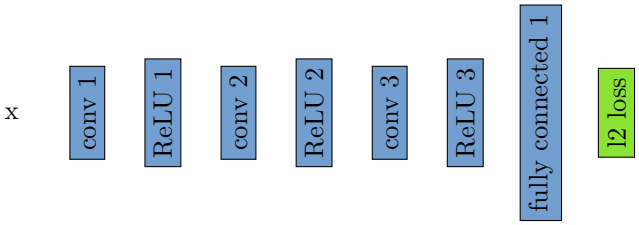
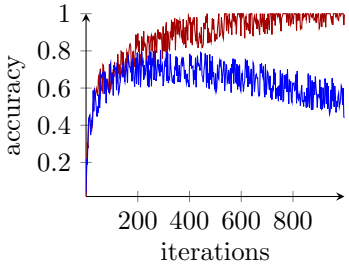
Exercise 4: Overfitting - Solution

Name: ..... UTID: .....

You haven't quite given up on solving these quizzes using a deep network. In Exercise 3, you trained a deepnet to solve these quizzes automatically: Inspired by the success of deep networks in computer vision you decide to train a deep network to answer simple quiz questions for your. The input is a screenshot x ($45 \times 45 \times 3$ image) of the particular question you want answered, the output is a scalar (positive for the "yes", negative for "no"). After creating and answering thousands of mock quizzes (as training data), you design a simple network to do all the heavy lifting for you. After many iterations and painful model tuning you arrive at the perfect architecture:





However, the model hopelessly overfits to the training data, as shown in the figure on the right. The red curve is the training accuracy, while the blue curve is validation accuracy. The longer you train, the larger the gap between the two gets.

a) Let's try to understand overfitting a bit better. Which statements about overfitting are true?

- ☐ Larger networks generally overfit more.
- ☐ Overfitting is measured as the difference between training and testing loss.
- ☒ Overfitting is measured as the difference between training and testing accuracy.
- ☐ Overfitting is generally bad and should be avoided. The best models do not overfit.
- ☒ All that matters is generalization performance on a validation set.

b) Now that we have a rough idea what overfitting does, let's look at ways to prevent it. Which methods would you use to prevent overfitting:

- ☐ Batch Normalization
- ☐ Early Stopping right before you overfit
- ☒ Early Stopping after you started to overfit
- ☐ Weight regularization
- ☒ Parameter Sharing
- ☒ Dropout

c) You see great improvement on the validation accuracy after employing all of the above techniques. However it still doesn't fully cut it. In a last ditch attempt you design your own data augmentation. Describe two different data augmentation strategies that apply to this setting.

Good ideas:

- *Random cropping*
- *Color augmentation*
- *Small random rotations*

Bad ideas:

- *Left-Right flip or large rotations (it changes the semantics of a question)*

d) You finally realize that you may have been approaching this entire problem in the wrong way. Taking a picture of a question (text) and feeding it into a convolutional neural network is wrong. Briefly describe why this is the wrong approach and how should instead structure your model.

It is too hard of a problem to recognize and parse a question using a single deep network with only a few convolutions. While the convolutional network might be good at recognizing individual characters, it is not an ideal model to reason about sentences. A much better structure would be to split the network into two, a network that recognizes characters and parses text from an image. A second network that reasons about language and questions using either a convolutional network on text, or a recurrent network (which we will see later in class).

e) If you train a convolutional network on images to answer questions about text, what do you expect the network to learn? (this is very open-ended, but try anyway)

In a best case scenario the convolutional network will learn to spot some key words (or parts of words) and memorize an answer for those key words. However, it will also likely confuse similar looking words (e.g. "word" vs "world", or "test" vs "text").