

Neural Networks & Deep Learning Assignment-8

Harshavardhan Reddy Pattepur
700754833

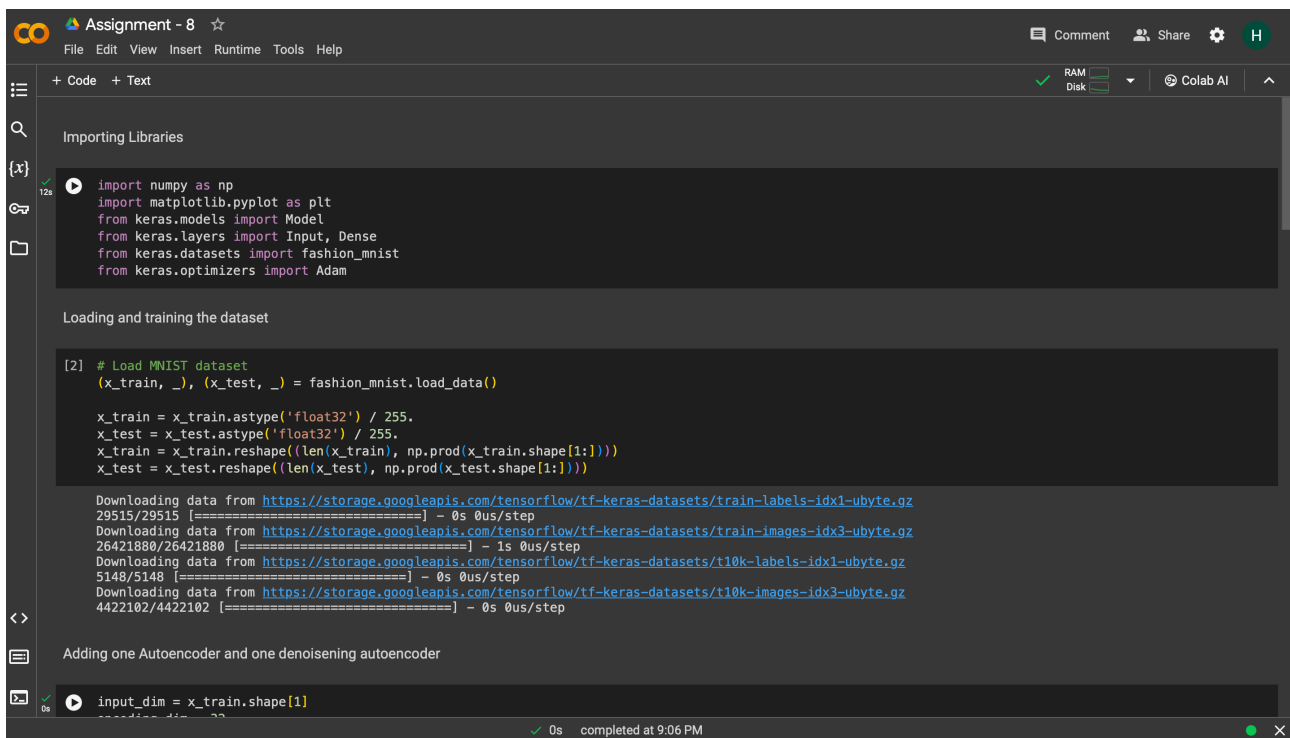
Repository Link :

<https://github.com/harshavardhanreddy27/NNDL-ASSIGNMENT---8>

Video Link:

https://drive.google.com/file/d/1RhgHQz7q_R-V6RSIZA55eyy7E0JBDL6t/view?usp=share_link

Code Screenshots:



```
Assignment - 8 ☆
File Edit View Insert Runtime Tools Help

+ Code + Text
RAM
Disk
Colab AI
H

Importing Libraries

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

Loading and training the dataset

[2] # Load MNIST dataset
(x_train, _), (x_test, _) = fashion_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:])))

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 [=====] - 1s 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

Adding one Autoencoder and one denoising autoencoder

input_dim = x_train.shape[1]
```

Assignment - 8

File Edit View Insert Runtime Tools Help All changes saved

Comment Share H

+ Code + Text

RAM Disk Colab AI

Adding one Autoencoder and one denoising autoencoder

```
input_dim = x_train.shape[1]
encoding_dim = 32

input_img = Input(shape=(input_dim,))
encoded = Dense(128, activation='relu')(input_img)
encoded = Dense(encoding_dim, activation='relu')(encoded)
decoded = Dense(128, activation='relu')(encoded)
decoded = Dense(input_dim, activation='sigmoid')(decoded)

autoencoder = Model(input_img, decoded)
```

```
[4] autoencoder.compile(optimizer='adam', loss='binary_crossentropy')
history = autoencoder.fit(x_train, x_train, epochs=10, batch_size=256, shuffle=True, validation_data=(x_test, x_test))
```

Epoch 1/10
235/235 [=====] - 8s 21ms/step - loss: 0.3715 - val_loss: 0.3141
Epoch 2/10
235/235 [=====] - 4s 16ms/step - loss: 0.3048 - val_loss: 0.3015
Epoch 3/10
235/235 [=====] - 4s 18ms/step - loss: 0.2959 - val_loss: 0.2951
Epoch 4/10
235/235 [=====] - 4s 19ms/step - loss: 0.2911 - val_loss: 0.2917
Epoch 5/10
235/235 [=====] - 4s 16ms/step - loss: 0.2883 - val_loss: 0.2891
Epoch 6/10
235/235 [=====] - 4s 18ms/step - loss: 0.2861 - val_loss: 0.2875
Epoch 7/10
235/235 [=====] - 4s 19ms/step - loss: 0.2844 - val_loss: 0.2862
Epoch 8/10
235/235 [=====] - 4s 16ms/step - loss: 0.2831 - val_loss: 0.2851
Epoch 9/10
235/235 [=====] - 4s 18ms/step - loss: 0.2819 - val_loss: 0.2837
Epoch 10/10
235/235 [=====] - 5s 19ms/step - loss: 0.2810 - val_loss: 0.2835

0s completed at 9:06 PM

Assignment - 8

File Edit View Insert Runtime Tools Help All changes saved

Comment Share H

+ Code + Text

RAM Disk Colab AI

Epoch 5/10
235/235 [=====] - 4s 16ms/step - loss: 0.2883 - val_loss: 0.2891
Epoch 6/10
235/235 [=====] - 4s 18ms/step - loss: 0.2861 - val_loss: 0.2875
Epoch 7/10
235/235 [=====] - 4s 19ms/step - loss: 0.2844 - val_loss: 0.2862
Epoch 8/10
235/235 [=====] - 4s 16ms/step - loss: 0.2831 - val_loss: 0.2851
Epoch 9/10
235/235 [=====] - 4s 18ms/step - loss: 0.2819 - val_loss: 0.2837
Epoch 10/10
235/235 [=====] - 5s 19ms/step - loss: 0.2810 - val_loss: 0.2835

Visualaizing Reconstruction of the data

```
decoded_imgs = autoencoder.predict(x_test)

n = 10
plt.figure(figsize=(20, 4))
for i in range(n):

    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)

    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()
```

0s completed at 9:06 PM

