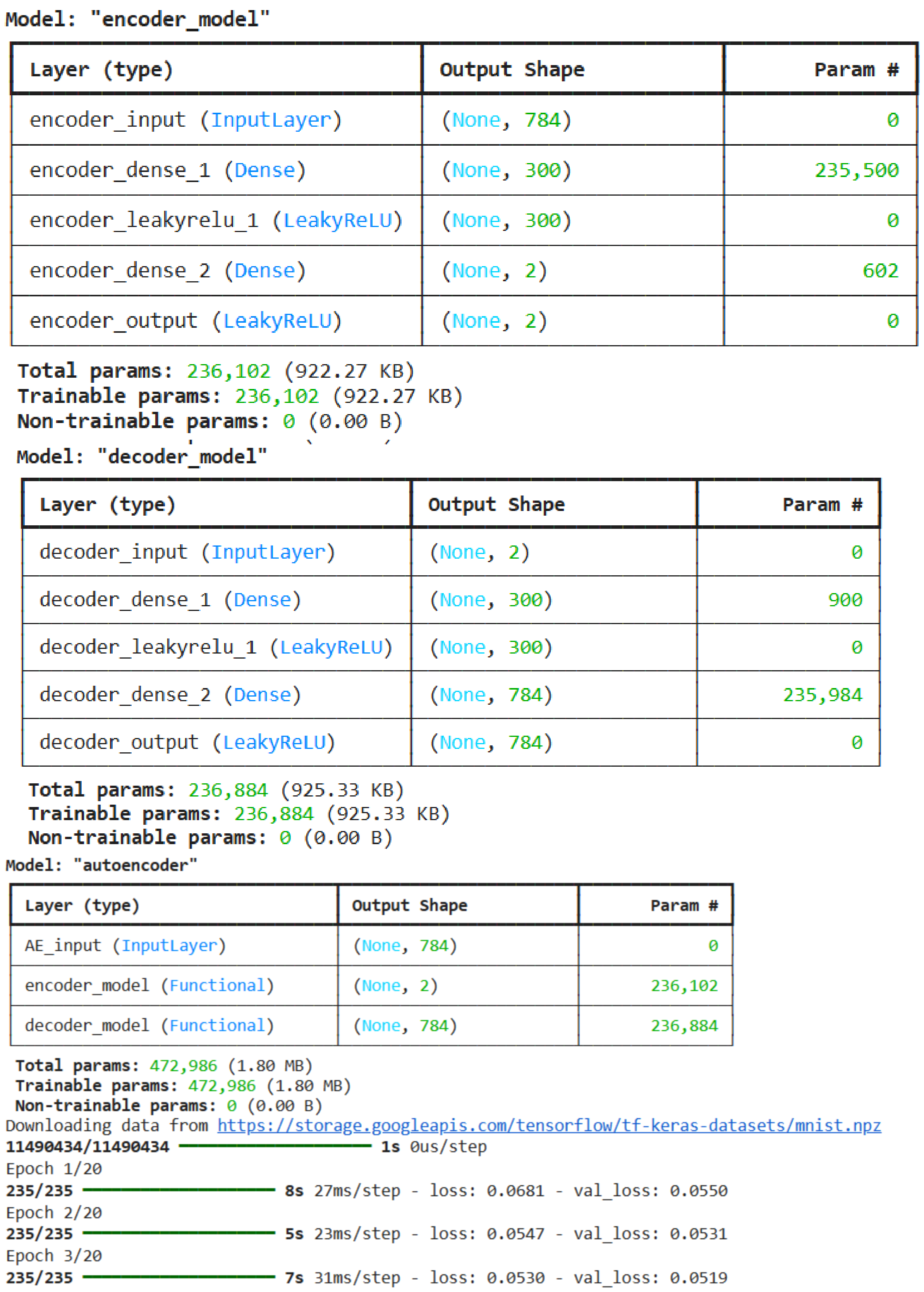
Program 4:

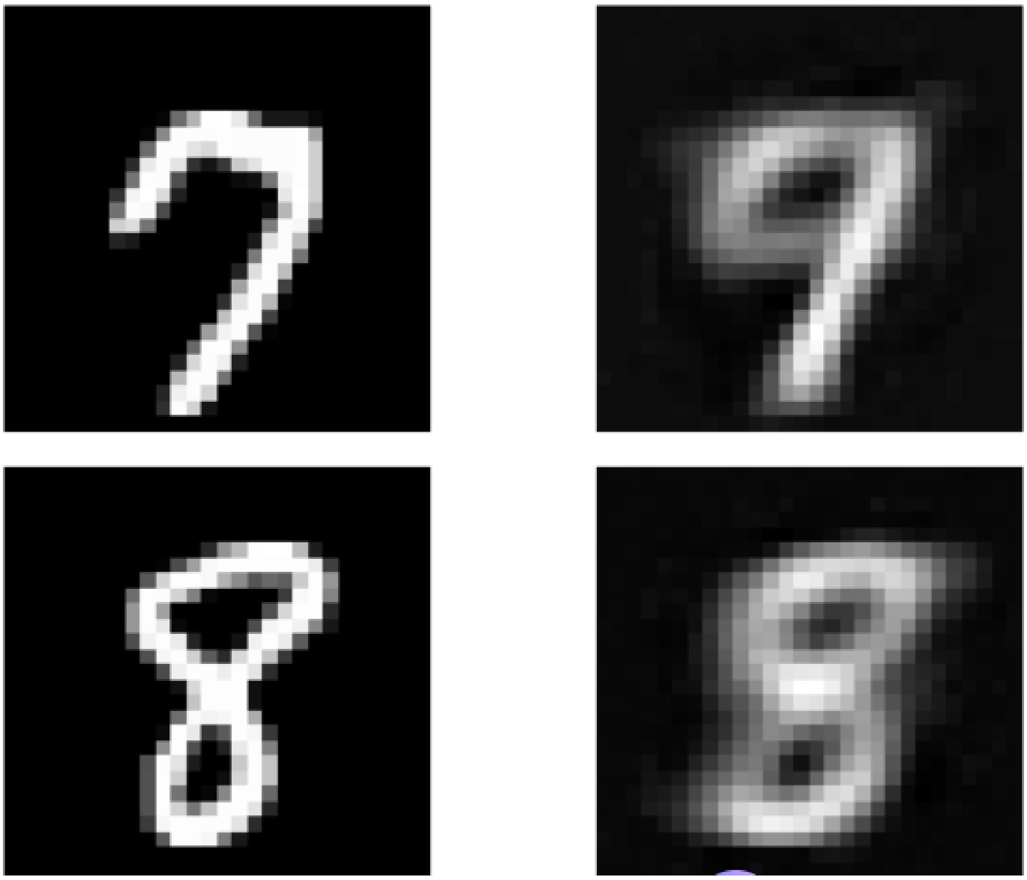
Build and demonstrate an autoencoder network using neural layers for data compression on image dataset.

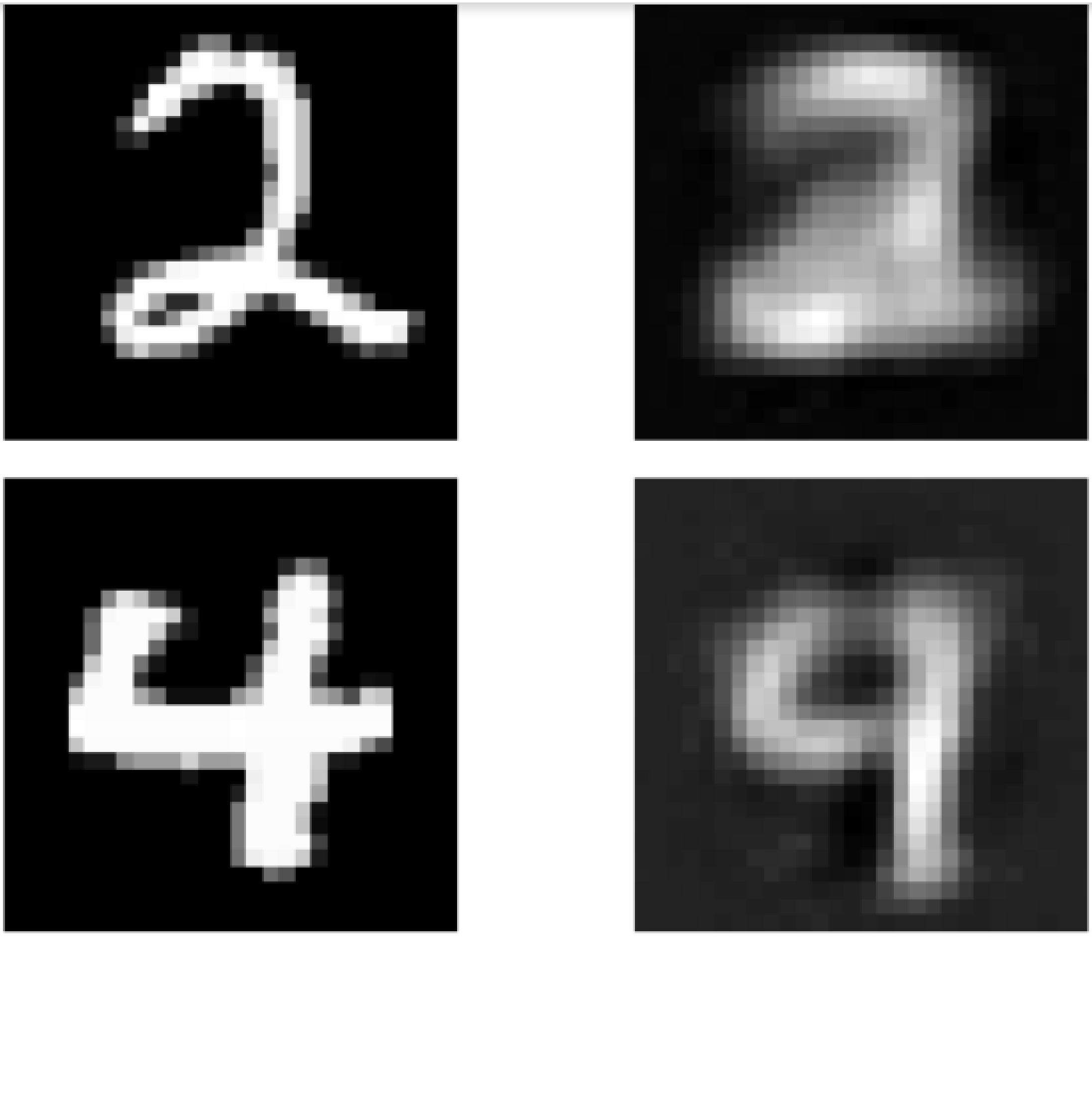
```
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras import layers, models, optimizers, datasets

# Encoder
encoder_input = layers.Input(shape=(784,), name="encoder_input")
x = layers.Dense(300, name="encoder_dense_1")(encoder_input)
x = layers.LeakyReLU(name="encoder_leakyrelu_1")(x)

x = layers.Dense(2, name="encoder_dense_2")(x)
encoder_output = layers.LeakyReLU(name="encoder_output")(x)
encoder = models.Model(encoder_input, encoder_output, name="encoder_model")
encoder.summary()

# 2. Decoder
decoder_input = layers.Input(shape=(2,), name="decoder_input")
x = layers.Dense(300, name="decoder_dense_1")(decoder_input)
x = layers.LeakyReLU(name="decoder_leakyrelu_1")(x)
x = layers.Dense(784, name="decoder_dense_2")(x)
decoder_output = layers.LeakyReLU(name="decoder_output")(x)

decoder = models.Model(decoder_input, decoder_output, name="decoder_model")
decoder.summary()

# Autoencoder
```

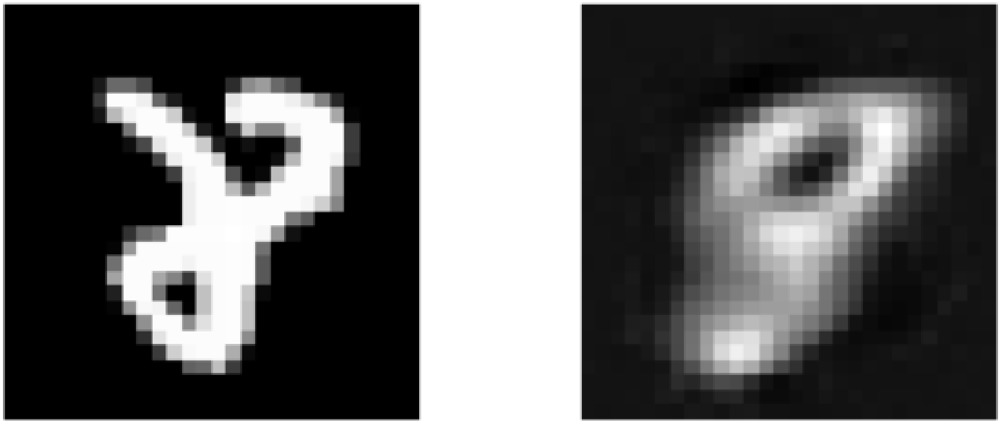
```
ae_input = layers.Input(shape=(784,), name="AE_input")
ae_output = decoder(encoder(ae_input))
autoencoder = models.Model(ae_input, ae_output, name="autoencoder")
autoencoder.summary()

#  Compile
autoencoder.compile(loss="mse",
                    optimizer=optimizers.Adam(learning_rate=0.0005))

#  Load & Preprocess MNIST
(x_train, y_train), (x_test, y_test) = datasets.mnist.load_data()
x_train = x_train.astype("float32") / 255.0
x_test  = x_test.astype("float32") / 255.0

# flatten 28x28 → 784
x_train = x_train.reshape((x_train.shape[0], 784))
x_test  = x_test.reshape((x_test.shape[0], 784))

# Train
autoencoder.fit(x_train, x_train,
                epochs=20,
                batch_size=256,
                shuffle=True,
                validation_data=(x_test, x_test))

#  Encode & Decode
encoded_images = encoder.predict(x_test)
decoded_images = decoder.predict(encoded_images)
decoded_images = decoded_images.reshape((-1, 28, 28))


# Display some results
num_show = 5
plt.figure(figsize=(6, num_show*2))
for i in range(num_show):
    idx = np.random.randint(0, x_test.shape[0])


    # original
    plt.subplot(num_show, 2, 2*i + 1)
    plt.imshow(x_test[idx].reshape(28, 28), cmap="gray")
    plt.axis("off")


    # reconstruction
    plt.subplot(num_show, 2, 2*i + 2)
    plt.imshow(decoded_images[idx], cmap="gray")
    plt.axis("off")


plt.tight_layout()
plt.show()


# Latent 2-D scatter colored by digit label
plt.figure(figsize=(6, 6))
plt.scatter(encoded_images[:, 0], encoded_images[:, 1], c=y_test, cmap="tab10", s=2)
plt.colorbar()
plt.title("2-D Latent Space")
plt.show()
```

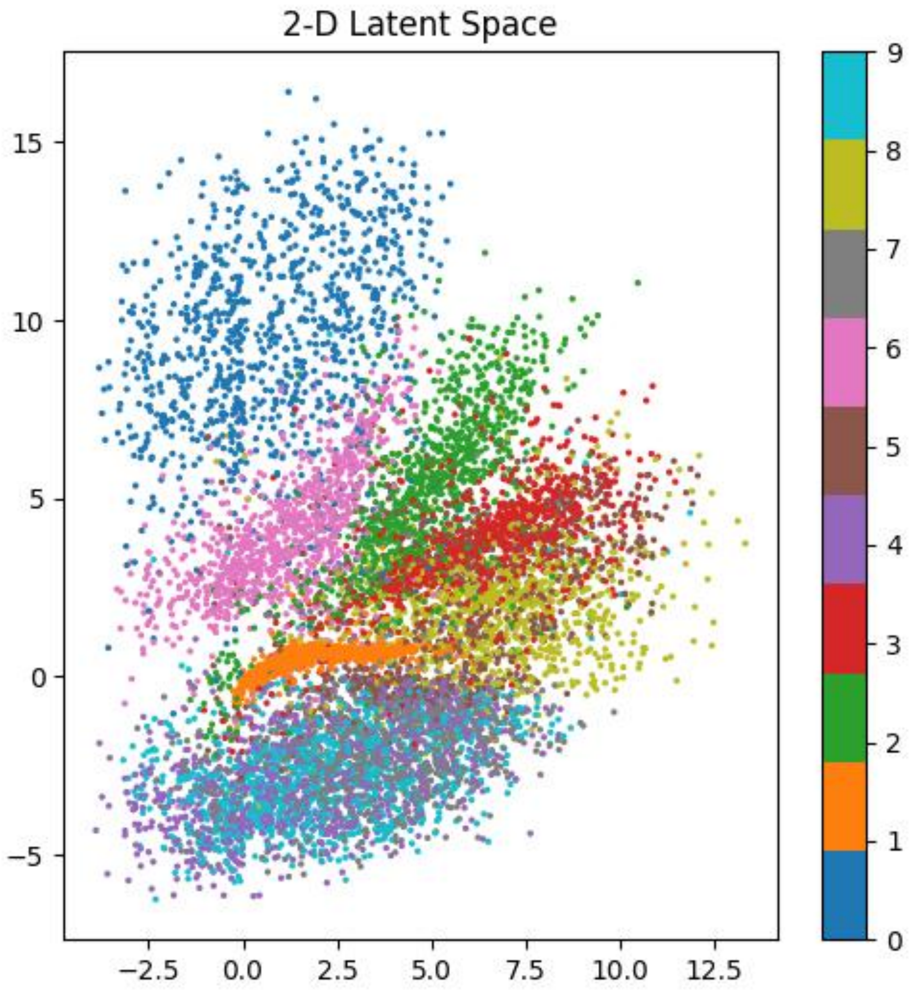


```
Model: "encoder_model"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━┓
┃ Layer (type)                   ┃ Output Shape          ┃      Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━┩
│ encoder_input (InputLayer)     │ (None, 784)           │            0 │
├────────────────────────────────┼───────────────────────┼──────────────┤
│ encoder_dense_1 (Dense)        │ (None, 300)           │      235,500 │
├────────────────────────────────┼───────────────────────┼──────────────┤
│ encoder_leakyrelu_1 (LeakyReLU)│ (None, 300)           │            0 │
├────────────────────────────────┼───────────────────────┼──────────────┤
│ encoder_dense_2 (Dense)        │ (None, 2)             │          602 │
├────────────────────────────────┼───────────────────────┼──────────────┤
│ encoder_output (LeakyReLU)     │ (None, 2)             │            0 │
└────────────────────────────────┴───────────────────────┴──────────────┘
 Total params: 236,102 (922.27 KB)
 Trainable params: 236,102 (922.27 KB)
 Non-trainable params: 0 (0.00 B)
Model: "decoder_model"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓
┃ Layer (type)                    ┃ Output Shape           ┃       Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩
│ decoder_input (InputLayer)      │ (None, 2)              │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ decoder_dense_1 (Dense)         │ (None, 300)            │           900 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ decoder_leakyrelu_1 (LeakyReLU) │ (None, 300)            │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ decoder_dense_2 (Dense)         │ (None, 784)            │       235,984 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ decoder_output (LeakyReLU)      │ (None, 784)            │             0 │
└─────────────────────────────────┴────────────────────────┴───────────────┘
 Total params: 236,884 (925.33 KB)
 Trainable params: 236,884 (925.33 KB)
 Non-trainable params: 0 (0.00 B)
Model: "autoencoder"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓
┃ Layer (type)                    ┃ Output Shape           ┃       Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩
│ AE_input (InputLayer)           │ (None, 784)            │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ encoder_model (Functional)      │ (None, 2)              │       236,102 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ decoder_model (Functional)      │ (None, 784)            │       236,884 │
└─────────────────────────────────┴────────────────────────┴───────────────┘
 Total params: 472,986 (1.80 MB)
 Trainable params: 472,986 (1.80 MB)
 Non-trainable params: 0 (0.00 B)
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 ━━━━━━━━━━━━━━━━━━━━ 1s 0us/step
Epoch 1/20
235/235 ━━━━━━━━━━━━━━━━━━━━ 8s 27ms/step - loss: 0.0681 - val_loss: 0.0550
Epoch 2/20
235/235 ━━━━━━━━━━━━━━━━━━━━ 5s 23ms/step - loss: 0.0547 - val_loss: 0.0531
Epoch 3/20
235/235 ━━━━━━━━━━━━━━━━━━━━ 7s 31ms/step - loss: 0.0530 - val_loss: 0.0519
```








2-D Latent Space
15
10
5
0
−5
−2.5
0.0
2.5
5.0
7.5
10.0
12.5
9
8
7
6
5
4
3
2
1
0