Homework 7 **NSC 3270 / 6270**
Due Thu Nov 11 at start of class **Fall 2021**
20 points

This assignment asks you to create multi-layer networks that learn to classify images of clothing. Fashion-MNIST is a dataset of clothing images consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes.

`Homework7.ipynb` supplies code to read in the dataset and display several examples. Since the dataset is structurally similar to that of MNIST, you might refer to previously-shared code for classifying MNIST digits for help.

The first problem asks you to create a network that first minimizes the dimensionality of the Fashion-MNIST dataset using an auto-associator and then use that reduced dimensionality as the input to a multi-layer dense network that learns to classify the images. The second problem asks you to create a convolutional network that learns to classify the original images (without first passing them the auto-associator).

Q1a (5 points). Create an autoassociator network that produces a lower-dimensional representation of the Fashion-MNIST images on its hidden layer. It will use a "flattened" (one-dimensional) version of these images (like we saw in earlier networks trained with MNIST digits).

Train the autoassociator on 1/2 of the Fashion-MNIST training images. Make sure you use the appropriate activation functions on the output layer and hidden layer for the network to approximate PCA (as discussed in class).

It is up to you to determine the dimensionality (number of hidden units) of the autoassociator as well as the number of training epochs. Make sure you use examine `val_loss` to avoid over-fitting. In a markdown cell, justify the number of hidden units (the reduced dimensionality) by describing what you did to determine the number you selected. If the dimensionality is too small, the classification network (Q1b) will likely perform quite poorly.

Include a plot of `loss` and `val_loss`.

After you train your autoassociator, you will need to create a reduced-dimensionality versions of the remaining 1/2 of the Fashion-MNIST training set. In other words, you will need to pass this remaining 1/2 of the Fashion-MNIST images through the network and retain the activations on the hidden layer (the lower-dimensional representations). The easiest way to do this is to use the weights and biases from the trained autoassociator to calculate the activation values on the hidden layer (see class slides for how to do this).

Q1b (5 points). Now, use these lower-dimensional (PCA-like) representations of the clothing images (remaining 1/2 of the original set) as the inputs to train a densely-connected multi-layered neural network that learns, via backpropagation, to classify the images as one of the 10 types of clothing. Make sure you pick the proper activation functions on the hidden units and pick the proper activation function on the output units for a classification network; make sure you also pick the proper loss function for a classification network.

It is up to you to determine things like the number of hidden layers, the number of hidden units on each layer, the number of training epochs, and settings for training your network. Check the `loss` and `val_loss` during training to adjust the architecture and parameters of your network to achieve the best performance.

Try to achieve the highest accuracy you can during training (without over-fitting) – make sure you store `accuracy` as a metric.

Include a plot of `loss` and `val_loss`.

In a markdown cell, describe the various choices you made and justify them. This should include describing the architectures and settings you tried and discussing why you went with a different architecture and set of settings.

After the network is selected and trained (and only after that), test your network using the test images and report its performance (using `network.evaluate()`). You should never adjust your network based on the performance with the test images.

Q2 (10 points). Create a convolutional neural network that classifies the Fashion-MNIST images. Recall that the convolutional network takes images as input (not flattened images); so for Q2, you will use the original Fashion-MNIST images, not vectors of reduced dimensionality as used in Q1.

Try to find a convolutional neural network that does the best job of classifying the images. You can play with the architecture (combinations of convolutional, max pooling, and dense networks), the numbers of convolutional maps in a layer, the number of dense layers and the number of hidden units they have. You will want to inspect the val_loss to make sure you are not overtraining the network. Check the `val_loss` and `val_accuracy` during training to adjust the architecture and parameters of your network to achieve the best performance.

Try to achieve the highest accuracy you can during training (without over-fitting) – make sure you store `accuracy` as a metric.

Include a plot of loss and val_loss.

In a markdown cell, describe the various choices you made and justify them. This should include describing the architectures and settings you tried and discussing why you went with a different architecture and set of settings.

Then use `network.evaluate()` on the test images.

Unexcused late assignments will be penalized 10% for every 24 hours late, starting from the time class ends, for a maximum of two days, after which they will earn a 0.