Update of the code for Coursework 2

24.11.2017

1. Issue

The previous version of the MLP function was based on the **resilient gradient descent backpropagation algorithm (RPROP)** [1], not the regular, classic version of the backpropagation algorithm taught as a part of the MLNC lecture. The idea behind the RPROP algorithm is to change the weights depending on the derivative of the changes in weights not just the deltas computed from the back-propagated error. As a result the algorithm is able to train neural network much faster in cases when back-propagated error is very small. For more details please see [1].

The reason for such choice was much better overall performance in terms of the classification accuracy obtained using much fewer training epochs compared to the classic backpropagation. In order to make the coursework less time-consuming we had the previous version of the code implemented that way. However in the coursework you are asked to investigate the influence of the learning rate on the learning curves. In fact in the training based on classic RPROP algorithm **is not** influenced by the learning rate, as some of you had noticed after reviewing the code. Therefore you should not have observed any consistent results in question **1a**.

2. Action

The only question affected by this mistake is 1a. Other questions that you had answered are unaffected and you don't have to redo them. To fix this problem we have slightly modified the code you were given. Please download the new version from the course website and from now on work on the updated version.

3. Changes in the code

Please familiarize yourself with the following changes we made in the code. The new version of the code contains the following files:

- **MLP_1a.m** – function that you should use to answer the question 1a. We have modified it, so now it employs **the classic backpropagation algorithm** (exactly as taught in the lecture) to train the neural network. The outputs of this function are the same as previous version of MLP.m, which is accuracy for each training epoch (learning curve) and best prediction obtained from the epoch that has yielded the highest accuracy. This time you should observe the consistent influence of the learning rate on the produced learning curves. As before you should not modify this code but we encourage you to familiarize yourself with it to better understand the idea behind the neural networks.
- **MLP_rest.m** – function you should use to answer the remaining MLP related question of this coursework. This function is the same as previous version of the MLP.m

employing **RPROP** to train the network. Additional change is the removal of the reset of the changes in weights after 200 epochs, as in the end we found it irrelevant for this coursework. As before you should not modify this code but we encourage you to familiarize yourself with it to better understand the idea behind the neural networks.

- run_MLP.m - the same script as before but slightly reorganized. Different sections of the code (marked with standard Matlab syntax %%) are now introduced and filled with the code you have to use to answer each MLP related question. Please use this code to modify the parameters, call the functions (already filled in the sections) and obtain your results.
- TrainClassifierX.m – no changes
- ClassifyX.m – no changes
- SanityCheck.m – no changes
- Minor bug fixes:
    - The 'best_prediction' output is returned in the correct format of a vector of predicted classes for the testing set.
    - There was an extreme unlike scenario (probably not affecting most of you) that your network could assign testing point to multiple classes by producing couple exactly equal outputs in the final layer. Now in such case the label is randomly chosen from those outputs that produced equal results.
    - To perform binary classification with any of the function you should change the labels of activities you are planning to classify to 1 and 2 only. For example if you are planning to classify walking upstairs (3) and walking downstairs (4) you should change their labels to respectively 1 and 2 after extracting them from the default dataset and before running the function.

We would like to apologize for this mistake and thank all of you who made an effort to look into the code and report any bugs or unclear parts either on Piazza or by contacting the GTAs directly. In case there are any other confusing or not clear parts of the code or the coursework please make sure to ask your GTAs on Piazza or during the labs.

Machine Learning and Neural Computation team

References

[1] Martin Riedmiller. Rprop – Description and Implementation Details. Technical report, 1994