## **Assignment 5 – Report**

**Step 4. Analysis and Improvement**

* **We experimented with total number of stumps that were being used in the model for Adaboost:**

These were the stump counts we considered: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 25, 50, 75 and 100.

Here are the results of the 5 fold cross validation (90% Train, 10% Test) where each stump was built after considering 1000 random pixel pairs:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Cross Validation Set 0 | |  | Cross Validation Set 1 | |
| # Stumps | Accuracy Percentage | Training in min. |  | Accuracy Percentage | Training in min. |
| 1 | 46.88 | 3.977016667 |  | 45.15 | 5.883383 |
| 2 | 52.77 | 6.60905 |  | 64.66 | 9.3789 |
| 3 | 66.21 | 9.4046 |  | 65.85 | 13.1009 |
| 4 | 65.51 | 11.32633333 |  | 64.26 | 16.51337 |
| 5 | 67.33 | 14.74508333 |  | 66.86 | 19.37835 |
| 6 | 66.69 | 19.89325 |  | 66.28 | 22.60097 |
| 7 | 68.22 | 21.93136667 |  | 68.73 | 25.70575 |
| 8 | 67.78 | 26.11415 |  | 68.24 | 29.67665 |
| 9 | 67.69 | 29.17036667 |  | 67.66 | 32.39652 |
| 10 | 67.1 | 32.43828333 |  | 68.91 | 35.2543 |
| 11 | 68.54 | 36.62983333 |  | 68.21 | 36.02868 |
| 25 | 71.11 | 73.75721667 |  | 70.83 | 73.04732 |
| 50 | 70.9 | 135.31785 |  | 70.93 | 139.4286 |
| 75 | 71.85 | 205.3504833 |  | 70.87 | 198.5339 |
| 100 | 72.02 | 265.2615167 |  | 69.89 | 270.6328 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Cross Validation Set 2 | |  | Cross Validation Set 3 | |
| # Stumps | Accuracy Percentage | Training in min. |  | Accuracy Percentage | Training in min. |
| 1 | 48.8 | 6.1755 |  | 49.2 | 5.38035 |
| 2 | 58.97 | 9.814167 |  | 59.01 | 9.64675 |
| 3 | 61.52 | 13.689 |  | 65.19 | 13.3636 |
| 4 | 65.78 | 16.57533 |  | 66.25 | 18.11042 |
| 5 | 66.77 | 20.46783 |  | 67.01 | 19.7181 |
| 6 | 68.06 | 23.456 |  | 65.74 | 24.00233 |
| 7 | 68.18 | 26.70242 |  | 69.13 | 26.04028 |
| 8 | 68.24 | 29.32987 |  | 69.13 | 29.69685 |
| 9 | 67.27 | 31.92778 |  | 68.8 | 31.90455 |
| 10 | 69.65 | 34.54398 |  | 70.04 | 35.47075 |
| 11 | 68.45 | 38.03082 |  | 69.19 | 37.90528 |
| 25 | 70.62 | 75.62317 |  | 71.16 | 73.03033 |
| 50 | 70.91 | 136.819 |  | 72.04 | 134.6611 |
| 75 | 71.58 | 212.4739 |  | 71.34 | 207.8133 |
| 100 | 69.74 | 273.3756 |  | 72.16 | 271.766 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Cross Validation Set 4 | |  | Final Model | |
| # Stumps | Accuracy Percentage | Training in min. |  | Avg Validation Accuracy % | Avg Training Min. |
| 1 | 46.55 | 6.261033 |  | 47.316 | 5.535457 |
| 2 | 54.75 | 9.959017 |  | 58.032 | 9.081577 |
| 3 | 66.81 | 13.02698 |  | 65.116 | 12.51702 |
| 4 | 66.96 | 16.44285 |  | 65.752 | 15.79366 |
| 5 | 68.5 | 20.12213 |  | 67.294 | 18.8863 |
| 6 | 67.84 | 22.66145 |  | 66.922 | 22.5228 |
| 7 | 68.65 | 27.46788 |  | 68.582 | 25.56954 |
| 8 | 70.07 | 28.54202 |  | 68.692 | 28.67191 |
| 9 | 69.41 | 32.45727 |  | 68.166 | 31.5713 |
| 10 | 68.05 | 35.5341 |  | 68.75 | 34.64828 |
| 11 | 70.56 | 37.28455 |  | 68.99 | 37.17583 |
| 25 | 71.58 | 77.00667 |  | 71.06 | 74.49294 |
| 50 | 71.46 | 138.08 |  | 71.248 | 136.8613 |
| 75 | 71.64 | 211.015 |  | 71.456 | 207.0373 |
| 100 | 71.89 | 269.77 |  | 71.14 | 270.1612 |

After testing on the given test data, using the 75 stumps model the best testing accuracy percentage found was found to be 70.2%

* **We also experimented with different features in Neural Network:**

The first two parameters tested for were the dimension of the hidden layer and the hidden layer activation function.

Having fixed the following parameters:

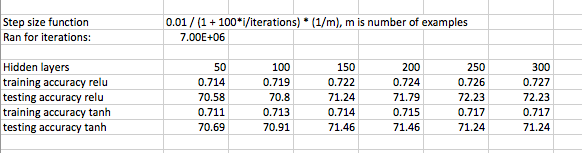
--> Number of iterations = 7\*1e6

--> Step size for iteration i = 0.01 / (1 + 100\*i/iterations) \* (1/m), m is number of examples

Testing values ranging from 50 to 300 with increments of 50, for both reLU

--> The results were best between 150 and 200, near the dimensionality of the data, with testing accuracy of 71.79% (best: ReLU nn\_hdim = 200)

Using tanh or or reLU as the hidden layer activation function gave test accuracy with a difference less than .5%.



ReLU was chosen for two further tests:

(1) check the effect of decreasing the step size 100 times less. The new function was:

Step size for iteration i = 0.01 / (1 + i/iterations) \* (1/m), m is number of examples

Number of iterations = 7\*1e6

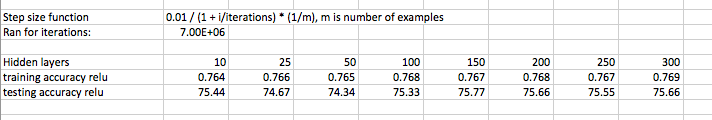
--> Training accuracy increased from 70-72% to 76-77%.

--> Testing accuracy increased from 70-72% to 74-76%.

--> We also tested for smaller hidden layer dimensions: 10 and 25, noticing that the results were no worse than those of higher dimensions.

This result is in line with the fact that adaBoost reached its best performance using only 20 features. We assume there is much redundancy in

the image data and all learning algorithms work efficiently with the key features.



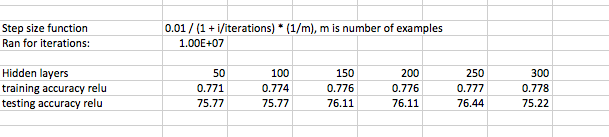
(2) check the effect of adding many more iterations

Number of iterations = 1e7:

Step size for iteration i = 0.01 / (1 + i/iterations) \* (1/m), m is number of examples

--> Training accucary increased from 76-77% to 77-78%.

--> Testing accuracy increased from 74-76% to 75-77%



* **Few Images that were found to be misclassified:**
  + Errors made by our models:
    - Patterns found in errors: The images were Grayscale so Adaboost failed to find the correct orientations.

|  |  |  |
| --- | --- | --- |
| Image | Given Orientation | Predicted Orientation |
| test/112713406.jpg | 0 | 180 |
| test/164225933.jpg | 180 | 0 |

* Other Patterns found in errors:
  + Also, here are a few images which were classified correctly by Adaboost, but were considered misclassified because of bugs in the training data:

|  |  |  |
| --- | --- | --- |
| Image | Given Orientation | Predicted Orientation |
| test/14645715459.jpg | 90 | 0 |
| test/14715745533.jpg | 180 | 0 |
| test/14826675581.jpg | 270 | 0 |
|  |  |  |

Some misclassified images with neural networks:

5

9

10

12

13

14

22

24

27

46

47

56

70

Were either black and white (nearly), or upside down, but the sky looked like the sea. these could be misinterpreted.