Name: Mikael Ravndal UiO-username: mikaelra

Instructions

- The main program is movements.py. Just run this and it should produce all the answers for the assignments.
- Movements.py is based of mlp.py which has the most important part of the code
- A lot of explaining is done in the commenting in the code

What I have done:

When I saw this assignment I tried to divide it up in several parts so it was easier to get started.

I started with programming the structure and then started basically from bottom up; starting with the forward method and ending with the earlystopping method. I made the confusion method after I made the train method to see how my neural network preformed.

In the end I made the K-fold crossvalidation.

Neural network with different nodes.

Each iteration in training is done with method number three from the assignments FAQ; it chooses a random set from the training data each time.

I have chosen to have each epoch be 10 iterations.

The early-stopping algorithm trains on the train-data, and tests it up to the validation data. When the error is lower than .27 it stops training. The reason for this is that I plotted how the error was in training and it didn't seem to get much lower than .27. Any lower might have resulted in an overfitting later on.

The confusion matrices for each mlp with the different nodes:

Training done! 19 epochs done

6 hidden nodes:

confusion matrix:

[[12. 0. 0. 0. 0. 0. 0. 0.]

[0. 15. 0. 0. 0. 0. 0. 0.]

[0. 0. 13. 1. 0. 0. 0. 0.]

[0. 0. 0. 2. 0. 0. 0. 0.]

[0. 0. 0. 5. 15. 0. 0. 0.]

[0. 0. 0. 0. 0. 9. 0. 0.]

[0. 0. 0. 3. 0. 5. 14. 0.]

[1. 0. 0. 2. 0. 0. 0. 14.]]

Percentage correct on each class:

[1. 1. 0.92857143 1. 0.75 1.

0.63636364 0.82352941]

Average percentage correct:

0.892308059587

Training done! 288 epochs done

8 hidden nodes:

confusion matrix:

[[11. 0. 0. 0. 0. 0. 0. 0.]

[0. 14. 0. 0. 0. 2. 0. 0.]

[1. 0. 13. 1. 0. 0. 1. 3.]

[0. 1. 0. 10. 2. 1. 0. 1.]

[0. 0. 0. 1. 13. 1. 0. 0.]

[0. 0. 0. 0. 0. 4. 0. 0.]

[0. 0. 0. 1. 0. 6. 13. 0.]

[1. 0. 0. 0. 0. 0. 0. 10.]]

Percentage correct on each class:

[1. 0.875 0.68421053 0.66666667 0.86666667 1. 0.65

0.90909091]

Average percentage correct:

0.831454346093

Training done! 76 epochs done

12 hidden nodes:

confusion matrix:

[[13. 0. 0. 0. 3. 0. 0. 0.]

 [0. 15. 0. 0. 0. 2. 0. 3.]

 [0. 0. 13. 1. 0. 0. 0. 1.]

 [0. 0. 0. 4. 0. 0. 0. 1.]

 [0. 0. 0. 1. 12. 0. 0. 0.]

 [0. 0. 0. 0. 0. 1. 0. 0.]

 [0. 0. 0. 7. 0. 11. 14. 0.]

 [0. 0. 0. 0. 0. 0. 0. 9.]]

Percentage correct on each class:

[0.8125 0.75 0.86666667 0.8 0.92307692 1. 0.4375

 1.]

Average percentage correct:

0.823717948718

As we can tell, it didn't really seem to matter how many nodes we used to get a decent result. One thing to notice is that it took 288 epochs to train the net with 8 nodes as hidden layer, compared to 19 with 6 and 76 with 12.

Looking at all of the different confusion tables, it seems to have the most trouble classifying class 7. It confuses it mostly with class 6. Maybe this has something to do with where the data is based, that class 6 and class 7 is similar in hand movement.

K-fold crossvalidation

The k-fold crossvalidation is at the bottom of the movements.py file.

First, I split the original data such that 47 of them is my testing data. Then I have 400 left, which I chose to split up into 10 pieces with 40 datasets in each.

Then it trains on 9 on the pieces, validates on one of them and lastly tests on the testdata from the beginning.

I chose to just use the average and standard deviation methods that numpy already have, as I made them in the previous assignment.

The output from the program.

Average percentage correct of each class:

[0.98888889 0.76095238 0.92666667 0.83964286 0.68306333 0.88919192

  0.78571429 0.815   ]

Standard deviation of the percentages:

[0.03333333 0.12125197 0.11718931 0.19676238 0.20635872 0.14929966

  0.18311676 0.22254213]