NNDL – ICP4

Name: Divya Chilukuri Student ID: 700744169

Assignment4.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
from keras.layers import Input, Dense
from keras.models import Model

# this is the size of our encoded representations
encoding_dim = 32  # 32 floats -> compression of factor 24.5, assuming the input is 784 floa

# 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)
# "decoded" is the lossy reconstruction of the input
decoded = Dense(784, activation='sigmoid')(encoded)
# this model maps an input to its reconstruction
autoencoder = Model(input_img, decoded)
# this model maps an input to its encoded representation
autoencoder.compile(optimizer='adadelta', loss='binary_crossentropy', metrics ='accuracy')
from keras.datasets import mnist, fashion_mnist
import numpy as np
(x_train, y_train), (x_test, y_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:])))

autoencoder.fit(x_train, x_train,
                epochs=5,
                batch_size=256,
                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 [==============================] - 10s 31ms/step - loss: 0.6955 - accuracy: 0.0028 - val_loss: 0.6954 - val_accuracy: 0.0033
Epoch 2/5
235/235 [==============================] - 5s 22ms/step - loss: 0.6953 - accuracy: 0.0028 - val_loss: 0.6952 - val_accuracy: 0.0031
Epoch 3/5
235/235 [==============================] - 3s 12ms/step - loss: 0.6951 - accuracy: 0.0030 - val_loss: 0.6950 - val_accuracy: 0.0033
Epoch 4/5
235/235 [==============================] - 5s 20ms/step - loss: 0.6949 - accuracy: 0.0031 - val_loss: 0.6948 - val_accuracy: 0.0033
Epoch 5/5
235/235 [==============================] - 6s 27ms/step - loss: 0.6947 - accuracy: 0.0031 - val_loss: 0.6947 - val_accuracy: 0.0032
<keras.callbacks.History at 0x7cc500199c30>
```

Github Link: https://github.com/DivyaReddy1999/NNDL_Assignment4.git

NNDL – ICP4

Name: Divya Chilukuri Student ID: 700744169

```
from keras.layers import Input, Dense
from keras.models import Model

# this is the size of our encoded representations
encoding_dim = 32  # 32 floats -> compression of factor 24.5, assuming the input is 784 floats

# 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)
# "decoded" is the lossy reconstruction of the input
decoded = Dense(784, activation='sigmoid')(encoded)
# this model maps an input to its reconstruction
autoencoder = Model(input_img, decoded)
# this model maps an input to its encoded representation
autoencoder.compile(optimizer='adadelta', loss='binary_crossentropy', metrics ='accuracy')
from keras.datasets import mnist, fashion_mnist
import numpy as np
(x_train, y_train), (x_test, y_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:])))

autoencoder.fit(x_train, x_train,
                epochs=5,
                batch_size=256,
                shuffle=True,
                validation_data=(x_test, x_test))
```

```
Epoch 1/5
235/235 [==============================] - 3s 10ms/step - loss: 0.6990 - accuracy: 5.8333e-04 - val_loss: 0.6988 - val_accuracy: 5.0000e-04
Epoch 2/5
235/235 [==============================] - 3s 11ms/step - loss: 0.6986 - accuracy: 7.0000e-04 - val_loss: 0.6984 - val_accuracy: 4.0000e-04
Epoch 3/5
235/235 [==============================] - 3s 15ms/step - loss: 0.6983 - accuracy: 7.3333e-04 - val_loss: 0.6981 - val_accuracy: 5.0000e-04
Epoch 4/5
235/235 [==============================] - 2s 11ms/step - loss: 0.6979 - accuracy: 7.3333e-04 - val_loss: 0.6977 - val_accuracy: 6.0000e-04
Epoch 5/5
235/235 [==============================] - 2s 10ms/step - loss: 0.6976 - accuracy: 7.5000e-04 - val_loss: 0.6974 - val_accuracy: 6.0000e-04
<keras.callbacks.History at 0x7cc4c6c8a050>
```

Github Link: https://github.com/DivyaReddy1999/NNDL_Assignment4.git

Name: Divya Chilukuri Student ID: 700744169

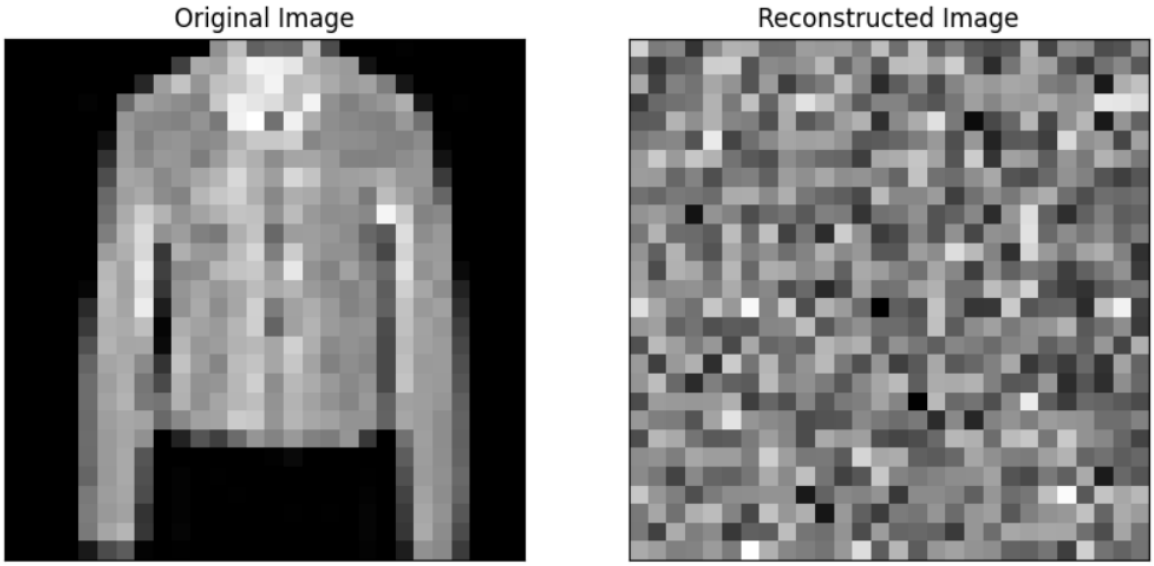
```
import matplotlib.pyplot as plt

# Get the reconstructed images for the test set
reconstructed_imgs = autoencoder.predict(x_test)

# Choose a random image from the test set
n = 10  # index of the image to be plotted
plt.figure(figsize=(10, 5))

# Plot the original image
ax = plt.subplot(1, 2, 1)
plt.imshow(x_test[n].reshape(28, 28))
plt.gray()
ax.get_xaxis().set_visible(False)
ax.get_yaxis().set_visible(False)
ax.set_title("Original Image")

# Plot the reconstructed image
ax = plt.subplot(1, 2, 2)
plt.imshow(reconstructed_imgs[n].reshape(28, 28))
plt.gray()
ax.get_xaxis().set_visible(False)
ax.get_yaxis().set_visible(False)
ax.set_title("Reconstructed Image")

plt.show()
```

```
313/313 [==============================] - 0s 1ms/step
```

Github Link: https://github.com/DivyaReddy1999/NNDL_Assignment4.git

NNDL – ICP4

Name: Divya Chilukuri Student ID: 700744169




```
from keras.layers import Input, Dense
from keras.models import Model

# this is the size of our encoded representations
encoding_dim = 32  # 32 floats -> compression of factor 24.5, assuming the input is 784 floats

# 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)
# "decoded" is the lossy reconstruction of the input
decoded = Dense(784, activation='sigmoid')(encoded)
# this model maps an input to its reconstruction
autoencoder = Model(input_img, decoded)
# this model maps an input to its encoded representation
autoencoder.compile(optimizer='adadelta', loss='binary_crossentropy',metrics ='accuracy')
from keras.datasets import fashion_mnist
import numpy as np
(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:])))
```

```
autoencoder.fit(x_train_noisy, x_train,
                epochs=10,
                batch_size=256,
                shuffle=True,
                validation_data=(x_test_noisy, x_test_noisy))
```

Github Link: https://github.com/DivyaReddy1999/NNDL_Assignment4.git

Name: Divya Chilukuri Student ID: 700744169

```
Epoch 1/10
235/235 [==============================] - 3s 11ms/step - loss: 0.6992 - accuracy: 9.5000e-04 - val_loss: 0.6989 - val_accuracy: 8.0000e-04
Epoch 2/10
235/235 [==============================] - 3s 13ms/step - loss: 0.6988 - accuracy: 9.5000e-04 - val_loss: 0.6986 - val_accuracy: 8.0000e-04
Epoch 3/10
235/235 [==============================] - 3s 11ms/step - loss: 0.6985 - accuracy: 9.6667e-04 - val_loss: 0.6983 - val_accuracy: 8.0000e-04
Epoch 4/10
235/235 [==============================] - 2s 10ms/step - loss: 0.6982 - accuracy: 9.5000e-04 - val_loss: 0.6980 - val_accuracy: 8.0000e-04
Epoch 5/10
235/235 [==============================] - 2s 9ms/step - loss: 0.6979 - accuracy: 9.3333e-04 - val_loss: 0.6977 - val_accuracy: 8.0000e-04
Epoch 6/10
235/235 [==============================] - 2s 10ms/step - loss: 0.6976 - accuracy: 9.3333e-04 - val_loss: 0.6974 - val_accuracy: 8.0000e-04
Epoch 7/10
235/235 [==============================] - 3s 13ms/step - loss: 0.6973 - accuracy: 9.1667e-04 - val_loss: 0.6971 - val_accuracy: 8.0000e-04
Epoch 8/10
235/235 [==============================] - 3s 11ms/step - loss: 0.6970 - accuracy: 9.1667e-04 - val_loss: 0.6969 - val_accuracy: 8.0000e-04
Epoch 9/10
235/235 [==============================] - 2s 10ms/step - loss: 0.6968 - accuracy: 9.3333e-04 - val_loss: 0.6966 - val_accuracy: 9.0000e-04
Epoch 10/10
235/235 [==============================] - 2s 10ms/step - loss: 0.6965 - accuracy: 9.3333e-04 - val_loss: 0.6964 - val_accuracy: 0.0010
<keras.callbacks.History at 0x7cc500912da0>
```

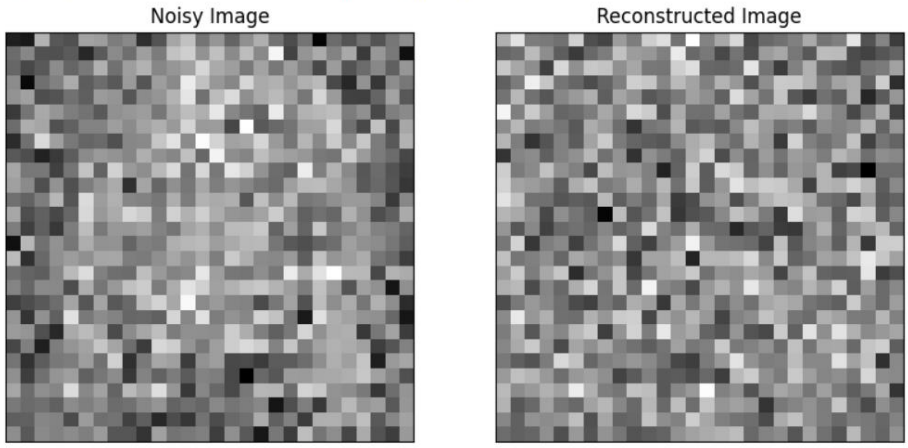
```python
import matplotlib.pyplot as plt

# Get the reconstructed images for the test set
reconstructed_imgs = autoencoder.predict(x_test_noisy)

# Choose a random image from the test set
n = 10  # index of the image to be plotted
plt.figure(figsize=(10, 5))

# Plot the original noisy image
ax = plt.subplot(1, 2, 1)
plt.imshow(x_test_noisy[n].reshape(28, 28))
plt.gray()
ax.get_xaxis().set_visible(False)
ax.get_yaxis().set_visible(False)
ax.set_title("Noisy Image")

# Plot the reconstructed image
ax = plt.subplot(1, 2, 2)
plt.imshow(reconstructed_imgs[n].reshape(28, 28))
plt.gray()
ax.get_xaxis().set_visible(False)
ax.get_yaxis().set_visible(False)
ax.set_title("Reconstructed Image")

plt.show()
```

Github Link: https://github.com/DivyaReddy1999/NNDL_Assignment4.git

NNDL – ICP4

Name: Divya Chilukuri Student ID: 700744169

```
313/313 [==============================] - 0s 1ms/step
```

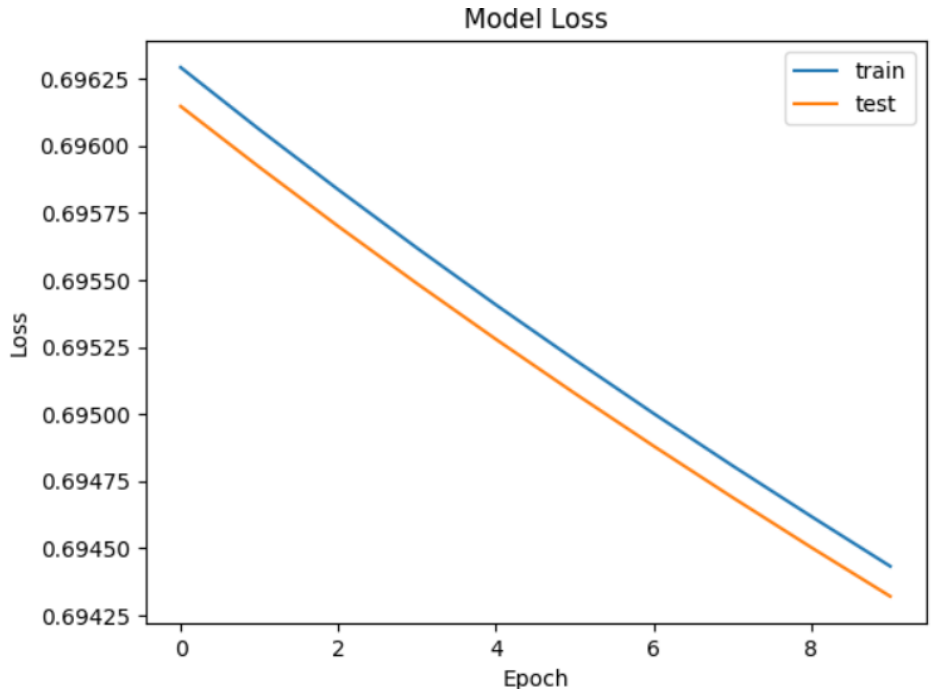


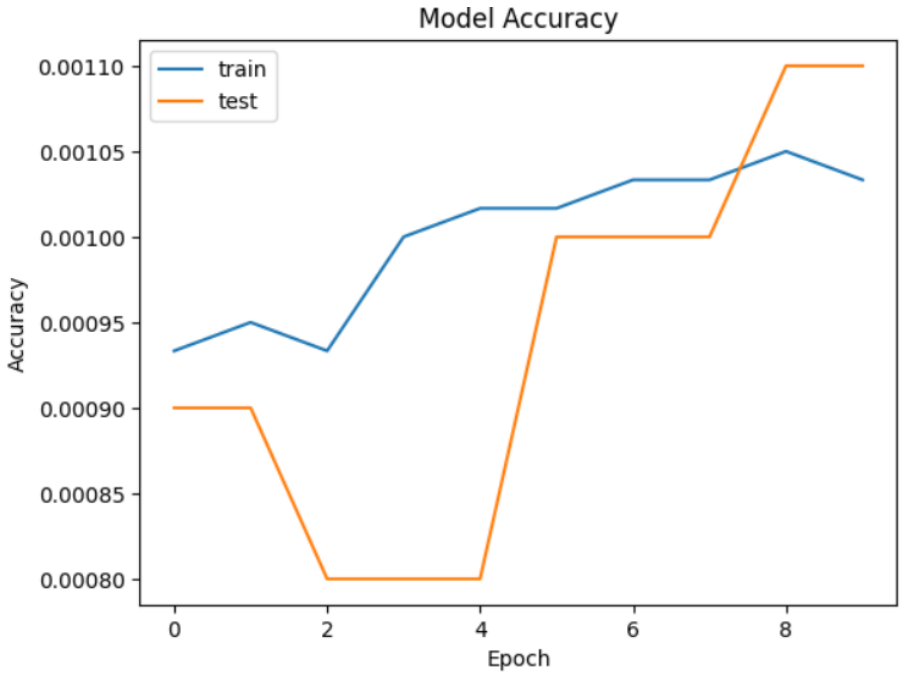

```python
import matplotlib.pyplot as plt

# Train the autoencoder
history = autoencoder.fit(x_train_noisy, x_train,
                epochs=10,
                batch_size=256,
                shuffle=True,
                validation_data=(x_test_noisy, x_test_noisy))

# Plot the loss
plt.plot(history.history['loss'], label='train')
plt.plot(history.history['val_loss'], label='test')
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend()
plt.show()

# Plot the accuracy
plt.plot(history.history['accuracy'], label='train')
plt.plot(history.history['val_accuracy'], label='test')
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend()
plt.show()
```

Github Link: https://github.com/DivyaReddy1999/NNDL_Assignment4.git

NNDL – ICP4

Name: Divya Chilukuri Student ID: 700744169





Github Link: https://github.com/DivyaReddy1999/NNDL_Assignment4.git