Course: Complex Data - Analysis and Visualization, 2016

©Computational Medicine, Dept. Medical Sciences, Uppsala University (EC+MG)

Version: 161121

Exercise #3 - Deep Learning

The general task here is to re-implement *Task D* from last week using mainly your own code, in order to get a better understanding of how a deep neural network of stacked autoencoders (SAE) works and how you should correctly estimate the resulting performance. Additionally, you should employ and train an alternative deep neural network as well as compare its performance with that one from last week. You should use again the digit font dataset provided by MATLAB® *R*2016*b*.

Task A

Re-implement the deep neural network consisting of two SAE and one final softmax layer with structure $784 - 25 - 10 - 1$ from last week in the following way:

(i) Perform pre-training on each and every component of the deep network separately. Instead of using the `encode` MATLAB® for retrieving the encoded features from the autoencoders, write your own code for that (meaning that your code should extract the required weights and biases from the three trained networks and then calculate the network output using the required matrix multiplications etc). For the training you can use the built-in MATLAB® functions `trainAutoencoder` and `trainSoftmaxLayer`. You should use the `encode` function in order to verify your results.

(ii) Stack the separate components and create your deep neural network by using the MATLAB® function `stack`. Also implement the same deep network "manually" means of your own code which should extract all weights and biases and then perform the required matrix multiplications etc. You should confirm (and somehow visualize in your report) that both implementations of the deep network yields the same outputs for a set of test examples.

(iii) Estimate the performance of the deep neural network on the test dataset after the pre-training by using your own code. More precisely, try to use the MATLAB® function `confusion` (http://se.mathworks.com/help/nnet/ref/confusion.html) and estimate the probabilities of detection and false alarm on your own by using its second output argument `cm`. Remark: You can use the function `plotconfusion` in order to visualize and confirm your results, but first you have to figure out which patch of the confusion matrix corresponds to the probability of detection and false alarm.

(iv) Since you have already extracted the weights and biases from the second hidden layer (second autoencoder), make a plot of the 10 neurons' response among your testing examples/dataset. In case you choose to make a common plot for all 10 neurons, you should use different colors and an appropriate legend, so that you are able to identify and present the responses from the different neurons. Similarly, make a plot of your final output (from the output layer of the deep neural network).

(v) Perform fine tuning on your deep neural network.

(vi) Estimate the performance of your fine tuned deep neural network on the test dataset and compare it with the performance you get after pre-training in question (iii). Do you see any difference? Do you think fine tuning is important?

(vii) Make a plot of your 10 neurons' response in the second hidden layer as well as a plot of your final output response. Compare these plots with the respective ones from question (iv). Do you see any differences? Can you identify and present results showing that particular neurons in the second hidden layer seem to behave as "3" or "non-3" digit detectors?

(viii) Find the images that maximize the response of the neurons in the second hidden layer. You do not have to solve any optimization problem: You can simply find the image that gives the maximum response for each of the 10 neurons. Plot the 10 input images yielding maximum responses, one image for each of the hidden neurons, in a common plot. In this way, you can get an idea of which digits seem to stimulate each one of the 10 neurons the most.

Task B

Repeat **Task A** by employing and training a deep neural network of two SAE and one feedforward network with one hidden layer for pattern recognition. In other words, your deep neural network should now consists of two autoencoders and one pattern recognition network (use the MATLAB® function `patternnet`) with seven neurons in the hidden layer and one neuron in the output layer. Thus, the structure of your deep neural network should be $784 - 25 - 10 - 7 - 1$. The final component, which is now a feedforward network, replaces the softmax layer in the previous task.

IMPORTANT: You should use the same non-linear transfer function for all layers of your deep neural network. You should use the standard sigmoid function[1] for a starter and you might try other options provided that there is time left.

More precisely,

(i) Perform all the questions described in **Task A** for the new deep neural network by using again your own code wherever needed.

(ii) Compare the performances after pre-training and fine tuning of the deep neural networks from Tasks **A** and **B**. Do they behave similarly?

[1] $f(x) = \frac{1}{1+e^{-x}}$

Report

You should write a nice report that explains your solutions and provide well written and well documented **source code** which should be straightforward to download and run by your teacher (including a readme.txt file with a description of your m-files files as well as running instructions). The report should have a clear structure with the following titles or something similar.

Introduction: Introduce the exercise and summarize the subtasks.
Material and Methods: Present an overview of the strategies and tools used to solve the different subtasks.
Results: Present selected results to answer specific questions and/or to show particular observations of interest.
Discussion: Present a discussion of your results in relation to the subtasks assigned.
Conclusion: Present your conclusions from the results presented.
Evaluation: Present a short evaluation (approx half a page) of the exercises in terms of what you have learned and what you think can be improved.

Important Dates

Oral Presentation: Friday, November 25

Written Report: Monday, November 28

Good Luck!