Examples for ALNfitDeep

A simple way to get started: You can browse to the text file DevilsTowerRotated.txt. There are two input columns and an output column at the right. The graph looks like a volcano on a square base. For other files, you might have to select the rightmost connection using the buttons Previous and Next then use the pulldown list of column names to choose the output column. Of course you then have to either Remove the original connection to that column or make that connection to the rightmost column. You can improve the result using techniques A and B (see below). Note how the time value at the start of each output file (except the diagnosis ones) keeps the outputs together which belong to a certain run. The .dtr output (which is a text file, viewable by Notepad) describes the solution. The solution is in form of a DTREE, which is an ALN without the learning machinery.

**A**. Using the DevilsTowerRotated.txt file, open the Processing options dialog, click on the button to set the tolerance and enter the value 0.00577 ( which is probably close to what was in the TrainProtocol.txt file of a run where the RMS noise was estimated).

B. The generalization, that is the accuracy of fit on data not used in training, can be improved by averaging over more ALNs which are based on different random samples for training. Set this to the maximum value 10.

C. Use the menu to Open a .fit file, say one using the DevilsTowerRotated.txt data file. Then use the browse button to select some file from the same source (or even the same file). This run does no training, just evaluation using the saved DTREE from a previous run.

D. Classification using this approach is limited to where neighboring classes are represented by successive integers and only the classes of successive integers overlap. Browse to the iris-data.txt file and see how accurate the classification is. One improvement on this software would be to learn the magnitude of the RMS noise as a function over the input space. In the current version of ALNfitDeep, the RMS noise is supposed constant. If it varies proportionally to the signal, one can train on the logarithm of the output value, replacing the output column in the data file.

E. Take the DevilsTowerRotated.txt file and extend it using a spreadsheet by adding about five new columns with arbitrary, say random, values as inputs. Start ALNfitDeep, browse to the extended data file, choose a new output column, the third one from the left, using the drop-down list, then remove the connections to the columns you added

F. For determining which inputs are really necessary for computing a certain output, train as in E above, but don’t remove the columns with random values. Examine the importance values of the inputs after training and remove the connections to the columns of least importance. Train again and repeat. To make this tougher, add noise to the values in all columns.

G. Do you remember the formula PV=nRT for the pressure, volume and temperature of a gas? Make a spreadsheet where P is the output and V and T are the inputs and P=T/V. Add some noise to all columns. The noisy data will not necessarily show that P increases as V decreases, but you know that the ideal function to be learned must show that property. So before training set P as increasing in T and decreasing in V. If you know bounds on V and T, then you can use bounds on the values of the partial derivatives to set weight bounds.

H. The default picks about 10% of the samples from the data to use for testing, then it divides the remaining samples into two halves to determine the noise level. It might be better to determine the noise level without holding back a test set (use 0% for test). Then in a following run, set the noise level in the Processsing options dialog as found in the TrainProtocol.txt file of the first run and train on all the data except 10% held back for testing.

I. Replace some the output values in a copy of the DevilsTowerRotated.txt file by 99999. Train on that data after selecting the Replace option in the Processing options dialog. Check the R output file of the run, where the replacements are done. You can try more complicated real-world data where there are missing values in several columns. The missing values may be replaced by several trainings where the values in columns not removed are completely defined. (advanced: Use knowledge of the functional dependencies in a relational database to determine the sequence of column replacements).

J. Deep learning is faster and allows more complicated functions to be learned. Smoothing allows one to use fewer flat pieces in the learned function, making it easier to analyze. In this software, the only step to promote deep learning is that smoothing can be switched off, however many other improvements could be made using this model of deep learning. This is just an indication of what is possible.