Fashion_MNIST

June 21, 2020

[1]:
```
import pandas as pd
import numpy as np
import tensorflow as tf
from tensorflow import keras
import matplotlib.pyplot as plt
```

[44]:
```
fashion_mnist = keras.datasets.fashion_mnist
(train_images, train_labels), (test_images, test_labels) = fashion_mnist.
 ↪load_data()
```

[45]:
```
print(train_images.shape)
```

```
(60000, 28, 28)
```

[46]:
```
train_images=train_images.reshape(60000, 28, 28, 1)
train_images=train_images / 255.0
test_images = test_images.reshape(10000, 28, 28, 1)
test_images=test_images/255.0
print(train_images.shape, test_images.shape)
```

```
(60000, 28, 28, 1) (10000, 28, 28, 1)
```

[47]:
```
from tensorflow.keras import layers
from tensorflow.keras.models import load_model
```

[64]:
```
model = keras.Sequential([
    tf.keras.layers.Conv2D(64, (3,3), activation='relu', input_shape=(28, 28,␣
 ↪1)),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    layers.Flatten(),
    layers.Dense(128, activation = 'relu'),
    layers.Dense(10, activation = 'softmax')
])
```

[65]:
```
class myCallback(tf.keras.callbacks.Callback):
    def on_epoch_end(self, epoch, logs={}):
        if logs.get('accuracy') > 0.98:
            print("\nReached 98% accuracy so cancelling training!")
            self.model.stop_training = True
```

[66]:
```
callbacks = myCallback()
```

[67]:
```
model.compile(optimizer = 'adam', loss = 'sparse_categorical_crossentropy',␣
 ↪metrics = ['accuracy'])
```

[68]:
```
model.summary()
```

```
Model: "sequential_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_10 (Conv2D)           (None, 26, 26, 64)        640
_________________________________________________________________
max_pooling2d_10 (MaxPooling (None, 13, 13, 64)        0
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 11, 11, 64)        36928
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 5, 5, 64)          0
_________________________________________________________________
flatten_6 (Flatten)          (None, 1600)              0
_________________________________________________________________
dense_12 (Dense)             (None, 128)               204928
_________________________________________________________________
dense_13 (Dense)             (None, 10)                1290
=================================================================
Total params: 243,786
Trainable params: 243,786
Non-trainable params: 0
_________________________________________________________________
```

[69]:
```
history = model.fit(train_images, train_labels, epochs = 100, verbose = 1,␣
 ↪callbacks = [callbacks])
```

```
Epoch 1/100
1875/1875 [==============================] - 73s 39ms/step - loss: 0.4300 -
accuracy: 0.8427
Epoch 2/100
1875/1875 [==============================] - 75s 40ms/step - loss: 0.2862 -
accuracy: 0.8956
Epoch 3/100
1875/1875 [==============================] - 78s 42ms/step - loss: 0.2419 -
```

```
accuracy: 0.9103
Epoch 4/100
1875/1875 [==============================] - 73s 39ms/step - loss: 0.2077 -
accuracy: 0.9211
Epoch 5/100
1875/1875 [==============================] - 70s 37ms/step - loss: 0.1835 -
accuracy: 0.9314
Epoch 6/100
1875/1875 [==============================] - 71s 38ms/step - loss: 0.1604 -
accuracy: 0.9389
Epoch 7/100
1875/1875 [==============================] - 76s 40ms/step - loss: 0.1402 -
accuracy: 0.9472
Epoch 8/100
1875/1875 [==============================] - 72s 39ms/step - loss: 0.1229 -
accuracy: 0.9526
Epoch 9/100
1875/1875 [==============================] - 70s 38ms/step - loss: 0.1071 -
accuracy: 0.9600
Epoch 10/100
1875/1875 [==============================] - 73s 39ms/step - loss: 0.0930 -
accuracy: 0.9646
Epoch 11/100
1875/1875 [==============================] - 73s 39ms/step - loss: 0.0806 -
accuracy: 0.9701
Epoch 12/100
1875/1875 [==============================] - 71s 38ms/step - loss: 0.0728 -
accuracy: 0.9724
Epoch 13/100
1875/1875 [==============================] - 72s 39ms/step - loss: 0.0631 -
accuracy: 0.9759
Epoch 14/100
1875/1875 [==============================] - 70s 37ms/step - loss: 0.0568 -
accuracy: 0.9787
Epoch 15/100
1875/1875 [==============================] - ETA: 0s - loss: 0.0493 - accuracy:
0.9819
Reached 98% accuracy so cancelling training!
1875/1875 [==============================] - 74s 40ms/step - loss: 0.0493 -
accuracy: 0.9819
```

[70]:
```
model.save("clothing_model.h5")
print("Saved Model to Disk")
```

```
Saved Model to Disk
```

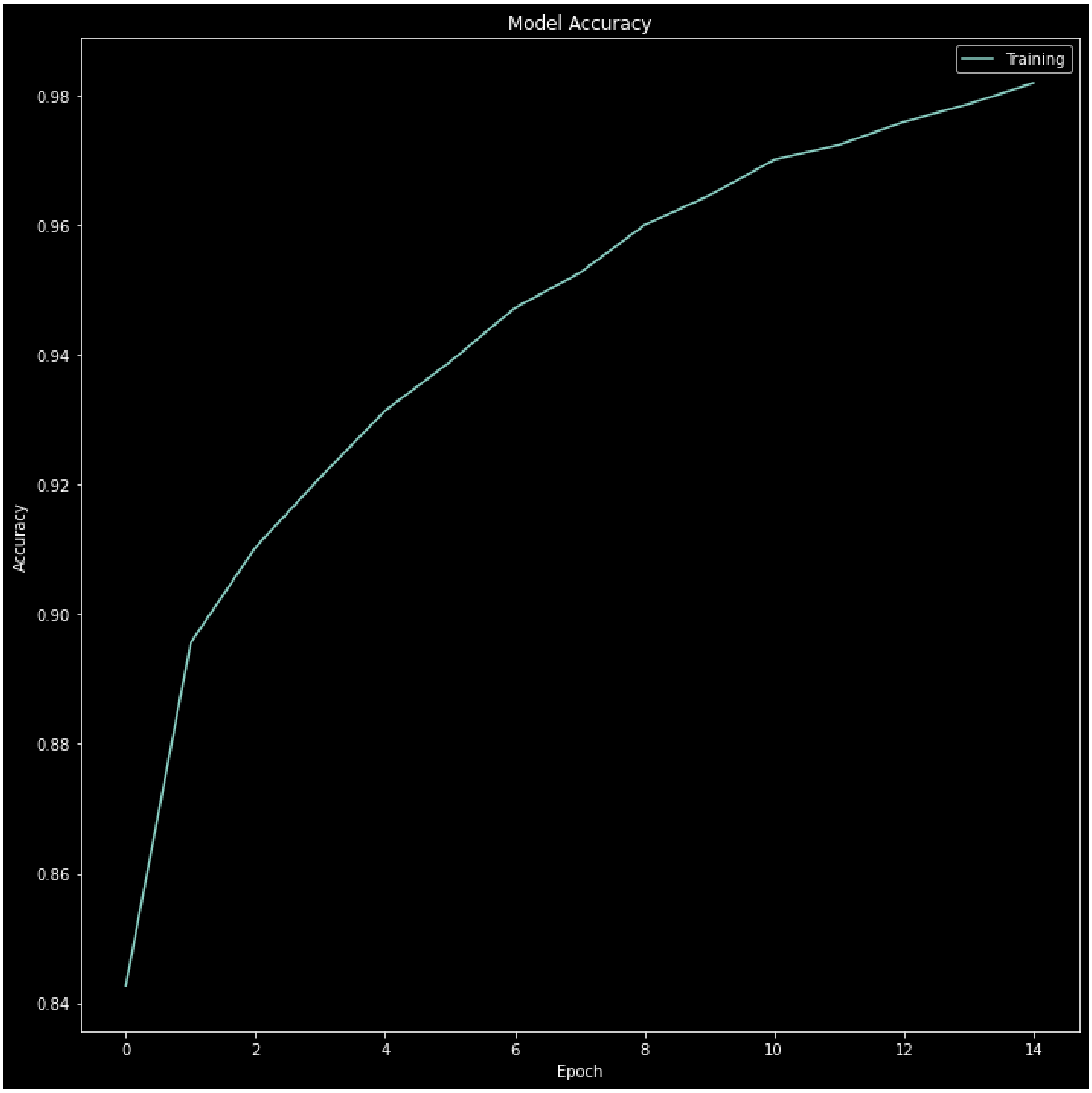
[71]:
```
model = load_model("clothing_model.h5")
y_hat = model.predict(test_images)
```

[72]:
```
model.evaluate(test_images, test_labels)
```

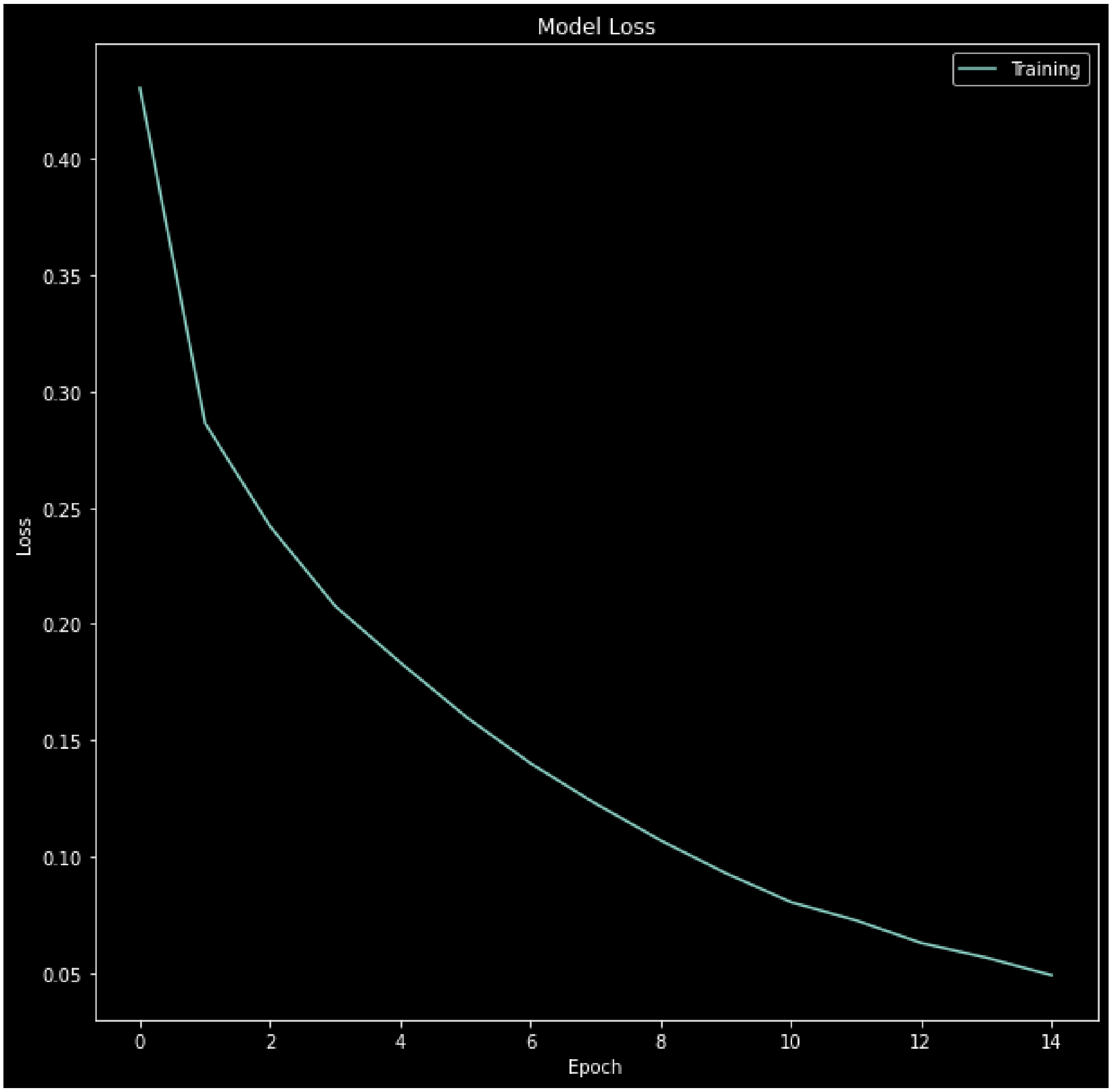
```
313/313 [==============================] - 3s 10ms/step - loss: 0.4329 -
accuracy: 0.9074
```

[72]:
```
[0.432949960231781, 0.9074000120162964]
```

[73]:
```
plt.figure(figsize=(10,10))
plt.style.use('dark_background')
plt.plot(history.history['accuracy'])
# plt.plot(history.history['val_accuracy'])
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Training', 'Testing'])
plt.tight_layout()
plt.show()
```



[74]:
```
plt.figure(figsize=(10,10))
plt.style.use('dark_background')
plt.plot(history.history['loss'])
# plt.plot(history.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Training', 'Testing'])
plt.show()
```


Model Loss
Training
Loss
Epoch
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0
2
4
6
8
10
12
14

[ ]: