Homework-1

Name: Hanumantha Rao Vakkalanka

Email: hvakkala@uncc.edu
Student ID: 801333188

Course: ECGR 5106 Real Time ML

Lab Number: Spring 2023

Problem: 1-A

[ ]:
```python
import tensorflow as tf
import matplotlib.pyplot as plt
```

[ ]:
```python
(x_training, y_training), (x_testing, y_testing) = tf.keras.datasets.
 ↪fashion_mnist.load_data()
```

[ ]:
```python
x_training = x_training.reshape(-1, 784) / 255.0
x_testing = x_testing.reshape(-1, 784) / 255.0
```

[ ]:
```python
model = tf.keras.models.Sequential()
model.add(tf.keras.layers.Dense(512, activation='relu', input_shape=(784,)))
model.add(tf.keras.layers.Dense(256, activation='relu'))
model.add(tf.keras.layers.Dense(128, activation='relu'))
model.add(tf.keras.layers.Dense(10, activation='softmax'))
```

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

The progress of the model during training and validation is recorded in a history object and displayed using matplotlib, through graphs of the training and validation loss and accuracy. If the validation loss and accuracy are significantly lower than the training loss and accuracy, this could indicate overfitting.

[ ]:
```python
history = model.fit(x_training, y_training, epochs=20,␣
 ↪validation_data=(x_testing, y_testing))
```

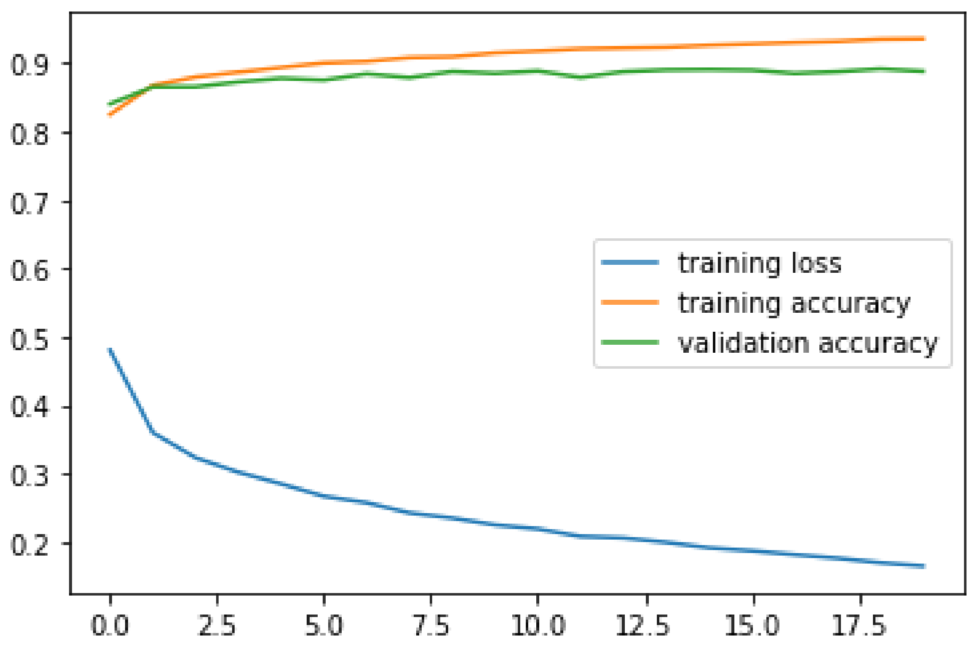
```
Epoch 1/20
1875/1875 [==============================] - 20s 10ms/step - loss: 0.4760 -
```

```
accuracy: 0.8273 - val_loss: 0.4171 - val_accuracy: 0.8421
Epoch 2/20
1875/1875 [==============================] - 21s 11ms/step - loss: 0.3626 -
accuracy: 0.8659 - val_loss: 0.3719 - val_accuracy: 0.8709
Epoch 3/20
1875/1875 [==============================] - 22s 11ms/step - loss: 0.3278 -
accuracy: 0.8790 - val_loss: 0.3651 - val_accuracy: 0.8685
Epoch 4/20
1875/1875 [==============================] - 16s 8ms/step - loss: 0.3012 -
accuracy: 0.8885 - val_loss: 0.3444 - val_accuracy: 0.8738
Epoch 5/20
1875/1875 [==============================] - 18s 10ms/step - loss: 0.2836 -
accuracy: 0.8946 - val_loss: 0.3468 - val_accuracy: 0.8767
Epoch 6/20
1875/1875 [==============================] - 18s 9ms/step - loss: 0.2692 -
accuracy: 0.8996 - val_loss: 0.3478 - val_accuracy: 0.8820
Epoch 7/20
1875/1875 [==============================] - 25s 13ms/step - loss: 0.2560 -
accuracy: 0.9036 - val_loss: 0.3273 - val_accuracy: 0.8845
Epoch 8/20
1875/1875 [==============================] - 17s 9ms/step - loss: 0.2466 -
accuracy: 0.9073 - val_loss: 0.3383 - val_accuracy: 0.8808
Epoch 9/20
1875/1875 [==============================] - 15s 8ms/step - loss: 0.2352 -
accuracy: 0.9107 - val_loss: 0.3508 - val_accuracy: 0.8844
Epoch 10/20
1875/1875 [==============================] - 15s 8ms/step - loss: 0.2264 -
accuracy: 0.9146 - val_loss: 0.3319 - val_accuracy: 0.8846
Epoch 11/20
1875/1875 [==============================] - 16s 8ms/step - loss: 0.2216 -
accuracy: 0.9158 - val_loss: 0.3350 - val_accuracy: 0.8867
Epoch 12/20
1875/1875 [==============================] - 16s 8ms/step - loss: 0.2125 -
accuracy: 0.9190 - val_loss: 0.3436 - val_accuracy: 0.8872
Epoch 13/20
1875/1875 [==============================] - 17s 9ms/step - loss: 0.2021 -
accuracy: 0.9219 - val_loss: 0.3573 - val_accuracy: 0.8876
Epoch 14/20
1875/1875 [==============================] - 15s 8ms/step - loss: 0.1978 -
accuracy: 0.9235 - val_loss: 0.3578 - val_accuracy: 0.8881
Epoch 15/20
1875/1875 [==============================] - 17s 9ms/step - loss: 0.1922 -
accuracy: 0.9269 - val_loss: 0.3582 - val_accuracy: 0.8814
Epoch 16/20
1875/1875 [==============================] - 15s 8ms/step - loss: 0.1862 -
accuracy: 0.9289 - val_loss: 0.3588 - val_accuracy: 0.8918
Epoch 17/20
1875/1875 [==============================] - 18s 9ms/step - loss: 0.1835 -
```

```
accuracy: 0.9286 - val_loss: 0.3850 - val_accuracy: 0.8920
Epoch 18/20
1875/1875 [==============================] - 15s 8ms/step - loss: 0.1755 -
accuracy: 0.9333 - val_loss: 0.3833 - val_accuracy: 0.8947
Epoch 19/20
1875/1875 [==============================] - 18s 9ms/step - loss: 0.1726 -
accuracy: 0.9340 - val_loss: 0.3766 - val_accuracy: 0.8921
Epoch 20/20
1875/1875 [==============================] - 19s 10ms/step - loss: 0.1672 -
accuracy: 0.9365 - val_loss: 0.4240 - val_accuracy: 0.8859
```

[ ]:
```
plt.plot(history.history['loss'], label='training loss')
plt.plot(history.history['accuracy'], label='training accuracy')
plt.plot(history.history['val_accuracy'], label='validation accuracy')
plt.legend()
plt.show()

model.save('fashion_mnist_mlp.h5')
```



Problem 1-B

[ ]:
```
import tensorflow as tf
import matplotlib.pyplot as plt
```

[ ]:
```
(x_training, y_training), (x_testing, y_testing) = tf.keras.datasets.
 ↪fashion_mnist.load_data()
```

[ ]:
```
x_training = x_training.reshape(-1, 784) / 255.0
x_testing = x_testing.reshape(-1, 784) / 255.0

model = tf.keras.models.Sequential([
tf.keras.layers.Dense(512, activation='relu', kernel_regularizer=tf.keras.
 ↪regularizers.l2(0.01), input_shape=(784,)),
tf.keras.layers.Dense(256, activation='relu', kernel_regularizer=tf.keras.
 ↪regularizers.l2(0.01)),
tf.keras.layers.Dense(128, activation='relu', kernel_regularizer=tf.keras.
 ↪regularizers.l2(0.01)),
tf.keras.layers.Dense(10, activation='softmax')
])
```

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

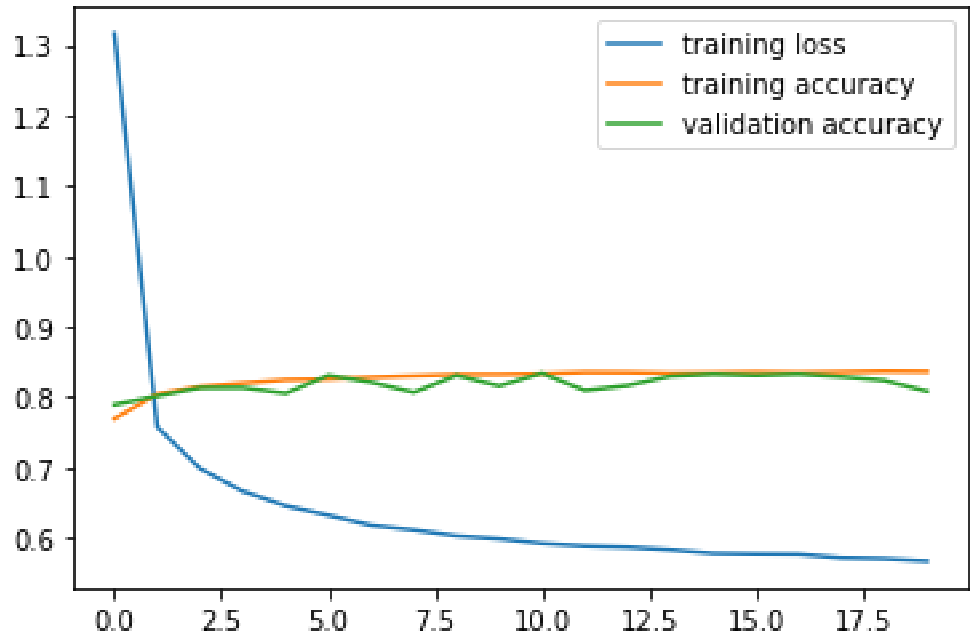
The dense layers in the model have been augmented with L2 weight decay regularization, with a coefficient of 0.01. This coefficient can be altered to observe its effect on the model's performance.

[ ]:
```
history = model.fit(x_training, y_training, epochs=20,␣
 ↪validation_data=(x_testing, y_testing))

plt.plot(history.history['loss'], label='training loss')
plt.plot(history.history['accuracy'], label='training accuracy')
plt.plot(history.history['val_accuracy'], label='validation accuracy')
plt.legend()
plt.show()
```

```
Epoch 1/20
1875/1875 [==============================] - 22s 11ms/step - loss: 1.3166 -
accuracy: 0.7697 - val_loss: 0.8296 - val_accuracy: 0.7895
Epoch 2/20
1875/1875 [==============================] - 18s 10ms/step - loss: 0.7580 -
accuracy: 0.8046 - val_loss: 0.7342 - val_accuracy: 0.8018
Epoch 3/20
1875/1875 [==============================] - 19s 10ms/step - loss: 0.6989 -
accuracy: 0.8152 - val_loss: 0.7022 - val_accuracy: 0.8135
Epoch 4/20
1875/1875 [==============================] - 20s 10ms/step - loss: 0.6666 -
accuracy: 0.8202 - val_loss: 0.6916 - val_accuracy: 0.8137
Epoch 5/20
1875/1875 [==============================] - 20s 10ms/step - loss: 0.6458 -
accuracy: 0.8248 - val_loss: 0.6873 - val_accuracy: 0.8056
Epoch 6/20
1875/1875 [==============================] - 18s 10ms/step - loss: 0.6323 -
accuracy: 0.8259 - val_loss: 0.6206 - val_accuracy: 0.8313
Epoch 7/20
```

```
1875/1875 [==============================] - 21s 11ms/step - loss: 0.6181 -
accuracy: 0.8286 - val_loss: 0.6410 - val_accuracy: 0.8215
Epoch 8/20
1875/1875 [==============================] - 19s 10ms/step - loss: 0.6114 -
accuracy: 0.8305 - val_loss: 0.6503 - val_accuracy: 0.8070
Epoch 9/20
1875/1875 [==============================] - 21s 11ms/step - loss: 0.6030 -
accuracy: 0.8324 - val_loss: 0.6121 - val_accuracy: 0.8312
Epoch 10/20
1875/1875 [==============================] - 18s 10ms/step - loss: 0.5988 -
accuracy: 0.8324 - val_loss: 0.6379 - val_accuracy: 0.8162
Epoch 11/20
1875/1875 [==============================] - 20s 11ms/step - loss: 0.5923 -
accuracy: 0.8335 - val_loss: 0.6055 - val_accuracy: 0.8343
Epoch 12/20
1875/1875 [==============================] - 18s 10ms/step - loss: 0.5886 -
accuracy: 0.8355 - val_loss: 0.6439 - val_accuracy: 0.8097
Epoch 13/20
1875/1875 [==============================] - 22s 12ms/step - loss: 0.5869 -
accuracy: 0.8353 - val_loss: 0.6204 - val_accuracy: 0.8169
Epoch 14/20
1875/1875 [==============================] - 20s 11ms/step - loss: 0.5832 -
accuracy: 0.8342 - val_loss: 0.5928 - val_accuracy: 0.8301
Epoch 15/20
1875/1875 [==============================] - 20s 11ms/step - loss: 0.5781 -
accuracy: 0.8347 - val_loss: 0.5883 - val_accuracy: 0.8329
Epoch 16/20
1875/1875 [==============================] - 19s 10ms/step - loss: 0.5772 -
accuracy: 0.8356 - val_loss: 0.5909 - val_accuracy: 0.8312
Epoch 17/20
1875/1875 [==============================] - 20s 11ms/step - loss: 0.5769 -
accuracy: 0.8349 - val_loss: 0.5900 - val_accuracy: 0.8330
Epoch 18/20
1875/1875 [==============================] - 19s 10ms/step - loss: 0.5717 -
accuracy: 0.8358 - val_loss: 0.5912 - val_accuracy: 0.8293
Epoch 19/20
1875/1875 [==============================] - 19s 10ms/step - loss: 0.5705 -
accuracy: 0.8370 - val_loss: 0.5984 - val_accuracy: 0.8241
Epoch 20/20
1875/1875 [==============================] - 18s 10ms/step - loss: 0.5671 -
accuracy: 0.8365 - val_loss: 0.6459 - val_accuracy: 0.8085
```



[ ]:
```
model.save('fashion_mnist_mlp_with_l2_regularization.h5')
```

Problem 1-C

[ ]:
```
import tensorflow as tf
import matplotlib.pyplot as plt
```

[ ]:
```
(x_training, y_training), (x_testing, y_testing) = tf.keras.datasets.
 ↪fashion_mnist.load_data()
```

[ ]:
```
x_training = x_training.reshape(-1, 784) / 255.0
x_testing = x_testing.reshape(-1, 784) / 255.0
```

[ ]:
```
model = tf.keras.models.Sequential()
model.add(tf.keras.layers.Dense(128, activation='relu', input_shape=(784,)))
model.add(tf.keras.layers.Dropout(0.3))
model.add(tf.keras.layers.Dense(64, activation='relu'))
model.add(tf.keras.layers.Dropout(0.3))
model.add(tf.keras.layers.Dense(32, activation='relu'))
model.add(tf.keras.layers.Dropout(0.3))
model.add(tf.keras.layers.Dense(10, activation='softmax'))
```

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

The dropout can decrease overfitting by randomly excluding neurons during the training process, thus improving its ability to generalize to new data. The decision between using dropout or weight penalties depends on the specific task at hand and the preferred balance between model complexity and overfitting.

```
[ ]: history = model.fit(x_training, y_training, epochs=20,
      validation_data=(x_testing, y_testing))
```

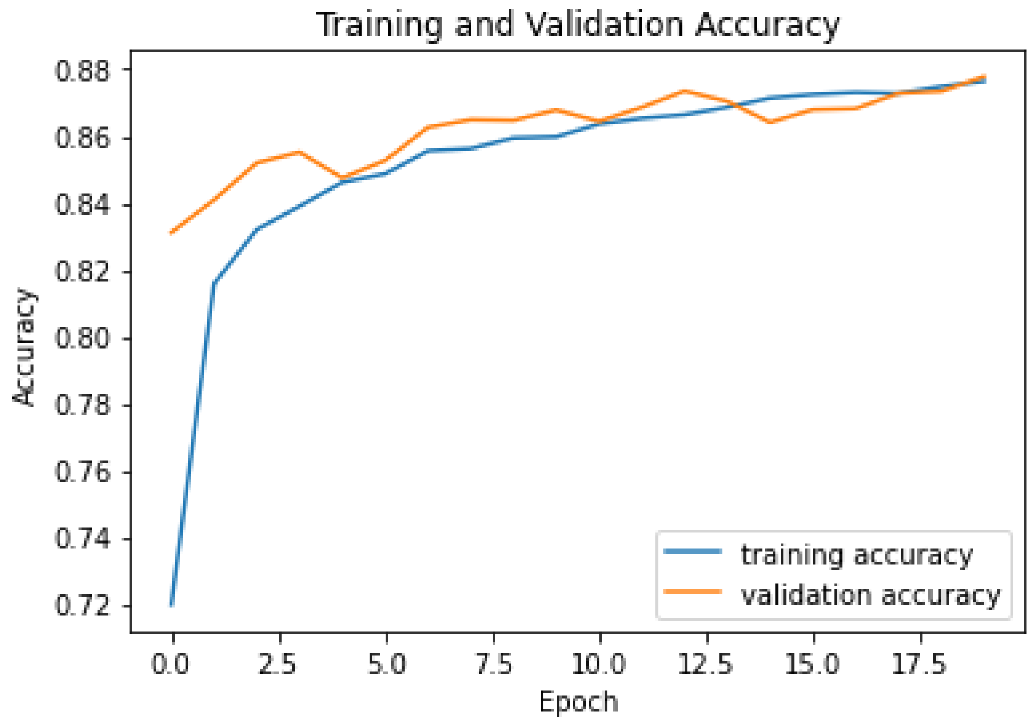
```
Epoch 1/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.7877 -
accuracy: 0.7197 - val_loss: 0.4692 - val_accuracy: 0.8311
Epoch 2/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.5383 -
accuracy: 0.8158 - val_loss: 0.4373 - val_accuracy: 0.8409
Epoch 3/20
1875/1875 [==============================] - 8s 5ms/step - loss: 0.4869 -
accuracy: 0.8320 - val_loss: 0.4145 - val_accuracy: 0.8520
Epoch 4/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.4670 -
accuracy: 0.8390 - val_loss: 0.4001 - val_accuracy: 0.8552
Epoch 5/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.4419 -
accuracy: 0.8462 - val_loss: 0.4147 - val_accuracy: 0.8475
Epoch 6/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.4337 -
accuracy: 0.8487 - val_loss: 0.4038 - val_accuracy: 0.8527
Epoch 7/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.4184 -
accuracy: 0.8556 - val_loss: 0.3803 - val_accuracy: 0.8626
Epoch 8/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.4136 -
accuracy: 0.8563 - val_loss: 0.3787 - val_accuracy: 0.8649
Epoch 9/20
1875/1875 [==============================] - 8s 5ms/step - loss: 0.3980 -
accuracy: 0.8594 - val_loss: 0.3808 - val_accuracy: 0.8647
Epoch 10/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3966 -
accuracy: 0.8598 - val_loss: 0.3678 - val_accuracy: 0.8678
Epoch 11/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3886 -
accuracy: 0.8636 - val_loss: 0.3752 - val_accuracy: 0.8644
Epoch 12/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3864 -
accuracy: 0.8653 - val_loss: 0.3672 - val_accuracy: 0.8686
Epoch 13/20
1875/1875 [==============================] - 10s 6ms/step - loss: 0.3759 -
accuracy: 0.8664 - val_loss: 0.3559 - val_accuracy: 0.8735
Epoch 14/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3748 -
```

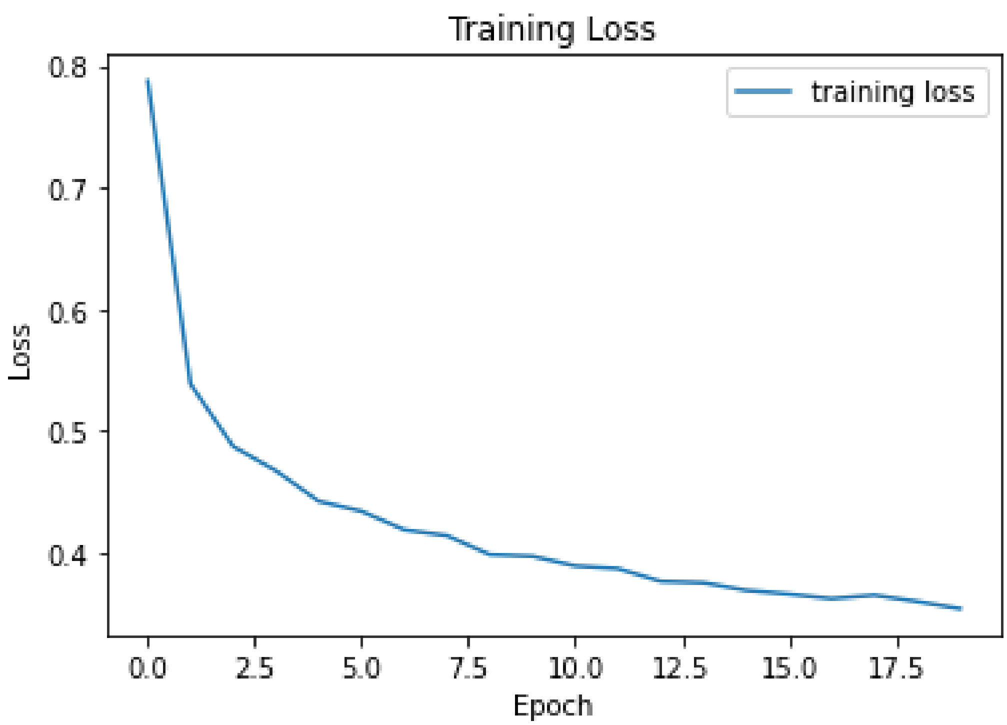
```
accuracy: 0.8686 - val_loss: 0.3591 - val_accuracy: 0.8704
Epoch 15/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3685 -
accuracy: 0.8714 - val_loss: 0.3813 - val_accuracy: 0.8641
Epoch 16/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.3654 -
accuracy: 0.8725 - val_loss: 0.3671 - val_accuracy: 0.8679
Epoch 17/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3617 -
accuracy: 0.8731 - val_loss: 0.3648 - val_accuracy: 0.8682
Epoch 18/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3646 -
accuracy: 0.8728 - val_loss: 0.3548 - val_accuracy: 0.8728
Epoch 19/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3593 -
accuracy: 0.8748 - val_loss: 0.3498 - val_accuracy: 0.8734
Epoch 20/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3534 -
accuracy: 0.8764 - val_loss: 0.3505 - val_accuracy: 0.8778
```

[ ]:

```python
plt.plot(history.history['accuracy'], label='training accuracy')
plt.plot(history.history['val_accuracy'], label='validation accuracy')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()

plt.plot(history.history['loss'], label='training loss')
plt.title('Training Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.show()
```


Training and Validation Accuracy
Accuracy
0.88
0.86
0.84
0.82
0.80
0.78
0.76
0.74
0.72
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Epoch
training accuracy
validation accuracy


Training Loss
Loss
0.8
0.7
0.6
0.5
0.4
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Epoch
training loss

Problem 1-D

```
[ ]: import tensorflow as tf
     import matplotlib.pyplot as plt
```

```
[ ]: (fashion_mnist_training_data, fashion_mnist_training_labels),␣
      ↪(fashion_mnist_testing_data, fashion_mnist_testing_labels) = tf.keras.
      ↪datasets.fashion_mnist.load_data()
```

```
[ ]: scaled_training_data = fashion_mnist_training_data.reshape(-1, 784) / 255.0
     scaled_testing_data = fashion_mnist_testing_data.reshape(-1, 784) / 255.0
```

```
[ ]: model = tf.keras.models.Sequential([
     tf.keras.layers.Dense(128, activation='relu', input_shape=(784,)),
     tf.keras.layers.Dense(64, activation='relu'),
     tf.keras.layers.Dense(10, activation='softmax')
     ])
```

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

```
[ ]: model.load_weights("model_weights.h5")
```

This will retrieve the pre-trained weights from the file "model_weights.h5" and evaluate the accuracy of the model on the test data. It will then proceed to train the model for 20 cycles, displaying a plot that illustrates the accuracy of the model on both the training and validation sets as the training progresses over each epoch.

```
[ ]: loss, accuracy = model.evaluate(scaled_testing_data,␣
      ↪fashion_mnist_testing_labels, verbose=False)
     print("Test Accuracy: {:.4f}".format(accuracy))
```

```
Test Accuracy: 0.0911
```

```
[ ]: history = model.fit(scaled_training_data, fashion_mnist_training_labels,␣
      ↪epochs=20,
     validation_data=(scaled_testing_data, fashion_mnist_testing_labels))
```

```
Epoch 1/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.4953 -
accuracy: 0.8225 - val_loss: 0.4120 - val_accuracy: 0.8520
Epoch 2/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3665 -
accuracy: 0.8666 - val_loss: 0.4060 - val_accuracy: 0.8589
Epoch 3/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3316 -
accuracy: 0.8778 - val_loss: 0.3996 - val_accuracy: 0.8558
Epoch 4/20
```

```
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3077 -
accuracy: 0.8869 - val_loss: 0.3729 - val_accuracy: 0.8665
Epoch 5/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2918 -
accuracy: 0.8909 - val_loss: 0.3558 - val_accuracy: 0.8710
Epoch 6/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.2791 -
accuracy: 0.8965 - val_loss: 0.3340 - val_accuracy: 0.8817
Epoch 7/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2690 -
accuracy: 0.8995 - val_loss: 0.3463 - val_accuracy: 0.8741
Epoch 8/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.2550 -
accuracy: 0.9049 - val_loss: 0.3342 - val_accuracy: 0.8801
Epoch 9/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2469 -
accuracy: 0.9059 - val_loss: 0.3367 - val_accuracy: 0.8835
Epoch 10/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.2370 -
accuracy: 0.9108 - val_loss: 0.3296 - val_accuracy: 0.8840
Epoch 11/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.2310 -
accuracy: 0.9132 - val_loss: 0.3468 - val_accuracy: 0.8792
Epoch 12/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2227 -
accuracy: 0.9153 - val_loss: 0.3581 - val_accuracy: 0.8779
Epoch 13/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2175 -
accuracy: 0.9183 - val_loss: 0.3448 - val_accuracy: 0.8818
Epoch 14/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.2095 -
accuracy: 0.9201 - val_loss: 0.3400 - val_accuracy: 0.8882
Epoch 15/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.2026 -
accuracy: 0.9231 - val_loss: 0.3509 - val_accuracy: 0.8867
Epoch 16/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.1973 -
accuracy: 0.9243 - val_loss: 0.3477 - val_accuracy: 0.8899
Epoch 17/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.1958 -
accuracy: 0.9256 - val_loss: 0.3390 - val_accuracy: 0.8898
Epoch 18/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.1901 -
accuracy: 0.9271 - val_loss: 0.3474 - val_accuracy: 0.8853
Epoch 19/20
1875/1875 [==============================] - 7s 3ms/step - loss: 0.1839 -
accuracy: 0.9305 - val_loss: 0.3690 - val_accuracy: 0.8830
Epoch 20/20
```

```
1875/1875 [==============================] - 7s 4ms/step - loss: 0.1792 -
accuracy: 0.9313 - val_loss: 0.3643 - val_accuracy: 0.8909
```

The Decay model seems to cause a 20-second increase in training time on an, while the Drop model only adds an additional 10 seconds. The model that uses pre-training with weight decay and dropout has a comparable training time to the Decay model. Nevertheless, the basic model attains stability faster, needing fewer training cycles, whereas the Decay and Drop models reach stability at around 15 epochs. Although, there is only a slight difference in their generalization capability.

[ ]:
```python
plt.plot(history.history['accuracy'], label='training accuracy')
plt.plot(history.history['val_accuracy'], label = 'validation accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend(loc='lower right')
plt.show()
```