**Homework 6** **NSC 3270 / 6270**
**Due Tue Oct 26 at start of class** **Fall 2021**
**20 points**

Note that I am assigning this the Tue before Fall Break, but it is due in two weeks. You can start working on it now (though some of material needed to fully complete this assignment will be covered the Tue after Fall Break), if you want to get a head start, or you can wait to start until after Fall Break, if you want a well-deserved break.

This homework assignment builds off of Homework 5, asking you to build neural networks using Keras, but now networks with one or more hidden layers.

I am giving you no starting code this time. You are free to adapt the code used in Homework 5 (that I gave you and that you wrote for that assignment) as well as code that I gave you previously in class (on Brightspace).

You will need to refer to class slides (especially from Tue Oct 12 and Tue Oct 19) for information on how to do some of the parts outlined below.

This is going to be more explicit coding than other assignments, especially since I am giving you no code to start with, but you have been given a lot of code that can be used as a starting point.

**Q1 (2 points).** Use the approach used in Homework 5 to create an "XOR problem" (as illustrated in the class slides on Tue Oct 12). Allows for 200 training patterns per category (400 training patterns total) and assume the same standard deviations and correlations used for Homework 5. You will need to format the matrix of training patterns and the matrix of teachers to allow for the problem to be learned by a neural network using Keras (as you did in Homework 5). Make sure you display your XOR Problem in a figure.

**Q2 (5 points).** Create a multi-layer neural network that can learn this XOR problem using Keras. You will need to decide the number of hidden layers and nodes per layer (and justify those choices), as well as choose an optimizer and choose a number of epochs and batch size (and justify these choices). Hold out 20% of the training patterns for validation (as will be discussed in class). It will take some iterating of architecture and settings to arrive at choices that work (and that you can justify). Use the plots in Q3 to inform your choices.

In a markdown cell, briefly document the different options you investigated (different network architectures and Keras settings) and why you chose the particular combinations you arrived at.

**Q3 (3 points).** Create a plot of training accuracy as a function of epoch (which you did in Homework 5), and create a plot of both the loss `'loss'` and validation loss `'val_loss'` on the same graph as a function of epoch (which you did not do in Homework 5). In addition, adapting the code from Homework 5, test the network using a mesh grid of test patterns and display the results as a “3D graph” using `plottest()`.

**Q4. (3 points).** Now I want you to create your own classification problem for a neural network to learn (as discussed in class). It must be a problem that is not linearly separable (and is not simply an XOR problem or a match of the example I showed in class). It can be a problem with just two classes of patterns to learn or more than two classes of patterns to learn (the networks and its settings need to be chosen appropriately); if your problem has two classes, there must be more than one distribution per class. I want to see that the distributions that make up your problem have unequal variances and non-zero correlations. Make sure you display your problem in a figure.

**Q5. (5 points).** As with Q2, create a multi-layer neural network that can learn your problem using Keras. Again, you will need to decide the number of hidden layers and nodes per layer (and justify those choices), as well as choose an optimizer and choose a number of epochs and batch size (and justify these choices). Like Q2, hold out 20% of the training patterns for validation. It will take some iterating of architecture and settings to arrive at choices that work (and that you can justify). Use the plots in Q6 to inform your choices.

We will be looking to see that your network is reasonably small (number of layers and number of nodes per layer) to learn your problem (that its complexity is justified). A network with several hidden layers and dozens of nodes per layer might learn your problem, but would be undoubtedly far too complex to be justifiable.

In a markdown cell, briefly document the different options you investigated and why you chose the particular combinations you arrived at.

**Q6 (2 points).** As in Q3, create a plot of training accuracy as a function of epoch, and create a plot of both the loss `'loss'` and validation loss `'val_loss'` on the same graph as a function of epoch. In addition, adapting the code from Homework 5, test the network using a mesh grid of test patterns and display the results as a “3D graph” using a version of `plottest()`. You may need to adjust the mesh grid of test patterns and edit the `plottest()` function to work with your classification problem.

*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.*