0501
Epoch 2 Loss : 0.0211
Epoch 3 Loss : 0.0154
Epoch 4 Loss : 0.0128
Epoch 5 Loss : 0.0114

Deep AE

Epoch 1, Loss : 0.0445
Epoch 2, Loss : 0.0199
Epoch 3, Loss : 0.0150
Epoch 4, Loss : 0.0124
Epoch 5, Loss : 0.0120.

Result:

Successfully Implement the Autoencoder for MNIST Dataset.

dltlab10.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

Connect T4

```
pip install torch torchvision matplotlib
```

```
Requirement already satisfied: torch in /usr/local/lib/python3.12/dist-packages (2.8.0+cu126)
Requirement already satisfied: torchvision in /usr/local/lib/python3.12/dist-packages (0.23.0+cu126)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.12/dist-packages (3.10.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from torch) (3.20.0)
Requirement already satisfied: typing-extensions>=4.10.0 in /usr/local/lib/python3.12/dist-packages (from torch) (4.15.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from torch) (75.2.0)
Requirement already satisfied: sympy>=1.13.3 in /usr/local/lib/python3.12/dist-packages (from torch) (1.13.3)
Requirement already satisfied: networkx in /usr/local/lib/python3.12/dist-packages (from torch) (3.5)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from torch) (3.1.6)
Requirement already satisfied: fsspec in /usr/local/lib/python3.12/dist-packages (from torch) (2025.3.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.6.80 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.80)
Requirement already satisfied: nvidia-cudnn-cu12==9.10.2.21 in /usr/local/lib/python3.12/dist-packages (from torch) (9.10.2.21)
Requirement already satisfied: nvidia-cublas-cu12==12.6.4.1 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.4.1)
Requirement already satisfied: nvidia-cufft-cu12==11.3.0.4 in /usr/local/lib/python3.12/dist-packages (from torch) (11.3.0.4)
Requirement already satisfied: nvidia-curand-cu12==10.3.7.77 in /usr/local/lib/python3.12/dist-packages (from torch) (10.3.7.77)
Requirement already satisfied: nvidia-cusolver-cu12==11.7.1.2 in /usr/local/lib/python3.12/dist-packages (from torch) (11.7.1.2)
Requirement already satisfied: nvidia-cusparse-cu12==12.5.4.2 in /usr/local/lib/python3.12/dist-packages (from torch) (12.5.4.2)
Requirement already satisfied: nvidia-cusparselt-cu12==0.7.1 in /usr/local/lib/python3.12/dist-packages (from torch) (0.7.1)
Requirement already satisfied: nvidia-nccl-cu12==2.27.3 in /usr/local/lib/python3.12/dist-packages (from torch) (2.27.3)
Requirement already satisfied: nvidia-nvtx-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-nvjitlink-cu12==12.6.85 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.85)
Requirement already satisfied: nvidia-cufile-cu12==1.11.1.6 in /usr/local/lib/python3.12/dist-packages (from torch) (1.11.1.6)
Requirement already satisfied: triton==3.4.0 in /usr/local/lib/python3.12/dist-packages (from torch) (3.4.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.12/dist-packages (from torchvision) (2.0.2)
Requirement already satisfied: pillow!=8.3.*,>=5.3.0 in /usr/local/lib/python3.12/dist-packages (from torchvision) (11.3.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (1.3.3)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (4.60.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (1.4.9)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (25.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (3.2.5)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil>=2.7->matplotlib) (1.17.0)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.12/dist-packages (from sympy>=1.13.3->torch) (1.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from jinja2->torch) (3.0.3)
```

```
import torch
import torch.nn as nn
import torch.optim as optim
from torchvision import datasets, transforms
from torch.utils.data import DataLoader
```

dltlab10.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

Share V

Connect T4

```
criterion = nn.MSELoss()
opt_simple = optim.Adam(simpleAE.parameters(), lr=1e-3)
opt_deep = optim.Adam(deepAE.parameters(), lr=1e-3)
def train_model(model, optimizer, name):
    losses = []
    for epoch in range(5):
        total_loss = 0
        for imgs, _ in train_loader:
            imgs = imgs.to(device)
            out = model(imgs)
            loss = criterion(out, imgs)
            optimizer.zero_grad()
            loss.backward()
            optimizer.step()
            total_loss += loss.item()
        avg = total_loss / len(train_loader)
        losses.append(avg)
        print(f"{name} | Epoch [{epoch+1}/5], Loss: {avg:.4f}")
    return losses
loss_simple = train_model(simpleAE, opt_simple, "SimpleAE")
loss_deep = train_model(deepAE, opt_deep, "DeepAE")
plt.plot(loss_simple, label='Simple AE')
plt.plot(loss_deep, label='Deep AE')
plt.title("Training Loss Comparison")
plt.xlabel("Epochs")
plt.ylabel("MSE Loss")
plt.legend()
plt.show()
def show_reconstruction(model, name):
    model.eval()
    imgs, _ = next(iter(train_loader))
    imgs = imgs[:5].to(device)
    with torch.no_grad():
        recons = model(imgs)
    imgs, recons = imgs.cpu(), recons.cpu()
    fig, axes = plt.subplots(2, 5, figsize=(10, 4))
    for i in range(5):
        axes[0, i].imshow(imgs[i].squeeze(), cmap='gray')
        axes[0, i].axis('off')
        axes[1, i].imshow(recons[i].squeeze(), cmap='gray')
        axes[1, i].axis('off')
    plt.suptitle(f"{name} Reconstruction (Top: Original, Bottom: Reconstructed)")
    plt.show()

show_reconstruction(simpleAE, "Simple Autoencoder")
show_reconstruction(deepAE, "Deep Autoencoder")
```

dltlab10.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all Connect T4

```
class SimpleAE(nn.Module):
    def __init__(self):
        super(SimpleAE, self).__init__()
        self.encoder = nn.Sequential(
            nn.Linear(28*28, 128),
            nn.ReLU(),
            nn.Linear(128, 32)
        )
        self.decoder = nn.Sequential(
            nn.Linear(32, 128),
            nn.ReLU(),
            nn.Linear(128, 28*28),
            nn.Sigmoid()
        )
    def forward(self, x):
        x = x.view(-1, 28*28)
        z = self.encoder(x)
        out = self.decoder(z)
        return out.view(-1, 1, 28, 28)
class DeepAE(nn.Module):
    def __init__(self):
        super(DeepAE, self).__init__()
        self.encoder = nn.Sequential(
            nn.Linear(784, 512),
            nn.ReLU(),
            nn.Linear(512, 128),
            nn.ReLU(),
            nn.Linear(128, 32)
        )
        self.decoder = nn.Sequential(
            nn.Linear(32, 128),
            nn.ReLU(),
            nn.Linear(128, 512),
            nn.ReLU(),
            nn.Linear(512, 784),
            nn.Sigmoid()
        )
    def forward(self, x):
        x = x.view(-1, 784)
        z = self.encoder(x)
        out = self.decoder(z)
        return out.view(-1, 1, 28, 28)
simpleAE = SimpleAE().to(device)
deepAE = DeepAE().to(device)
criterion = nn.MSELoss()
```

dltlab10.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all Connect

```
    fig, axes = plt.subplots(2, 5, figsize=(10, 4))
    for i in range(5):
        axes[0, i].imshow(imgs[i].squeeze(), cmap='gray')
        axes[0, i].axis('off')
        axes[1, i].imshow(recons[i].squeeze(), cmap='gray')
        axes[1, i].axis('off')
    plt.suptitle(f"{name} Reconstruction (Top: Original, Bottom: Reconstructed)")
    plt.show()

show_reconstruction(simpleAE, "Simple Autoencoder")
show_reconstruction(deepAE, "Deep Autoencoder")
```

```
100%|██████████| 28.9k/28.9k [00:00<00:00, 483kB/s]
100%|██████████| 1.65M/1.65M [00:00<00:00, 4.51MB/s]
100%|██████████| 4.54k/4.54k [00:00<00:00, 3.40MB/s]
SimpleAE | Epoch [1/5], Loss: 0.0501
SimpleAE | Epoch [2/5], Loss: 0.0211
SimpleAE | Epoch [3/5], Loss: 0.0154
SimpleAE | Epoch [4/5], Loss: 0.0128
SimpleAE | Epoch [5/5], Loss: 0.0114
DeepAE | Epoch [1/5], Loss: 0.0445
DeepAE | Epoch [2/5], Loss: 0.0199
DeepAE | Epoch [3/5], Loss: 0.0150
DeepAE | Epoch [4/5], Loss: 0.0124
DeepAE | Epoch [5/5], Loss: 0.0108
```

dltlab10.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

Share

Connect


Simple Autoencoder Reconstruction (Top: Original, Bottom: Reconstructed)

Deep Autoencoder Reconstruction (Top: Original, Bottom: Reconstructed)