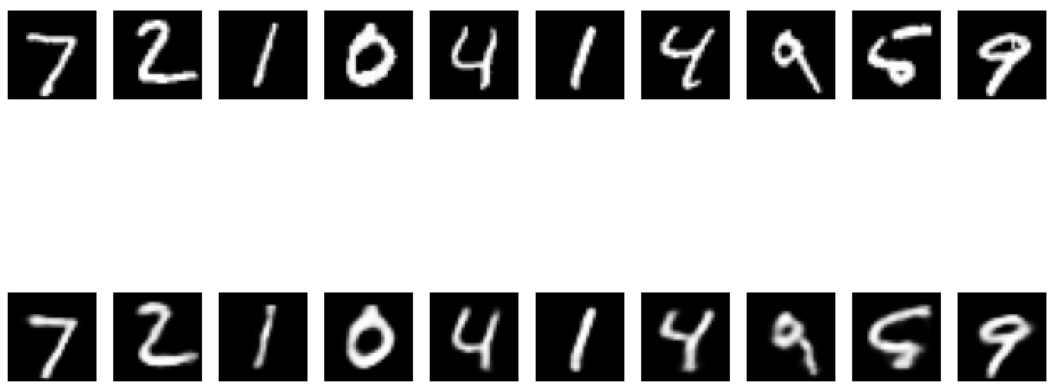
```
# ================= IMPORTS =================
import torch
import torch.nn as nn
import torch.optim as optim
from torchvision import datasets, transforms
from torch.utils.data import DataLoader
import matplotlib.pyplot as plt
import numpy as np

# ================= DEVICE =================
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

# ================= DATASET =================
batch_size = 128

transform = transforms.Compose([
    transforms.ToTensor()
])

train_dataset = datasets.MNIST(
    root="./data", train=True, transform=transform, download=True
)
test_dataset = datasets.MNIST(
    root="./data", train=False, transform=transform, download=True
)

train_loader = DataLoader(train_dataset, batch_size=batch_size, shuffle=True)
test_loader = DataLoader(test_dataset, batch_size=batch_size, shuffle=False)

# ================= VAE MODEL =================
class VAE(nn.Module):
    def __init__(self, latent_dim=20):
        super(VAE, self).__init__()

        # -------- Encoder --------
        self.fc1 = nn.Linear(28*28, 400)
        self.fc_mu = nn.Linear(400, latent_dim)
        self.fc_logvar = nn.Linear(400, latent_dim)

        # -------- Decoder --------
        self.fc2 = nn.Linear(latent_dim, 400)
        self.fc3 = nn.Linear(400, 28*28)

    def encode(self, x):
        h = torch.relu(self.fc1(x))
        mu = self.fc_mu(h)
        logvar = self.fc_logvar(h)
        return mu, logvar

    def reparameterize(self, mu, logvar):
        std = torch.exp(0.5 * logvar)
        eps = torch.randn_like(std)
        return mu + eps * std

    def decode(self, z):
        h = torch.relu(self.fc2(z))
        return torch.sigmoid(self.fc3(h))

    def forward(self, x):
        mu, logvar = self.encode(x)
        z = self.reparameterize(mu, logvar)
        recon_x = self.decode(z)
        return recon_x, mu, logvar

# ================= LOSS FUNCTION =================
def vae_loss(recon_x, x, mu, logvar):
    recon_loss = nn.functional.binary_cross_entropy(
        recon_x, x, reduction='sum'
    )
    kl_loss = -0.5 * torch.sum(
        1 + logvar - mu.pow(2) - logvar.exp()
    )
    return recon_loss + kl_loss, recon_loss, kl_loss

# ================= INITIALIZE =================
latent_dim = 20
model = VAE(latent_dim).to(device)
optimizer = optim.Adam(model.parameters(), lr=1e-3)

epochs = 30
train_losses = []
val_losses = []

# ================= TRAINING =================
for epoch in range(1, epochs + 1):
    model.train()
    train_loss = 0

    for x, _ in train_loader:
        x = x.view(-1, 28*28).to(device)

        recon_x, mu, logvar = model(x)
        loss, _, _ = vae_loss(recon_x, x, mu, logvar)

        optimizer.zero_grad()
        loss.backward()
        optimizer.step()

        train_loss += loss.item()

    train_loss /= len(train_loader.dataset)
    train_losses.append(train_loss)

    # -------- Validation --------
    model.eval()
    val_loss = 0
    with torch.no_grad():
        for x, _ in test_loader:
            x = x.view(-1, 28*28).to(device)
            recon_x, mu, logvar = model(x)
            loss, _, _ = vae_loss(recon_x, x, mu, logvar)
            val_loss += loss.item()

    val_loss /= len(test_loader.dataset)
    val_losses.append(val_loss)

    print(f"Epoch {epoch}/{epochs} | Train Loss: {train_loss:.2f} | Val Loss: {val_loss:.2f}")

# ================= RECONSTRUCT IMAGES =================
model.eval()
with torch.no_grad():
    x, _ = next(iter(test_loader))
    x = x.view(-1, 28*28).to(device)
    recon_x, _, _ = model(x)

x = x.view(-1, 1, 28, 28).cpu()
recon_x = recon_x.view(-1, 1, 28, 28).cpu()

plt.figure(figsize=(8,4))
for i in range(10):
    plt.subplot(2,10,i+1)
    plt.imshow(x[i][0], cmap='gray')
    plt.axis('off')
    plt.subplot(2,10,i+11)
    plt.imshow(recon_x[i][0], cmap='gray')
    plt.axis('off')
plt.suptitle("Top: Original | Bottom: Reconstructed")
plt.show()

# ================= GENERATE NEW SAMPLES =================
with torch.no_grad():
    z = torch.randn(16, latent_dim).to(device)
    samples = model.decode(z).view(-1, 1, 28, 28).cpu()

plt.figure(figsize=(4,4))
for i in range(16):
    plt.subplot(4,4,i+1)
    plt.imshow(samples[i][0], cmap='gray')
    plt.axis('off')
plt.suptitle("Generated Samples from Latent Space")
plt.show()

# ================= LOSS CURVES =================
plt.figure(figsize=(6,4))
plt.plot(train_losses, label="Train Loss")
plt.plot(val_losses, label="Validation Loss")
plt.xlabel("Epoch")
plt.ylabel("Loss")
plt.title("VAE Training & Validation Loss")
plt.legend()
plt.grid(True)
```

```
plt.show()
```

```
100%|██████████| 9.91M/9.91M [00:01<00:00, 4.98MB/s]
100%|██████████| 28.9k/28.9k [00:00<00:00, 129kB/s]
100%|██████████| 1.65M/1.65M [00:01<00:00, 1.23MB/s]
100%|██████████| 4.54k/4.54k [00:00<00:00, 8.43MB/s]
Epoch 1/30 | Train Loss: 165.23 | Val Loss: 127.79
Epoch 2/30 | Train Loss: 121.68 | Val Loss: 115.79
Epoch 3/30 | Train Loss: 114.70 | Val Loss: 111.95
Epoch 4/30 | Train Loss: 111.82 | Val Loss: 109.87
Epoch 5/30 | Train Loss: 110.10 | Val Loss: 108.62
Epoch 6/30 | Train Loss: 108.94 | Val Loss: 107.84
Epoch 7/30 | Train Loss: 108.10 | Val Loss: 107.02
Epoch 8/30 | Train Loss: 107.43 | Val Loss: 106.58
Epoch 9/30 | Train Loss: 107.00 | Val Loss: 106.52
Epoch 10/30 | Train Loss: 106.53 | Val Loss: 105.83
Epoch 11/30 | Train Loss: 106.20 | Val Loss: 105.70
Epoch 12/30 | Train Loss: 105.85 | Val Loss: 105.34
Epoch 13/30 | Train Loss: 105.61 | Val Loss: 105.17
Epoch 14/30 | Train Loss: 105.34 | Val Loss: 104.93
Epoch 15/30 | Train Loss: 105.17 | Val Loss: 104.98
Epoch 16/30 | Train Loss: 104.95 | Val Loss: 104.64
Epoch 17/30 | Train Loss: 104.76 | Val Loss: 104.49
Epoch 18/30 | Train Loss: 104.62 | Val Loss: 104.49
Epoch 19/30 | Train Loss: 104.41 | Val Loss: 104.21
Epoch 20/30 | Train Loss: 104.29 | Val Loss: 104.10
Epoch 21/30 | Train Loss: 104.21 | Val Loss: 104.10
Epoch 22/30 | Train Loss: 104.07 | Val Loss: 103.78
Epoch 23/30 | Train Loss: 103.93 | Val Loss: 103.78
Epoch 24/30 | Train Loss: 103.78 | Val Loss: 103.90
Epoch 25/30 | Train Loss: 103.72 | Val Loss: 103.47
Epoch 26/30 | Train Loss: 103.60 | Val Loss: 103.63
Epoch 27/30 | Train Loss: 103.52 | Val Loss: 103.36
Epoch 28/30 | Train Loss: 103.41 | Val Loss: 103.38
Epoch 29/30 | Train Loss: 103.31 | Val Loss: 103.33
Epoch 30/30 | Train Loss: 103.21 | Val Loss: 103.21
```

Top: Original | Bottom: Reconstructed



Generated Samples from Latent Space