ECE 607 Project 2 Part 2

Task 1

Files: Task1\_ReLu.py Task1\_sigmoid.py

For this task I used Keras to define a model with the specified architecture. I used a for loop to try the model with the different numbers of hidden units given. The two files are the same except one of them has the activation function set to sigmoid and the other has it set to ReLu. When compiling the model, I used mean squared error for the loss function and adam for the optimizer. I had the model train for 15 epochs. I then had the model predict outputs for the training and testing data and used the mean\_squared\_error function from Scikit-Learn to calculate the error for each output. The training and testing error for each model and each output is reproduced below. As can be seen from the table, no model is optimal for every output, but for the most part, sigmoid performed better than ReLu.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Training MSE | | | | | | | |
| Function/Units | Output 1 | Output 2 | Output 3 | Output 4 | Output 5 | Output 6 | Output 7 |
| Sigmoid/1 | 0.0406 | 0.0004 | 0.4574 | 0.0004 | 0.4646 | 0.0831 | 0.000008 |
| Sigmoid/4 | 0.1237 | 0.0004 | 0.4574 | 0.0004 | 0.4645 | 0.0831 | 0.000008 |
| Sigmoid/8 | 0.0500 | 0.0011 | 0.4501 | 0.0028 | 0.3420 | 0.1116 | 0.000280 |
| ReLu/1 | 1.5850 | 0.0004 | 0.4799 | 0.0004 | 0.4868 | 0.0831 | 0.000008 |
| ReLu/4 | 0.1613 | 0.0010 | 1.2410 | 0.0076 | 0.5504 | 0.1531 | 0.070718 |
| ReLu/8 | 0.2864 | 0.0037 | 0.4940 | 0.0249 | 0.7762 | 0.2780 | 0.001112 |
| Testing MSE | | | | | | | |
| Function/Units | Output 1 | Output 2 | Output 3 | Output 4 | Output 5 | Output 6 | Output 7 |
| Sigmoid/1 | 0.0410 | 0.0004 | 0.4936 | 0.0004 | 0.5119 | 0.0831 | 0.000009 |
| Sigmoid/4 | 0.1245 | 0.0004 | 0.4936 | 0.0004 | 0.5109 | 0.0830 | 0.000009 |
| Sigmoid/8 | 0.0511 | 0.0011 | 0.4860 | 0.0028 | 0.3734 | 0.1067 | 0.000270 |
| ReLu/1 | 1.5880 | 0.0004 | 0.5217 | 0.0004 | 0.5437 | 0.0830 | 0.000009 |
| ReLu/4 | 0.1680 | 0.0010 | 1.3086 | 0.0075 | 0.5985 | 0.1465 | 0.070638 |
| ReLu/8 | 0.3162 | 0.0039 | 0.5367 | 0.0277 | 0.8651 | 0.2875 | 0.001227 |

Task 2

Files: Task2\_ReLu.py Task2\_Sigmoid.py

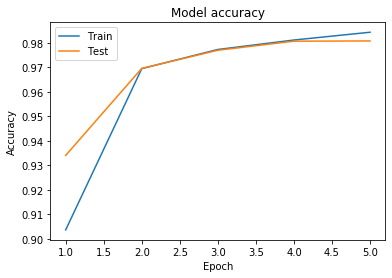
For this task, I used Keras to define the specified model and used a for loop to try every given number of hidden units. I used one file per activation function in this task as well. I used the Keras function to\_categorical to encode the classes. For this problem, I used categorical\_crossentropy for the loss function and adam for the optimizer. I trained the model for 15 epochs. The training and testing accuracy for each activation function and number of hidden units is reported in the table below. As can be seen, ReLu performed better than sigmoid, and the model performed better with a higher number of units in the hidden layer.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Functions/Units | Sigmoid/5 | Sigmoid/10 | Sigmoid/13 | ReLu/5 | ReLu/10 | ReLu/13 |
| Training Accuracy | 0.776 | 0.868 | 0.886 | 0.855 | 0.898 | 0.914 |
| Testing Accuracy | 0.762 | 0.853 | 0.862 | 0.830 | 0.869 | 0.896 |

Task 3

Files: Task3.py

I used Keras to import the dataset and define the given model. The plot below is the training and testing accuracy over 5 epochs of training. I used categorical\_crossentropy for the loss function and Adadelta for the optimizer. The training accuracy was 0.9861 and the testing accuracy was 0.9807.



I tried changing the model and the architecture which achieved the highest accuracy is below. It achieved a training accuracy of 0.9952 and a testing accuracy of 0.9869. The plot for the accuracies at each epoch for this model is below.

First convolutional layer: 16@3x3 + bachNormaliztion + MaxPooling(2x2) + Relu

Second convolutional layer: 32@3x3 + bachNormaliztion + MaxPooling(2x2) + Relu

Third convolutional layer: 64@3x3 + bachNormaliztion + MaxPooling(2x2) + Relu

Fourth fully connected layer: 100 hidden units + Relu

The last classifier layer with softmax for classification (10 units)

