CS156 (Introduction to AI), Fall 2022

Homework 9 submission

Roster Name: Preet LNU

Student ID: 014755741

Email address: preet.lnu@sjsu.edu

References and sources

Autoencoders.MNIST.ipynb

Solution

Load libraries and set random number generator seed

```
import numpy as np
import tensorflow as tf
from tensorflow import keras
from sklearn.model_selection import train_test_split

from tensorflow.keras import layers
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Reshape
from tensorflow.keras.layers import Input
from tensorflow.keras.models import Model

import matplotlib.pyplot as plt
```

```
np.random.seed(42)
```

Code the solution

Loading and preparing image data

```
(x_train_valid, y_train_valid), (x_test, y_test) = keras.datasets.fashion_mnist.load_c

x_train, x_validation, y_train, y_validation = train_test_split(x_train_valid, y_trair

x_train = x_train.astype("float32") / 255
x_validation = x_validation.astype("float32") / 255
x_test = x_test.astype("float32") / 255

x_train.shape, x_validation.shape, x_test.shape
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-dataset:
29515/29515 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-dataset:
26421880/26421880 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-dataset:
5148/5148 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-dataset:
4422102/4422102 [==============================] - 0s 0us/step
((48000, 28, 28), (12000, 28, 28), (10000, 28, 28))
```

Autoencoder model and its performance for first 10 images in test set

```
x_train = x_train.reshape(-1, 784)
x_validation = x_validation.reshape(-1, 784)
x_test = x_test.reshape(-1, 784)

x_train.shape, x_validation.shape, x_test.shape
```

```
((48000, 784), (12000, 784), (10000, 784))
```

```
input_layer = Input(shape=(784,))
encoded = layers.Dense(128, activation='relu')(input_layer)
encoded = layers.Dense(64, activation='relu')(encoded)
encoded = layers.Dense(32, activation='relu')(encoded)

decoded = layers.Dense(64, activation='sigmoid')(encoded)
decoded = layers.Dense(128, activation='sigmoid')(decoded)
decoded = layers.Dense(784, activation='sigmoid')(decoded)

autoencoder = keras.Model(input_layer, decoded)
autoencoder.summary()

encoder = keras.Model(input_layer, encoded)

encoded_input = keras.Input(shape=(128,))
```

```
decoder_layer = autoencoder.layers[-1]
decoder = keras.Model(encoded_input, decoder_layer(encoded_input))
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 784)]             0

 dense (Dense)               (None, 128)               100480

 dense_1 (Dense)             (None, 64)                8256

 dense_2 (Dense)             (None, 32)                2080

 dense_3 (Dense)             (None, 64)                2112

 dense_4 (Dense)             (None, 128)               8320

 dense_5 (Dense)             (None, 784)               101136

=================================================================
Total params: 222,384
Trainable params: 222,384
Non-trainable params: 0
_________________________________________________________________
```

Fitting/training the model

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

autoencoder.fit(x_train, x_train,
                epochs=30,
                batch_size=2048,
                shuffle=True,
                validation_data=(x_test, x_test))
```

```
Epoch 3/30
24/24 [==============================] - 5s 200ms/step - loss: 0.4873 - val_loss
Epoch 4/30
24/24 [==============================] - 4s 160ms/step - loss: 0.4742 - val_loss
Epoch 5/30
24/24 [==============================] - 3s 107ms/step - loss: 0.4514 - val_loss
Epoch 6/30
24/24 [==============================] - 2s 104ms/step - loss: 0.4331 - val_loss
Epoch 7/30
24/24 [==============================] - 3s 108ms/step - loss: 0.4181 - val_loss
Epoch 8/30
24/24 [==============================] - 2s 102ms/step - loss: 0.4037 - val_loss

Epoch 9/30
24/24 [==============================] - 2s 104ms/step - loss: 0.3967 - val_loss
Epoch 10/30
24/24 [==============================] - 2s 102ms/step - loss: 0.3899 - val_loss
```

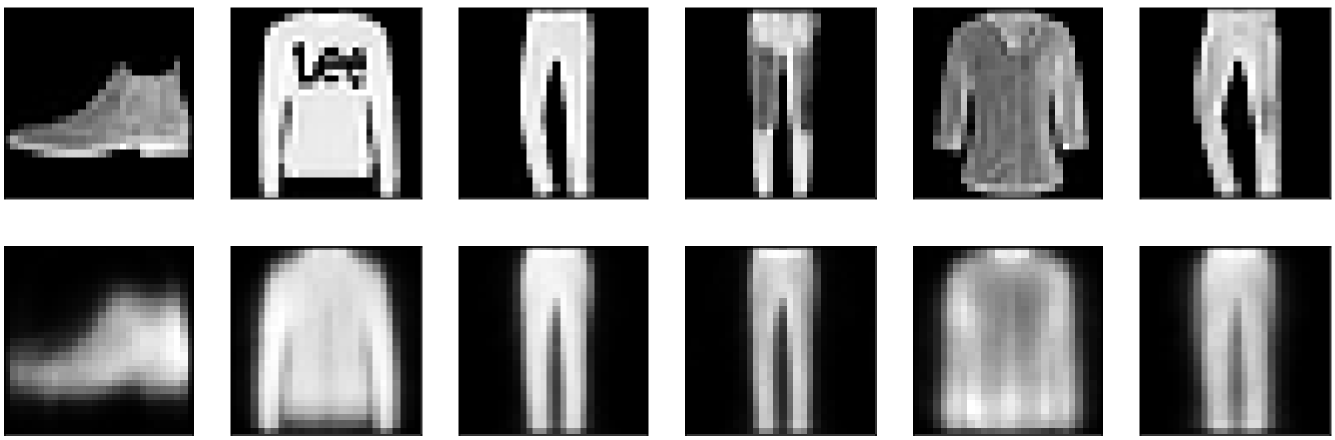
```
Epoch 11/30
24/24 [==============================] - 2s 102ms/step - loss: 0.3850 - val_loss
Epoch 12/30
24/24 [==============================] - 3s 105ms/step - loss: 0.3817 - val_loss
Epoch 13/30
24/24 [==============================] - 3s 110ms/step - loss: 0.3787 - val_loss
Epoch 14/30
24/24 [==============================] - 3s 108ms/step - loss: 0.3761 - val_loss
Epoch 15/30
24/24 [==============================] - 2s 104ms/step - loss: 0.3730 - val_loss
Epoch 16/30
24/24 [==============================] - 3s 109ms/step - loss: 0.3700 - val_loss
Epoch 17/30
24/24 [==============================] - 3s 109ms/step - loss: 0.3667 - val_loss
Epoch 18/30
24/24 [==============================] - 3s 107ms/step - loss: 0.3633 - val_loss
Epoch 19/30
24/24 [==============================] - 3s 106ms/step - loss: 0.3602 - val_loss
Epoch 20/30
24/24 [==============================] - 3s 107ms/step - loss: 0.3571 - val_loss
Epoch 21/30
24/24 [==============================] - 2s 104ms/step - loss: 0.3546 - val_loss
Epoch 22/30
24/24 [==============================] - 3s 108ms/step - loss: 0.3517 - val_loss
Epoch 23/30
24/24 [==============================] - 3s 108ms/step - loss: 0.3497 - val_loss
Epoch 24/30
24/24 [==============================] - 2s 104ms/step - loss: 0.3477 - val_loss
Epoch 25/30
24/24 [==============================] - 3s 105ms/step - loss: 0.3461 - val_loss
Epoch 26/30
24/24 [==============================] - 2s 104ms/step - loss: 0.3448 - val_loss
Epoch 27/30
24/24 [==============================] - 3s 106ms/step - loss: 0.3428 - val_loss
Epoch 28/30
24/24 [==============================] - 3s 105ms/step - loss: 0.3413 - val_loss
Epoch 29/30
24/24 [==============================] - 2s 101ms/step - loss: 0.3402 - val_loss
Epoch 30/30
24/24 [==============================] - 3s 106ms/step - loss: 0.3390 - val_loss
<keras.callbacks.History at 0x7ffa56bb3750>
```

Plotting the reconstructed images

```
predictions = 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(predictions[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
plt.show()
```

```
313/313 [==============================] - 1s 2ms/step
```



Reshaping and denoising images with an autoencoder model + plotting

```
x_train = x_train.reshape(-1, 28, 28, 1)
x_validation = x_validation.reshape(-1, 28, 28, 1)
x_test = x_test.reshape(-1, 28, 28, 1)

x_train.shape, x_validation.shape, x_test.shape
```

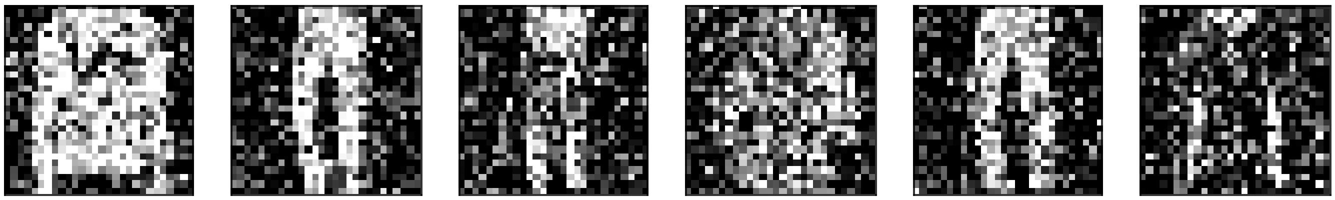
```
((48000, 28, 28, 1), (12000, 28, 28, 1), (10000, 28, 28, 1))
```

```
noise_factor = 0.4
x_train_noisy = x_train + noise_factor * np.random.normal(loc=0.0, scale=1.0, size=x_t
x_validation_noisy = x_validation + noise_factor * np.random.normal(loc=0.0, scale=1.(
x_test_noisy = x_test + noise_factor * np.random.normal(loc=0.0, scale=1.0, size=x_tes

x_train_noisy = np.clip(x_train_noisy, 0., 1.)
x_validation_noisy = np.clip(x_validation_noisy, 0., 1.)
x_test_noisy = np.clip(x_test_noisy, 0., 1.)
```

Plotting noised images

```
n = 10
```

```
plt.figure(figsize=(20, 2))
for i in range(1, n + 1):
    ax = plt.subplot(1, n, i)
    plt.imshow(x_test_noisy[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
plt.show()
```



```
input_layer = keras.Input(shape=(28, 28, 1))

x = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(input_layer)
x = layers.MaxPooling2D((2, 2), padding='same')(x)
x = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(x)
encoded = layers.MaxPooling2D((2, 2), padding='same')(x)


x = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(encoded)
x = layers.UpSampling2D((2, 2))(x)
x = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(x)
x = layers.UpSampling2D((2, 2))(x)
decoded = layers.Conv2D(1, (3, 3), activation='sigmoid', padding='same')(x)

autoencoder = keras.Model(input_layer, decoded)
autoencoder.summary()
```

```
Model: "model_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 input_3 (InputLayer)        [(None, 28, 28, 1)]       0         
                                                                 
 conv2d (Conv2D)             (None, 28, 28, 32)        320       
                                                                 
 max_pooling2d (MaxPooling2D  (None, 14, 14, 32)       0         
 )                                                               
                                                                 
 conv2d_1 (Conv2D)           (None, 14, 14, 32)        9248      
                                                                 
 max_pooling2d_1 (MaxPooling  (None, 7, 7, 32)         0         
 2D)                                                             
                                                                 
 conv2d_2 (Conv2D)           (None, 7, 7, 32)          9248      
                                                                 
 up_sampling2d (UpSampling2D  (None, 14, 14, 32)       0         
 )                                                               
```

```
 conv2d_3 (Conv2D)           (None, 14, 14, 32)        9248

 up_sampling2d_1 (UpSampling  (None, 28, 28, 32)       0
 2D)

 conv2d_4 (Conv2D)           (None, 28, 28, 1)         289

=================================================================
Total params: 28,353
Trainable params: 28,353
Non-trainable params: 0
_________________________________________________________________
```

Training the noise model and plotting

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

autoencoder.fit(x_train_noisy, x_train,
                epochs=30,
                batch_size=2048,
                shuffle=True,
                validation_data=(x_validation_noisy, x_validation))
```

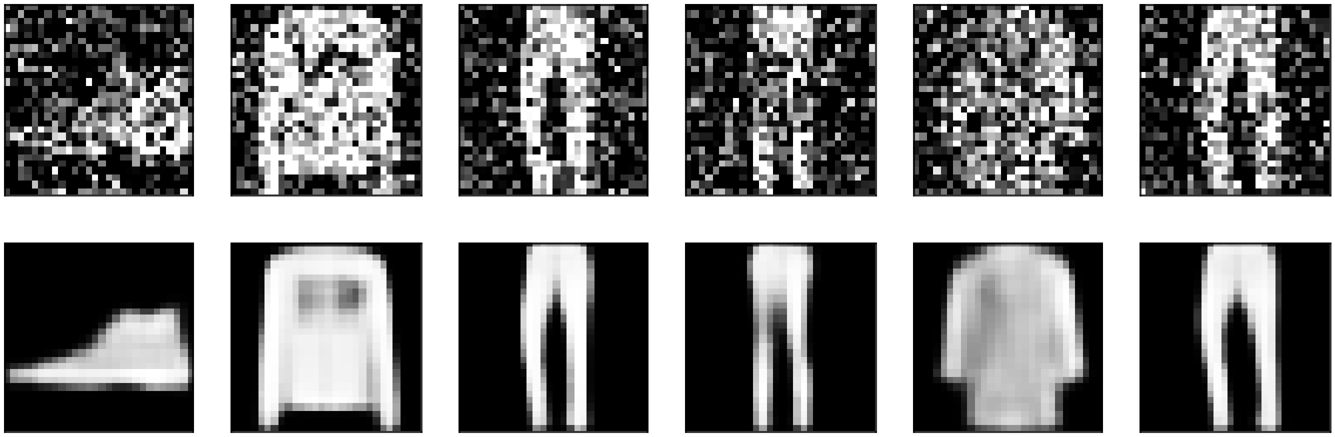
```
24/24 [==============================] - 129s 5s/step - loss: 0.3526 - val_loss:
Epoch 3/30
24/24 [==============================] - 130s 5s/step - loss: 0.3277 - val_loss:
Epoch 4/30
24/24 [==============================] - 128s 5s/step - loss: 0.3186 - val_loss:
Epoch 5/30
24/24 [==============================] - 130s 5s/step - loss: 0.3137 - val_loss:
Epoch 6/30
24/24 [==============================] - 128s 5s/step - loss: 0.3104 - val_loss:
Epoch 7/30
24/24 [==============================] - 130s 5s/step - loss: 0.3080 - val_loss:
Epoch 8/30
24/24 [==============================] - 129s 5s/step - loss: 0.3062 - val_loss:
Epoch 9/30
24/24 [==============================] - 128s 5s/step - loss: 0.3047 - val_loss:
Epoch 10/30
24/24 [==============================] - 130s 5s/step - loss: 0.3034 - val_loss:
Epoch 11/30
24/24 [==============================] - 128s 5s/step - loss: 0.3024 - val_loss:
Epoch 12/30
24/24 [==============================] - 129s 5s/step - loss: 0.3014 - val_loss:
Epoch 13/30
24/24 [==============================] - 128s 5s/step - loss: 0.3007 - val_loss:
Epoch 14/30
24/24 [==============================] - 129s 5s/step - loss: 0.2997 - val_loss:
Epoch 15/30
24/24 [==============================] - 129s 5s/step - loss: 0.2991 - val_loss:
Epoch 16/30
24/24 [==============================] - 133s 5s/step - loss: 0.2981 - val_loss:
Epoch 17/30
```

```
24/24 [==============================] - 130s 5s/step - loss: 0.2976 - val_loss:
Epoch 18/30
24/24 [==============================] - 138s 6s/step - loss: 0.2968 - val_loss:
Epoch 19/30
24/24 [==============================] - 132s 6s/step - loss: 0.2973 - val_loss:
Epoch 20/30
24/24 [==============================] - 133s 6s/step - loss: 0.2963 - val_loss:
Epoch 21/30
24/24 [==============================] - 133s 6s/step - loss: 0.2955 - val_loss:
Epoch 22/30
24/24 [==============================] - 131s 5s/step - loss: 0.2951 - val_loss:
Epoch 23/30
24/24 [==============================] - 132s 6s/step - loss: 0.2947 - val_loss:
Epoch 24/30
24/24 [==============================] - 131s 5s/step - loss: 0.2944 - val_loss:
Epoch 25/30
24/24 [==============================] - 133s 6s/step - loss: 0.2941 - val_loss:
Epoch 26/30
24/24 [==============================] - 131s 5s/step - loss: 0.2940 - val_loss:
Epoch 27/30
24/24 [==============================] - 132s 6s/step - loss: 0.2935 - val_loss:
Epoch 28/30
24/24 [==============================] - 132s 6s/step - loss: 0.2934 - val_loss:
Epoch 29/30
24/24 [==============================] - 130s 5s/step - loss: 0.2931 - val_loss:
Epoch 30/30
24/24 [==============================] - 132s 6s/step - loss: 0.2928 - val_loss:
<keras.callbacks.History at 0x7ffa53a9f990>
```

Plotting the reconstructed images

```
predictions = 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_noisy[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(predictions[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
plt.show()
```

```
313/313 [==============================] - 7s 21ms/step
```



✓ 8s completed at 7:56 PM