CPME 452 Assignment 2 – Backpropagation

Adam Grace
20055169
October 25th, 2020

Model 1 – From Scratch

I create a 784 input, 10 output backpropagation neural network using momentum to classify the handwritten digits from the MNIST dataset. In training and in testing I achieved an overall accuracy of 90.95% and 91.13% respectively. The parameters that I varied while investigating this model include the number nodes in the hidden layer, the number of epochs, the learning rate, the momentum rate, and the initial weight range. After trying a limited number of variable combinations, I settled on 30 hidden nodes, 5 epochs, a learning rate of 0.01, a momentum rate of 0.3 and an initial weight range of negative 1 to 1. Initial weights were chosen randomly from this range. Unfortunately, I was not able to change around the variables more than about 10 times due to the long computation time. For data preprocessing, I flattened the 28x28 2D input array into a 784 element 1D array. I also found that a higher overall accuracy was achieved when I introduced input normalization. The confusion matrices and the precision, recall and overall accuracy statistics are shown below. The confusion matrices goes from 0 to 9, starting from the top left.

```
TRAINING PHASE
Overall accuracy:
90.94833333333334 %
              precision    recall  f1-score   support

           0       0.93      0.97      0.95      5923
           1       0.94      0.97      0.96      6742
           2       0.92      0.87      0.89      5958
           3       0.91      0.88      0.89      6131
           4       0.90      0.91      0.91      5842
           5       0.88      0.86      0.87      5421
           6       0.92      0.95      0.94      5918
           7       0.93      0.92      0.93      6265
           8       0.87      0.87      0.87      5851
           9       0.88      0.89      0.88      5949

    accuracy                           0.91     60000
   macro avg       0.91      0.91      0.91     60000
weighted avg       0.91      0.91      0.91     60000

[[5729    1   17   14    7   43   44    4   57    7]
 [   2 6545   30   21   11   33   14   18   47   21]
 [  72   70 5194  108   96   21  125  105  130   37]
 [  45   27  135 5385   14  209   34   64  152   66]
 [  17   37   23    1 5326    6   83   15   60  274]
 [  99   25   36  172   93 4682  113   29  116   56]
 [  57   23   32    1   32   85 5628    6   50    4]
 [  33   66   86   26   92   15    2 5749   23  173]
 [  46  119   75  129   46  177   52   32 5062  113]
 [  51   28   26   85  199   43   11  143   94 5269]]
```

```
TESTING PHASE
Overall accuracy:
91.13 %
              precision    recall  f1-score   support

           0       0.93      0.98      0.95       980
           1       0.96      0.98      0.97      1135
           2       0.94      0.88      0.91      1032
           3       0.90      0.89      0.90      1010
           4       0.90      0.91      0.91       982
           5       0.87      0.87      0.87       892
           6       0.91      0.94      0.92       958
           7       0.93      0.90      0.92      1028
           8       0.87      0.86      0.87       974
           9       0.89      0.89      0.89      1009

    accuracy                           0.91     10000
   macro avg       0.91      0.91      0.91     10000
weighted avg       0.91      0.91      0.91     10000

[[ 958    0    1    2    0    4    9    1    5    0]
 [   0 1111    3    2    1    1    3    2   12    0]
 [  14   10  913   18   17    5   12   17   21    5]
 [   5    1   15  901    2   35    3   12   25   11]
 [   3    3    1    1  894    2   21    2   11   44]
 [  17    2    5   26   13  778   20    6   17    8]
 [  15    3    3    2   14   17  898    1    5    0]
 [   3   19   23    8    9    2    1  928    6   29]
 [  10    7    6   26    7   34   17   11  837   19]
 [   7    7    1   15   32   13    5   15   19  895]]
```

Model 2A – Using Predefined Libraries and Parameters from Model 1

Using the same parameters as in Model 1, I created the same neural network using predefined libraries from Keras. In training and in testing I achieved an overall accuracy of 89.98% and 90.60% respectively. The confusion matrices and the precision, recall and overall accuracy statistics are shown below. The confusion matrices goes from 0 to 9, starting from the top left.

```
TRAINING PHASE
Overall accuracy:
89.975 %
              precision    recall  f1-score   support

           0       0.93      0.97      0.95      5923
           1       0.92      0.97      0.94      6742
           2       0.90      0.87      0.88      5958
           3       0.90      0.86      0.88      6131
           4       0.89      0.91      0.90      5842
           5       0.89      0.80      0.84      5421
           6       0.91      0.95      0.93      5918
           7       0.92      0.91      0.92      6265
           8       0.88      0.86      0.87      5851
           9       0.86      0.88      0.87      5949

    accuracy                           0.90     60000
   macro avg       0.90      0.90      0.90     60000
weighted avg       0.90      0.90      0.90     60000

[[5728    0   29   13   19   20   46    5   58    5]
 [   0 6536   39   22    5   37   13   19   58   13]
 [  69   84 5183   78  119   13  125  124  134   29]
 [  36   43  180 5283    8  229   53   87  118   94]
 [   8   48   35    2 5310    4   83   14   39  299]
 [ 137   78   58  231  105 4346  144   35  203   84]
 [  58   24   49    4   40   69 5642    0   31    1]
 [  39  109   81   15   82    9    3 5730   13  184]
 [  49  177   85  150   41  137   67   29 5006  110]
 [  62   38   33  104  214   38    6  187   46 5221]]
```

```
TESTING PHASE
Overall accuracy:
90.6 %
              precision    recall  f1-score   support

           0       0.92      0.98      0.95       980
           1       0.95      0.98      0.96      1135
           2       0.92      0.87      0.89      1032
           3       0.90      0.89      0.90      1010
           4       0.89      0.91      0.90       982
           5       0.91      0.81      0.86       892
           6       0.91      0.95      0.93       958
           7       0.91      0.90      0.91      1028
           8       0.88      0.86      0.87       974
           9       0.87      0.88      0.88      1009

    accuracy                           0.91     10000
   macro avg       0.91      0.90      0.90     10000
weighted avg       0.91      0.91      0.91     10000

[[ 963    0    2    1    0    0    9    1    4    0]
 [   0 1108    1    5    1    1    5    2   12    0]
 [  14    7  901   16   17    0   17   21   33    6]
 [   4    2   21  902    1   27    5   17   18   13]
 [   1    5    5    0  898    1   14    2    5   51]
 [  20    4    7   45   16  723   19   13   37    8]
 [  16    3    4    1    8   12  910    1    3    0]
 [   6   22   27    2   11    0    0  926    1   33]
 [  10   12   11   22    9   21   18   14  841   16]
 [  18    8    3   11   45    9    2   18    7  888]]
```

Model 2B – Using Predefined Libraries and New Parameters

Model 2A was used with new parameters in an attempt to achieve better results. 100 hidden nodes, 5 epochs, a learning rate of 0.1, a momentum rate of 0.5 and an initial weight range of negative 1 to 1 was used. In training and in testing I improved upon previous models, achieving an overall accuracy of 89.98% and 90.60% respectively. The confusion matrices and the precision, recall and overall accuracy statistics are shown below. The confusion matrices goes from 0 to 9, starting from the top left.

```
TRAINING PHASE
Overall accuracy:
96.29 %
              precision    recall  f1-score   support

           0       0.97      0.98      0.98      5923
           1       0.98      0.98      0.98      6742
           2       0.96      0.96      0.96      5958
           3       0.95      0.95      0.95      6131
           4       0.94      0.97      0.95      5842
           5       0.97      0.95      0.96      5421
           6       0.98      0.97      0.98      5918
           7       0.97      0.97      0.97      6265
           8       0.94      0.97      0.95      5851
           9       0.97      0.92      0.95      5949

    accuracy                           0.96     60000
   macro avg       0.96      0.96      0.96     60000
weighted avg       0.96      0.96      0.96     60000

[[5818    1    7    5    9    9   24    5   41    4]
 [   1 6581   37   36   15   11    2   13   40    6]
 [  25   15 5701   42   53    3   11   50   52    6]
 [   3   10   65 5850    5   53   14   46   66   19]
 [  13   10   23    3 5677    2   21    9   19   65]
 [  25    6   17   82   25 5171   32    6   36   21]
 [  37   10    8    3   27   56 5755    0   22    0]
 [  12   20   40   20   44    5    2 6078   14   30]
 [  19   35   21   47   17   26   18    7 5651   10]
 [  28   10   12   82  181   17    1   68   58 5492]]
```

```
TESTING PHASE
Overall accuracy:
95.86 %
              precision    recall  f1-score   support

           0       0.95      0.99      0.97       980
           1       0.98      0.98      0.98      1135
           2       0.96      0.96      0.96      1032
           3       0.94      0.97      0.95      1010
           4       0.94      0.97      0.95       982
           5       0.96      0.93      0.95       892
           6       0.96      0.96      0.96       958
           7       0.96      0.96      0.96      1028
           8       0.95      0.97      0.96       974
           9       0.97      0.91      0.94      1009

    accuracy                           0.96     10000
   macro avg       0.96      0.96      0.96     10000
weighted avg       0.96      0.96      0.96     10000

[[ 968    0    0    4    0    3    2    1    1    1]
 [   0 1115    2    3    0    1    5    2    7    0]
 [   8    0  988    7    8    0    4    7   10    0]
 [   0    0   10  978    0    6    1    9    6    0]
 [   2    1    4    1  949    0    5    3    5   12]
 [   8    1    1   25    3  827   14    1    8    4]
 [  10    3    3    2    6   10  917    0    7    0]
 [   3   10   16    3    4    0    0  983    2    7]
 [   5    1    2    7    5    4    5    3  942    0]
 [  10    6    1   13   39    6    0   10    5  919]]
```

Discussion

Interestingly, on all models the testing overall accuracy was greater than the training overall accuracy. This was unexpected since neural networks tend to fit to the data it is trained on. This indicates that overfitting was not an issue. In Model 2B, I achieved much better overall accuracy, precision and recall. I believe that the greater part of this improvement is due to the increase in hidden nodes from 30 to 100. It was observed that models 1 and 2A had some difficulty classifying 5 and 9, possibly due to not enough hidden nodes. Unrelated to the results, I also noticed that the predefined libraries were able to train the network in less time than my program from scratch, liekly due to operation optimization.