from tensorflow.keras.datasets import mnist

(x\_train, \_), (x\_test, \_) = mnist.load\_data()

x\_train = x\_train.astype('float32') / 255.

x\_test = x\_test.astype('float32') / 255.

x\_train = x\_train.reshape((len(x\_train), 784))

x\_test = x\_test.reshape((len(x\_test), 784))

from tensorflow.keras.models import Model

from tensorflow.keras.layers import Input, Dense

latent\_dim = 32 # Change this to 16, 64 for comparison

input\_img = Input(shape=(784,))

encoded = Dense(latent\_dim, activation='relu')(input\_img)

decoded = Dense(784, activation='sigmoid')(encoded)

autoencoder = Model(input\_img, decoded)

autoencoder.compile(optimizer='adam', loss='binary\_crossentropy')

autoencoder.fit(x\_train, x\_train,

epochs=20,

batch\_size=256,

shuffle=True,

validation\_data=(x\_test, x\_test))

import matplotlib.pyplot as plt

decoded\_imgs = autoencoder.predict(x\_test)

n = 10

plt.figure(figsize=(20, 4))

for i in range(n):

# Original

ax = plt.subplot(2, n, i + 1)

plt.imshow(x\_test[i].reshape(28, 28), cmap='gray')

ax.axis('off')

# Reconstructed

ax = plt.subplot(2, n, i + 1 + n)

plt.imshow(decoded\_imgs[i].reshape(28, 28), cmap='gray')

ax.axis('off')

plt.show()