A High Accuracy CNN for MNIST

This is based on problem #9 in Chapter 14. Yes, the notebook is there with a solution.

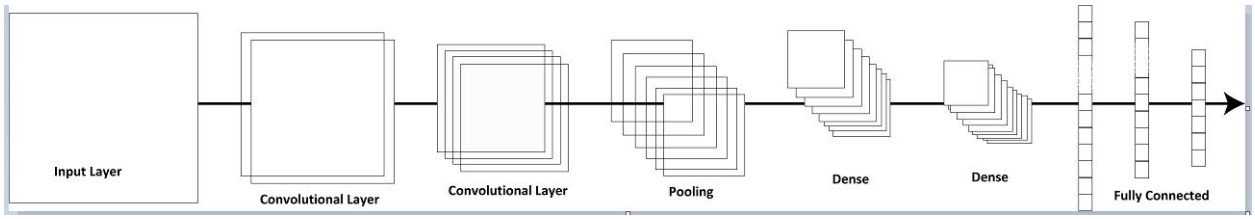
Read the chapter so that you become familiar with the approach used in Convolution Neural Nets (CNNs). Then do the following with this problem:

1. Run the Notebook and document your answer. Did you get the same answer as the default in the notebook? What was your runtime (wall-clock time)?

Notebook's answer: [0.027682604857745575, 0.992]
My answer : [0.030678831207423134, 0.9919]

Yes, I got the same accuracy 99.2% as the notebook.
My runtime was 15 minutes for this.

2. Diagram the network in a drawing program of your choice.



3. Is it possible, without making major changes to the program, to get slightly better accuracy?
Note: These models are computationally intensive, so stick with adjusting parameters here rather than adding features to the network. Things like the hidden layer size or the number of epochs are fair game. Document your results. I am not interested in your code here -- I want you to explain to me in words what experiments you ran, what the accuracy was of each experiment, and what it did to your runtime.

1. I decreased the number of epochs to 5 and the accuracy is now 99.04% and it ran in 7 minutes. This makes sense as it took half the time of the default notebook, but the accuracy was not affected a lot so it might be worth saving some time without sacrificing too much on accuracy.
2. I increased the number of epochs to 15 and the accuracy is now 99.23% so a little bit of increase and it ran in 23 minutes. This makes sense as it took 150% of the time of the default notebook as I increased the epochs 150% too, but the accuracy was not affected a lot so it might not be worth the extra time and gaining not much on accuracy.
3. With 10 epochs, adding another Dense hidden layer with activation function relu with 256 filters (keras.layers.Dense(256, activation="relu")) and then using 50% as the dropout after this layer, I got an accuracy of 99.13% with a runtime of 16 minutes.

4. With 10 epochs, adding another Dense hidden layer with activation function relu with 256 filters but without using the 50% dropout after this layer, I got an accuracy of 99.18% with a runtime of 16 minutes.
5. With 10 epochs, replacing the Dense hidden layer with 128 filters with 2 Dense hidden layers with activation function relu with 256 filters each and using the 50% dropout after each of them, I got an accuracy of 99.12 % with a runtime of 21 minutes.
6. This time, I added a convolutional layer with 128 filters along with 2 Dense hidden layers with activation function relu with 256 filters each and using the 50% dropout after each of them, I got an accuracy of 99.18% with a runtime of 48 minutes.
7. The best accuracy I could find was 99.23% on (2) in this.

4. Look back at your prior assignments where we used the MNIST digit dataset. How do other modelling methods compare to this approach? (Accuracy, etc.). Again, I'm not looking for your code here, I am looking for a well-crafted explanation!

In homework 3, the highest accuracy was with a Voting Classifier Ensemble w/ RandomForestClassifier, ExtraTreesClassifier, MLPClassifier and hard voting where I got an accuracy score of 0.9738.

So, this CNN has been the model with the highest accuracy for me.