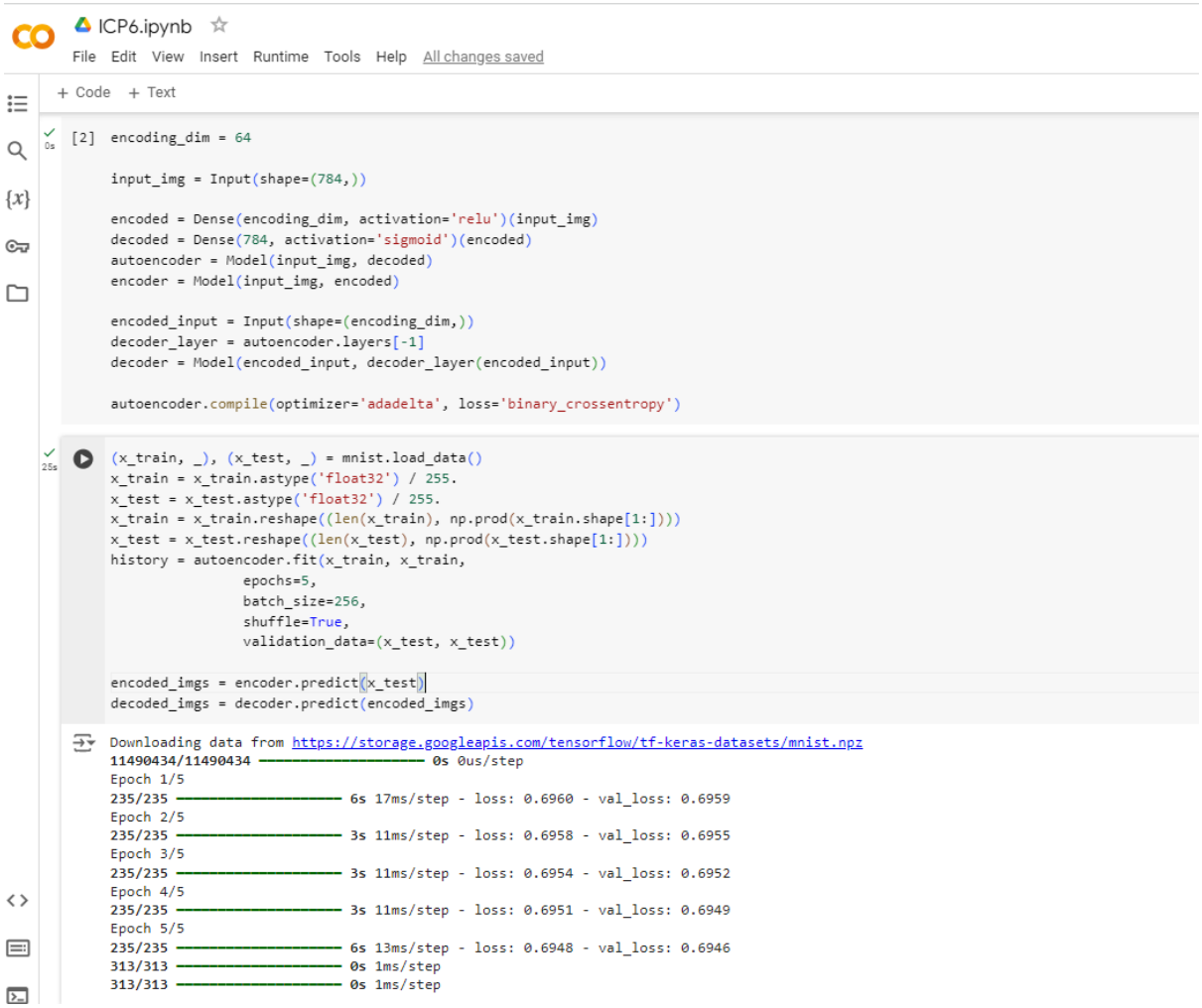
NEURAL NETWORK DEEP LEARNING
ICP 6
700758238
AKHIL KASANAGOTTU

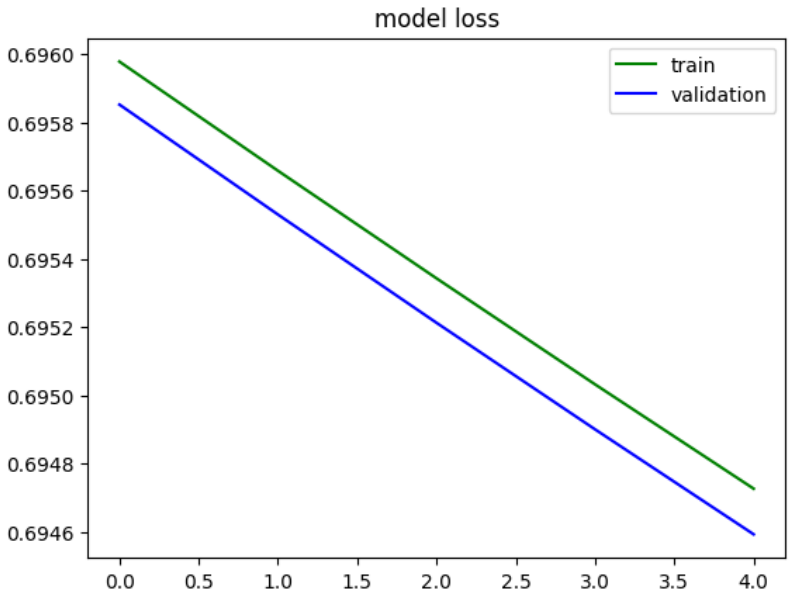
GitHub:
Repository URL for the source code:
https://github.com/axk82380/NNPL/tree/main/ICP_6

Video Link:
https://drive.google.com/file/d/1dOJwO20nMdCU-MkfSiyuI6BAOE8rIpXD/view?usp=sharing



Autoencoder without hidden layer.

```python
# graph
import matplotlib.pyplot as plt
plt.plot(history.history['loss'], color="green")
plt.plot(history.history['val_loss'], color="blue")
plt.title('model loss')
plt.legend(['train', 'validation'], loc='upper right')
plt.show()
```



Graph for validation and training.

ICP6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[5]  input_size = 784
     hidden_size = 128
     code_size = 32

     input_img = Input(shape=(input_size,))
     hidden_1 = Dense(hidden_size, activation='relu')(input_img)
     code = Dense(code_size, activation='relu')(hidden_1)
     hidden_2 = Dense(hidden_size, activation='relu')(code)
     output_img = Dense(input_size, activation='sigmoid')(hidden_2)

     autoencoder = Model(input_img, output_img)
     autoencoder.compile(optimizer='adam', loss='binary_crossentropy')
```

```
[6]  (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), np.prod(x_train.shape[1:])))
     x_test = x_test.reshape((len(x_test), np.prod(x_test.shape[1:])))
     history = autoencoder.fit(x_train, x_train,
                     epochs=5,
                     batch_size=256,
                     shuffle=True,
                     validation_data=(x_test, x_test))
```

```
Epoch 1/5
235/235 ━━━━━━━━━━━━━━━━━━━━ 6s 18ms/step - loss: 0.3247 - val_loss: 0.1528
Epoch 2/5
235/235 ━━━━━━━━━━━━━━━━━━━━ 4s 14ms/step - loss: 0.1427 - val_loss: 0.1200
Epoch 3/5
235/235 ━━━━━━━━━━━━━━━━━━━━ 6s 19ms/step - loss: 0.1187 - val_loss: 0.1097
Epoch 4/5
235/235 ━━━━━━━━━━━━━━━━━━━━ 4s 14ms/step - loss: 0.1090 - val_loss: 0.1035
Epoch 5/5
235/235 ━━━━━━━━━━━━━━━━━━━━ 5s 15ms/step - loss: 0.1039 - val_loss: 0.0998
```

Autoencoder with hidden layer. Here the validation loss is 0.0998.

ICP6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

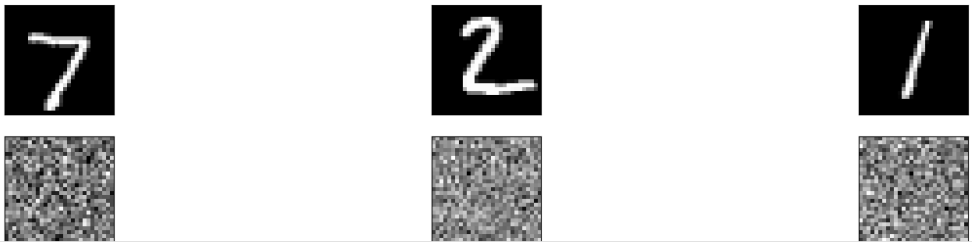
```
[7]  encoded_imgs = encoder.predict(x_test)
     decoded_imgs = decoder.predict(encoded_imgs)

     import matplotlib.pyplot as plt

     n = 3
     plt.figure(figsize=(20, 4))
     for i in range(n):
         # display original
         ax = plt.subplot(2, n, i + 1)
         plt.imshow(x_test[i].reshape(28, 28))
         plt.gray()
         ax.get_xaxis().set_visible(False)
         ax.get_yaxis().set_visible(False)

         # display reconstruction
         ax = plt.subplot(2, n, i + 1 + n)
         plt.imshow(decoded_imgs[i].reshape(28, 28))
         plt.gray()
         ax.get_xaxis().set_visible(False)
         ax.get_yaxis().set_visible(False)
     plt.show()
```

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
313/313 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
```



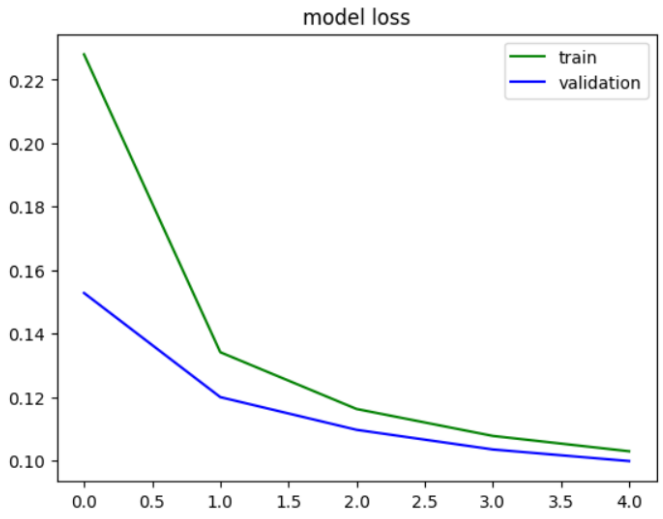
Printing original and reconstructed images

ICP6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[8] # graph
    plt.plot(history.history['loss'], color="green")
    plt.plot(history.history['val_loss'], color="blue")
    plt.title('model loss')
    plt.legend(['train', 'validation'], loc='upper right')
    plt.show()
```



Printing model loss after adding hidden layer.

ICP6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[9] Compression factor: 12.25
/usr/local/lib/python3.10/dist-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Model: "sequential"
```

Layer (type)	Output Shape	Param #
dense_6 (Dense)	(None, 64)	50,240
dense_7 (Dense)	(None, 784)	50,960

```
Total params: 101,200 (395.31 KB)
Trainable params: 101,200 (395.31 KB)
Non-trainable params: 0 (0.00 B)
Model: "functional_6"
```

Layer (type)	Output Shape	Param #
input_layer_4 (InputLayer)	(None, 784)	0
dense_6 (Dense)	(None, 64)	50,240

```
Total params: 50,240 (196.25 KB)
Trainable params: 50,240 (196.25 KB)
Non-trainable params: 0 (0.00 B)
Epoch 1/5
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 11ms/step - loss: 0.3517 - val_loss: 0.1604
Epoch 2/5
235/235 ━━━━━━━━━━━━━━━━━━━━ 6s 15ms/step - loss: 0.1503 - val_loss: 0.1257
Epoch 3/5
235/235 ━━━━━━━━━━━━━━━━━━━━ 4s 11ms/step - loss: 0.1224 - val_loss: 0.1087
Epoch 4/5
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 11ms/step - loss: 0.1072 - val_loss: 0.0980
Epoch 5/5
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 10ms/step - loss: 0.0972 - val_loss: 0.0904
313/313 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
```

Computing and printing compression factor, and validation loss.