ICP-6

NAME: GOWTHAM SAI REDDY MADDIREDDY

STUDENT ID: 700759075

GITHUB: [ICP-6/700759075_ICP 6.ipynb at main · Gowthamsai08/ICP-6]()

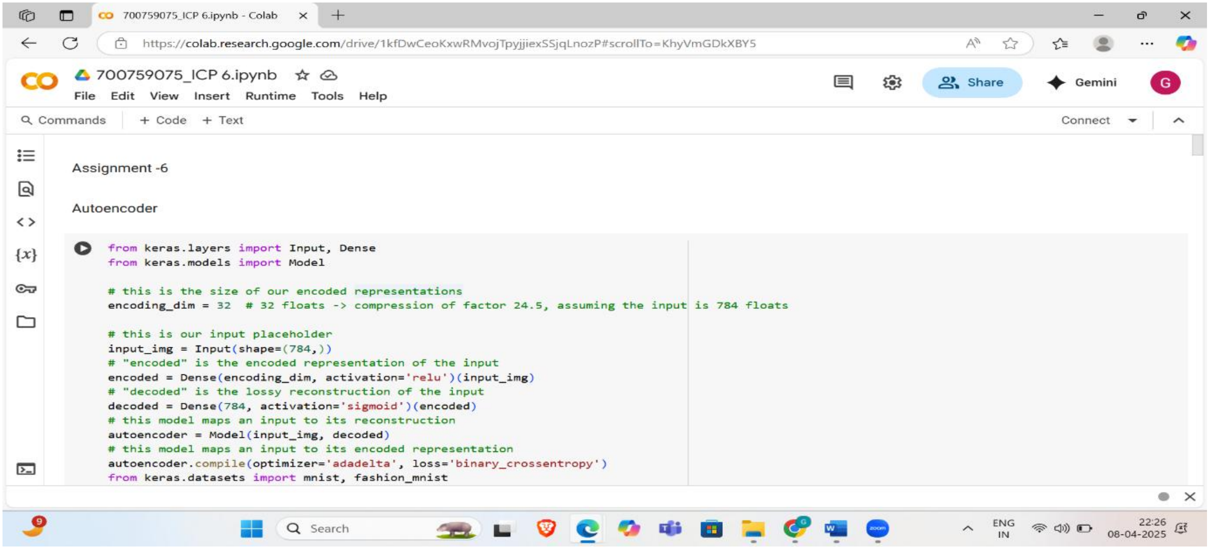
VIDEO LINK:
https://drive.google.com/file/d/18ihOBjDY6ocdojPwGOpxx4UB3yMTOz5U/view?usp=drive_link

1. Add one more hidden layer to autoencoder

2. Do the prediction on the test data and then visualize one of the reconstructed version of that test data. Also, visualize the same test data before reconstruction using Matplotlib

3. Repeat the question 2 on the denoisening autoencoder

4. plot loss and accuracy using the history object



```
                shuffle=True,
                validation_data=(x_test, x_test))
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-labels-idx1-ubyte.gz
29515/29515 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-images-idx3-ubyte.gz
26421880/26421880 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-labels-idx1-ubyte.gz
5148/5148 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-images-idx3-ubyte.gz
4422102/4422102 [==============================] - 0s 0us/step
Epoch 1/5
235/235 [==============================] - 9s 29ms/step - loss: 0.6965 - val_loss: 0.6964
Epoch 2/5
235/235 [==============================] - 4s 15ms/step - loss: 0.6962 - val_loss: 0.6961
Epoch 3/5
235/235 [==============================] - 4s 17ms/step - loss: 0.6960 - val_loss: 0.6959
Epoch 4/5
235/235 [==============================] - 4s 15ms/step - loss: 0.6958 - val_loss: 0.6957
Epoch 5/5
235/235 [==============================] - 3s 15ms/step - loss: 0.6956 - val_loss: 0.6955
<keras.src.callbacks.History at 0x7bd676228a90>
```

1.Adding hidden layer to Autoencoder

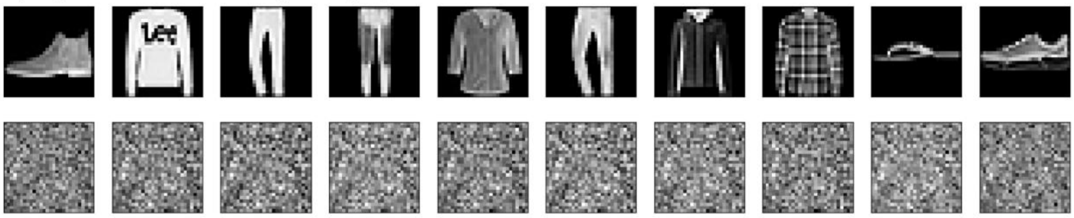
1.Adding hidden layer to Autoencoder

```
from keras.layers import Input, Dense
from keras.models import Model
from keras.datasets import mnist, fashion_mnist
import numpy as np

# this is the size of our encoded representations
encoding_dim = 32
# this is our input placeholder
input_img = Input(shape=(784,))
# "encoded" is the encoded representation of the input
encoded = Dense(encoding_dim, activation='relu')(input_img)

# Adding an additional hidden layer
hidden_layer_dim = 64
hidden_layer = Dense(hidden_layer_dim, activation='relu')(encoded)

# "decoded" is the lossy reconstruction of the input, now connected to the hidden layer instead of 'encoded'
decoded = Dense(784, activation='sigmoid')(hidden_layer)

# this model maps an input to its reconstruction
```

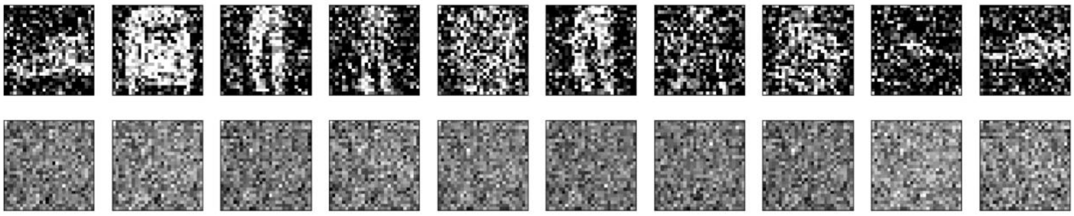
2.Prediction on the test data and then visualize one of the reconstructed version of that test data. Also, visualize the same test data before reconstruction using Matplotlib.

```
from keras.layers import Input, Dense
from keras.models import Model
from keras.datasets import mnist, fashion_mnist
import numpy as np
import matplotlib.pyplot as plt

# Define the model architecture
encoding_dim = 32
hidden_layer_dim = 64

input_img = Input(shape=(784,))
encoded = Dense(encoding_dim, activation='relu')(input_img)
hidden_layer = Dense(hidden_layer_dim, activation='relu')(encoded)  # Additional hidden layer
decoded = Dense(784, activation='sigmoid')(hidden_layer)

autoencoder = Model(input_img, decoded)
autoencoder.compile(optimizer='adadelta', loss='binary_crossentropy')

# Load and prepare data
```

```
Epoch 1/5
235/235 [==============================] - 9s 32ms/step - loss: 0.6925 - val_loss: 0.6924
Epoch 2/5
235/235 [==============================] - 6s 24ms/step - loss: 0.6923 - val_loss: 0.6922
Epoch 3/5
235/235 [==============================] - 4s 18ms/step - loss: 0.6921 - val_loss: 0.6920
Epoch 4/5
235/235 [==============================] - 3s 13ms/step - loss: 0.6919 - val_loss: 0.6918
Epoch 5/5
235/235 [==============================] - 4s 16ms/step - loss: 0.6918 - val_loss: 0.6917
313/313 [==============================] - 1s 2ms/step
```



```
Epoch 15/20
235/235 [==============================] - 3s 13ms/step - loss: 0.6920 - val_loss: 0.6919
Epoch 16/20
235/235 [==============================] - 2s 11ms/step - loss: 0.6917 - val_loss: 0.6916
Epoch 17/20
235/235 [==============================] - 3s 12ms/step - loss: 0.6915 - val_loss: 0.6914
Epoch 18/20
235/235 [==============================] - 3s 13ms/step - loss: 0.6913 - val_loss: 0.6912
Epoch 19/20
235/235 [==============================] - 3s 15ms/step - loss: 0.6911 - val_loss: 0.6909
Epoch 20/20
235/235 [==============================] - 3s 11ms/step - loss: 0.6908 - val_loss: 0.6907
313/313 [==============================] - 1s 2ms/step
```



700759075_ICP 6.ipynb - Colab
https://colab.research.google.com/drive/1kfDwCeoKxwRMvojTpyjjiexSSjqLnozP#scrollTo=KhyVmGDkXBY5
700759075_ICP 6.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Connect

4.Plot loss and accuracy using the history object

```
from keras.layers import Input, Dense
from keras.models import Model
from keras.datasets import fashion_mnist
from keras.utils import to_categorical
import numpy as np
import matplotlib.pyplot as plt
from keras.optimizers import Adam

# Load and prepare the Fashion MNIST data
(x_train, y_train), (x_test, y_test) = fashion_mnist.load_data()
x_train = x_train.reshape(-1, 784).astype('float32') / 255
x_test = x_test.reshape(-1, 784).astype('float32') / 255

# Convert labels to one-hot encoding
num_classes = 10
y_train = to_categorical(y_train, num_classes)
y_test = to_categorical(y_test, num_classes)

# Model architecture
```

Search ENG IN 22:27 08-04-2025


700759075_ICP 6.ipynb - Colab
https://colab.research.google.com/drive/1kfDwCeoKxwRMvojTpyjjiexSSjqLnozP#scrollTo=KhyVmGDkXBY5
700759075_ICP 6.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text
Share
Gemini
Connect
235/235 [==============================] - 2s 7ms/step - loss: 0.2838 - accuracy: 0.8978 - val_loss: 0.3479 - val_accuracy: 0.8768
Training and Validation Accuracy
Training Accuracy
Validation Accuracy
Accuracy
0.90
0.88
0.86
0.84
0.82
0.80
Training and Validation Loss
Training Loss
Validation Loss
Loss
0.60
0.55
0.50
0.45
0.40
0.35
0.30
Search
ENG
IN
22:27
08-04-2025