FREEDOM ALEN

ID:330726

# Pattern recognition laboratory #5&#6:

# Fashion items classification with neural networks

# 1. Modifying given set of scripts

The scripts, originally designed for single-hidden-layer networks, have been modified to support an arbitrary number of hidden layers. However, due to Octave's xlswrite function limitations, only the first two layers' sizes are included in the XLS reports. The scripts allow for numerous trials by specifying different setups in the TESTBENCH.m file. The training process halts if there's no improvement after seven consecutive epochs. This balance minimizes unnecessary training while still allowing for potential long-term performance improvements.

# 2. Reference solution

Over fifty neural networks were trained, with a few consisting of a single hidden layer. However, these were unable to surpass the benchmark result of 87.42%. The transition to double-hidden-layer networks led to improved performance, exceeding the target by approximately 0.5 percentage points without implementing additional strategies from the article. The networks primarily tested had equal layer sizes. Reducing the size of subsequent layers did not significantly decrease the testing time needed to evaluate the network's potential. II suspect that one needs, in terms of double-hidden-layer network, at least a size of 90 to beat the predefined result. In this report I included layers with 140 neurons.

# 3. Reference solution

## 3.1 Discussion about parameters and their influence

During the laboratory I trained more than 50 neural networks – they differ in such parameters as:

- Number of neurons
- Number of layers
- Number of epoch
- Learning rate

The first observation was that for the networks having single-hidden-layer network the success rate was much smaller than one presented on the task description (87,42%). Therefore several approaches was performed in order to obtain sufficient result.

a) **Multi-hidden-layer approach** (in particular, a double-hidden-layer network) During the experiments, I noticed that the number of layers in a neural network has a great impact on the success rate. Too many or too few layers have a negative influence on the obtained results. In this case, the best option was to choose a double-hidden-layer network. It allowed me to achieve much better results than in the case of a single-hidden-layer network without any additional modifications.

b) **Number of neurons** in the network I performed several experiments that took into consideration the number of neurons in the network. I decided to start with a small number of neurons (50) and see what would happen. The obtained results showed that in order to get a higher OK value than the reference one, I needed more than 90 neurons. However, using too many neurons is not a good idea either. Therefore, I decided to use 140 neurons as it was an optimal value.

c) **Learning rate** For this parameter, I tried two possible approaches: dynamic learning rate and fixed value of learning rate. The outcome of both approaches showed that for this case, the better option is to use a fixed value of learning rate. The best results were obtained for a learning rate value between 0.01 and 0.1 depending on other parameters (such as the number of neurons). Although I noticed that the lower value of learning rate is better due to the fact that using a 0.1 value causes a high variance – overfitting. Therefore, I chose a learning rate equal to 0.01.

d) **Number of epochs** I decided to use 50 epochs as it is given in the script. In general, too high a number of epochs allowed to run may be the reason for overfitting the network. And I also set a condition to stop learning if there is no improvement for seven consecutive epochs.

## 3.2 Experiment results

Takin under consideration all mentioned above parameters I decided to use as a reference solution network having:

- No. of layers: 2
- No. of neurons : 140
- Learning rate: 0.1

***Table : Success rate and training time for training and testing sets.***

| Epoch no | Time | Train Set OK | Test Set OK |
|---:|---:|---:|---:|
| 1 | 291.1112 | 0.7760 | 0.7658 |
| 2 | 291.168 | 0.8437 | 0.8286 |
| 3 | 292.7369 | 0.8554 | 0.8376 |
| 4 | 291.0661 | 0.8674 | 0.8532 |
| 5 | 291.5266 | 0.8745 | 0.8595 |
| 6 | 290.7171 | 0.8799 | 0.8656 |
| 7 | 290.9874 | 0.8861 | 0.8666 |
| 8 | 292.25 | 0.8882 | 0.8669 |
| 9 | 292.0845 | 0.8893 | 0.8678 |
| 10 | 290.8745 | 0.8930 | 0.8682 |
| 11 | 291.8059 | 0.8952 | 0.8685 |
| 12 | 291.8683 | 0.8968 | 0.8699 |
| 13 | 290.2479 | 0.8980 | 0.8710 |
| 14 | 291.8818 | 0.8991 | 0.8713 |
| 15 | 291.1922 | 0.9007 | 0.8728 |
| 16 | 290.0419 | 0.9036 | 0.8745 |
| 17 | 290.5668 | 0.9050 | 0.8763 |
| 18 | 291.8467 | 0.9048 | 0.8737 |
| 19 | 292.2578 | 0.9048 | 0.8727 |
| 20 | 290.3683 | 0.9039 | 0.8746 |
| 21 | 290.8111 | 0.9064 | 0.8741 |
| 22 | 292.6171 | 0.9112 | 0.8781 |

| | | | |
|---|---|---|---|
| 23 | 290.3368 | 0.9100 | 0.8777 |
| 24 | 291.6101 | 0.9142 | 0.8786 |
| 25 | 291.6882 | 0.9124 | 0.8780 |
| 26 | 291.3929 | 0.9130 | 0.8769 |
| 27 | 290.4603 | 0.9158 | 0.8777 |
| 28 | 301.2906 | 0.9140 | 0.8774 |
| 29 | 292.2072 | 0.9138 | 0.8740 |
| 30 | 291.3752 | 0.9131 | 0.8762 |
| 31 | 293.9685 | 0.9164 | 0.8787 |
| 32 | 293.0036 | 0.9147 | 0.8748 |
| 33 | 293.0142 | 0.9173 | 0.8768 |
| 34 | 294.0627 | 0.9175 | 0.8787 |
| 35 | 292.8802 | 0.9193 | 0.8787 |
| 36 | 294.2323 | 0.9202 | 0.8791 |
| 37 | 292.5626 | 0.9199 | 0.8784 |
| 38 | 293.1228 | 0.9216 | 0.8800 |
| 39 | 293.7389 | 0.9246 | 0.8798 |
| 40 | 293.0184 | 0.9234 | 0.8795 |
| 41 | 292.6469 | 0.9218 | 0.8780 |
| 42 | 297.9292 | 0.9244 | 0.8803 |
| 43 | 292.5008 | 0.9249 | 0.8794 |
| 44 | 292.1617 | 0.9256 | 0.8807 |
| 45 | 292.7337 | 0.9281 | 0.8791 |
| 46 | 293.0702 | 0.9269 | 0.8794 |
| 47 | 291.9018 | 0.9283 | 0.8789 |
| 48 | 292.3028 | 0.9281 | 0.8767 |
| 49 | 291.1906 | 0.9291 | 0.8807 |
| 50 | 291.185 | 0.9291 | 0.8788 |

At epoch 44, the test set accuracy reached 88.07%, which is higher than 87.42%. This is a positive indication that the added complexity of a second layer and additional neurons is helping the model to learn more intricate patterns in the data.

However, while the model's performance on the training set continues to improve, the test set accuracy seems to plateau after a certain point. This could be a sign of overfitting.

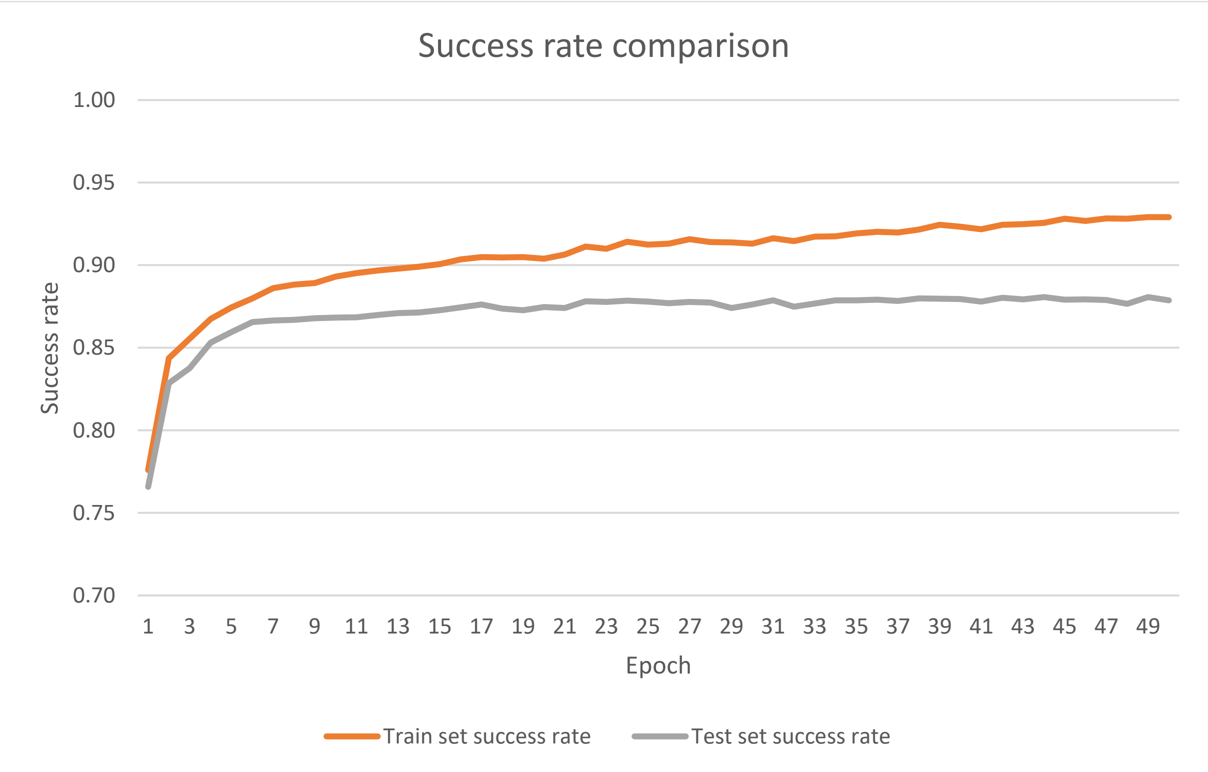
***Confusion matrix for epoch 44:***

| Class | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | Rejected |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 835 | 1 | 29 | 11 | 4 | 3 | 109 | 0 | 8 | 0 | 0 |
| 2 | 3 | 969 | 2 | 13 | 3 | 0 | 9 | 0 | 1 | 0 | 0 |
| 3 | 16 | 1 | 823 | 7 | 96 | 1 | 54 | 0 | 2 | 0 | 0 |
| 4 | 30 | 6 | 19 | 851 | 36 | 0 | 56 | 0 | 2 | 0 | 0 |
| 5 | 2 | 0 | 80 | 21 | 835 | 0 | 58 | 0 | 4 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 935 | 0 | 35 | 2 | 28 | 0 |
| 7 | 110 | 0 | 106 | 19 | 92 | 0 | 662 | 0 | 11 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 13 | 0 | 968 | 0 | 19 | 0 |
| 9 | 6 | 0 | 3 | 4 | 2 | 2 | 10 | 5 | 968 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 7 | 1 | 31 | 0 | 961 | 0 |

From the confusion matrix, it appears that the model struggles most with class 7. There are significant misclassifications happening with this class. For instance, instances of class 7 are often incorrectly predicted as class 1, 3, and 5. This could indicate that there are some shared features or characteristics between these classes that the model is finding hard to distinguish.

Moreover, the model also seems to have some difficulty with class 1 and 4, where a number of instances are misclassified as class 7. This could be due to the model overgeneralizing certain features of class 7.

***Comparison of training and test sets success rate:***




The overfitting visible in the graph above, which shows a divergence between the training and test success rates over epochs, is similar to what we've observed in the laboratory script.

## 4. Improving reference model

From the attached article I decided to apply 2 changes – **Normalization** of the input and the **Shuffling**.

### 3.3 Normalization of the input

As a first improvement of the reference model I applied the normalization of the input. The neural network was trained again with the constant neuron number but for different learning rate. Once again I check values from range 0.1 to 0.01 as they gave the best results in the reference.

***Table : Success rate and training time for training and testing sets.***

| Epoch no | Time | Train Set OK | Test Set OK |
|---|---|---|---|
| 1 | 290.49788 | 0.8255 | 0.8136 |
| 2 | 293.37612 | 0.8576833 | 0.8419 |
| 3 | 292.94179 | 0.8696833 | 0.8522 |
| 4 | 292.60265 | 0.8773333 | 0.8589 |
| 5 | 289.20057 | 0.8834 | 0.8631 |
| 6 | 289.6689 | 0.88855 | 0.8671 |
| 7 | 288.59899 | 0.89315 | 0.8694 |
| 8 | 285.62127 | 0.897 | 0.8725 |
| 9 | 285.779 | 0.9006833 | 0.8755 |
| 10 | 285.58679 | 0.9039333 | 0.8775 |
| 11 | 286.38625 | 0.9071667 | 0.8798 |
| 12 | 290.44523 | 0.9102 | 0.881 |
| 13 | 293.9483 | 0.9134833 | 0.882 |
| 14 | 289.51794 | 0.9167167 | 0.8828 |
| 15 | 291.54928 | 0.91935 | 0.8831 |
| 16 | 288.3963 | 0.92185 | 0.8832 |
| 17 | 285.68293 | 0.92435 | 0.8835 |
| 18 | 287.87141 | 0.9268 | 0.8848 |
| 19 | 287.56329 | 0.9294833 | 0.8855 |
| 20 | 282.48105 | 0.9316167 | 0.8856 |
| 21 | 287.0459 | 0.9337667 | 0.8857 |
| 22 | 287.75439 | 0.9356167 | 0.8859 |
| 23 | 281.24215 | 0.9375167 | 0.8857 |
| 24 | 286.39625 | 0.9394167 | 0.8861 |
| 25 | 284.50668 | 0.9414667 | 0.8857 |
| 26 | 288.98435 | 0.9430667 | 0.886 |
| 27 | 286.05846 | 0.9444667 | 0.8863 |
| 28 | 316.38214 | 0.9463 | 0.886 |
| 29 | 320.11458 | 0.9480333 | 0.886 |
| 30 | 291.41211 | 0.9494167 | 0.8867 |
| 31 | 292.31118 | 0.9507667 | 0.8867 |
| 32 | 291.75514 | 0.9520667 | 0.8865 |
| 33 | 290.62517 | 0.9533833 | 0.8858 |

| | | | |
|---|---|---|---|
| 34 | 291.53644 | 0.9546833 | 0.8855 |
| 35 | 290.96165 | 0.9560667 | 0.8855 |
| 36 | 291.46767 | 0.9572167 | 0.8853 |
| 37 | 291.82784 | 0.9585333 | 0.8856 |
| 38 | 291.8716 | 0.9597833 | 0.8863 |

The table shows the performance of a double-layer network with 140 neurons over 38 epochs. The training was stopped early due to no improvement in test set accuracy for 7 successive epochs. The key events are:

- The model started with a training set accuracy of 82.55% and a test set accuracy of 81.36% at epoch 1.
- The test set accuracy improved significantly and reached a peak of 88.67% at epoch 30.
- Despite the continuous improvement in the training set accuracy, the test set accuracy plateaued and slightly decreased after epoch 22, indicating potential overfitting.
- The training was stopped at epoch 38 due to the early stopping strategy implemented.

Overall, the normalization of input data and the early stopping strategy helped improve the model's performance and prevent overfitting. However, there's still room for improvement to close the gap between the training and test accuracies.

***Confusion matrix for epoch 30:***

| Class | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | Rejected |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 877 | 2 | 15 | 19 | 6 | 1 | 72 | 0 | 8 | 0 | 0 |
| 2 | 3 | 965 | 2 | 21 | 4 | 0 | 5 | 0 | 0 | 0 | 0 |
| 3 | 17 | 1 | 774 | 12 | 126 | 1 | 69 | 0 | 0 | 0 | 0 |
| 4 | 22 | 5 | 8 | 908 | 27 | 1 | 25 | 0 | 4 | 0 | 0 |
| 5 | 1 | 0 | 59 | 32 | 855 | 0 | 47 | 0 | 6 | 0 | 0 |
| 6 | 1 | 0 | 0 | 1 | 0 | 929 | 0 | 36 | 5 | 28 | 0 |
| 7 | 145 | 0 | 69 | 25 | 77 | 0 | 672 | 0 | 12 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 13 | 0 | 971 | 0 | 16 | 0 |
| 9 | 1 | 1 | 3 | 5 | 4 | 3 | 9 | 4 | 970 | 0 | 0 |
| 10 | 1 | 0 | 0 | 0 | 0 | 6 | 2 | 45 | 0 | 946 | 0 |

The confusion matrix shows that the model performs well on most classes. However, it seems to struggle with class 7, where a significant number of instances are misclassified. The model also seems to confuse between classes 3 and 5.

## 3.4 Shuffling

Shuffling the data can help improve the model's ability to generalize because it ensures that the model gets a mix of instances from all classes in each batch. This can lead to more robust learning and better performance on the test set.

***Table : Success rate and training time for training and testing sets.***

| Epoch no | Time(sec) | Train Set OK | Test Set OK |
|---|---|---|---|
| 1 | 291.11121 | 0.71395 | 0.7113 |
| 2 | 291.16799 | 0.76485 | 0.7559 |
| 3 | 292.73692 | 0.8017167 | 0.7946 |
| 4 | 291.06605 | 0.8364167 | 0.8229 |
| 5 | 291.52664 | 0.84655 | 0.8335 |
| 6 | 290.71707 | 0.8524333 | 0.8376 |
| 7 | 290.98736 | 0.8576333 | 0.8414 |
| 8 | 292.24998 | 0.86165 | 0.8451 |
| 9 | 292.0845 | 0.8651 | 0.8486 |
| 10 | 290.87448 | 0.8678833 | 0.8513 |
| 11 | 291.80593 | 0.8703333 | 0.8534 |
| 12 | 291.86833 | 0.8731667 | 0.8559 |
| 13 | 290.24794 | 0.87515 | 0.8576 |
| 14 | 291.88178 | 0.87695 | 0.8604 |
| 15 | 291.19223 | 0.8787667 | 0.8615 |
| 16 | 290.04186 | 0.8804333 | 0.8619 |
| 17 | 290.5668 | 0.88195 | 0.8631 |
| 18 | 291.84668 | 0.8833167 | 0.864 |
| 19 | 292.25776 | 0.88495 | 0.8652 |
| 20 | 290.36829 | 0.8863167 | 0.8667 |
| 21 | 290.8111 | 0.8881167 | 0.868 |
| 22 | 292.61714 | 0.8891167 | 0.8687 |
| 23 | 290.33684 | 0.8904333 | 0.8692 |
| 24 | 291.61014 | 0.8916167 | 0.8703 |
| 25 | 291.68821 | 0.8923833 | 0.8713 |
| 26 | 291.39285 | 0.8937167 | 0.8728 |
| 27 | 290.46027 | 0.8947333 | 0.8729 |
| 28 | 301.29059 | 0.8957333 | 0.8731 |
| 29 | 292.2072 | 0.8968833 | 0.874 |
| 30 | 291.3752 | 0.8976667 | 0.8747 |
| 31 | 293.96852 | 0.8987 | 0.8754 |
| 32 | 293.00362 | 0.89955 | 0.8754 |
| 33 | 293.01419 | 0.90035 | 0.8766 |
| 34 | 294.06269 | 0.9014167 | 0.8772 |
| 35 | 292.88024 | 0.9022667 | 0.8775 |
| 36 | 294.23234 | 0.903 | 0.8779 |
| 37 | 292.56256 | 0.90365 | 0.8781 |
| 38 | 293.12277 | 0.9046 | 0.8785 |
| 39 | 293.73894 | 0.9053 | 0.8787 |

| | | | |
|---|---|---|---|
| 40 | 293.01844 | 0.9061 | 0.8794 |
| 41 | 292.64686 | 0.9070333 | 0.8797 |
| 42 | 297.92924 | 0.9079167 | 0.8799 |
| 43 | 292.50078 | 0.9084667 | 0.8804 |
| 44 | 292.16165 | 0.9090833 | 0.8808 |
| 45 | 292.73368 | 0.9098167 | 0.881 |
| 46 | 293.07017 | 0.9104 | 0.8808 |
| 47 | 291.90178 | 0.9111 | 0.8806 |
| 48 | 292.30284 | 0.9118833 | 0.8808 |
| 49 | 291.19057 | 0.9125667 | 0.8806 |
| 50 | 291.18503 | 0.9133833 | 0.8804 |

The table shows the performance of your model over 50 epochs after shuffling the input data. Here are the key points:

- The training started with a training set accuracy of 71.40% and a test set accuracy of 71.13% at epoch 1.
- The test set accuracy improved significantly and reached a peak of 88.10% at epoch 45.
- Despite the continuous improvement in the training set accuracy, the test set accuracy plateaued and slightly decreased after epoch 44, indicating potential overfitting.
- The training continued until epoch 50, with the final training set accuracy at 91.34% and the test set accuracy at 88.04%.

***Confusion matrix for epoch 45:***

| Class | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | Rejected |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **1** | 823 | 1 | 16 | 33 | 5 | 3 | 111 | 0 | 8 | 0 | 0 |
| **2** | 3 | 966 | 1 | 22 | 2 | 0 | 4 | 0 | 2 | 0 | 0 |
| **3** | 14 | 0 | 842 | 10 | 67 | 1 | 64 | 0 | 2 | 0 | 0 |
| **4** | 17 | 9 | 10 | 901 | 25 | 0 | 33 | 0 | 5 | 0 | 0 |
| **5** | 0 | 0 | 122 | 40 | 751 | 0 | 83 | 0 | 4 | 0 | 0 |
| **6** | 0 | 0 | 0 | 1 | 0 | 945 | 0 | 33 | 1 | 20 | 0 |
| **7** | 105 | 0 | 101 | 30 | 53 | 0 | 696 | 0 | 15 | 0 | 0 |
| **8** | 0 | 0 | 0 | 0 | 0 | 23 | 0 | 953 | 0 | 24 | 0 |
| **9** | 3 | 1 | 3 | 3 | 2 | 1 | 11 | 5 | 971 | 0 | 0 |
| **10** | 0 | 0 | 0 | 0 | 0 | 7 | 1 | 30 | 0 | 962 | 0 |

The confusion matrix for epoch 45 shows that the model performs well on most classes. However, it seems to struggle with class 5 and 7, where a significant number of instances are misclassified. The model also seems to confuse between classes 7 and 1, and 3 and 5. Despite these issues, the overall performance is good with a test set accuracy of 88.10%.

# 4 Conclusion

In conclusion, both the reference solution and the modified script, which incorporated shuffling and normalization, encountered challenges in accurately classifying instances of class 7 and class 1. There were also less pronounced difficulties with class 7 and classes 3, 4, and 5, as well as between classes 3 and 5.

A potential remedy for these classification issues could be to discard certain results where the network's prediction confidence is low. While this approach may boost the overall accuracy (OK value), it would also lead to an increase in the number of rejected instances.