INM427 NECO Coursework troubleshooting

Do: Discuss with your fellow students about your choice of data set and what you are doing for the coursework.

Do: Tell us if you encounter any issues or problems. The most common issues and questions raised by your colleagues are:

(1) What is an appropriate size and level of complexity of the dataset?

(2) What are good methods to choose and what counts as a new method?

(3) How to use a time series dataset, recurrent networks and convolutional nets?

(4) How to use cross-validation or bootstrapping with early-stopping?

(5) How best to report average performance or should I re-train the best model on the entire data set?

In relation to (1), it's preferable to choose a manageable problem than a very complex one – leave the big data for the big data module or the dissertation. Of course, choose ideally a dataset that you find interesting. If it's too easy a dataset (that is, you get very good accuracies quickly no matter how you change the model or add noise to the dataset) then consider alternatives, because you will need to be able to evaluate results critically (see marking scheme). If you can't "break it" (that is, accuracy is very high too easily), you won't have varied graphs and interesting results to evaluate your neural network critically.

In relation to (2), the most popular choices are: perceptrons vs backprop, i.e. multilayer perceptrons (MLP), MLP vs SVMs, Hopfield nets vs. Boltzmann machines, including RBMs, MLP vs CNN, and MLP (sliding window) vs RNNs.

Make sure you do not choose to compare methods that are incomparable, e.g. backprop (supervised) and Hopfield nets (unsupervised). You can pre-process your data, e.g. with PCA, but this counts more as a variation (pre-processing) of a method and not as a new method. Convolutional nets can be evaluated in comparison with MLP and count (Conv. Nets and MLP) as two separate methods (even though both use backprop). We will only cover LSTMs and approaches such as Transformers very briefly or later in the module. These methods can be chosen but you'd have to start studying them independently and earlier than when they are seen in class.

The materials on recurrent nets (RNN) are based on an extension of backprop called backprop through time. Those interested in recurrent nets, e.g. for sequence learning, time series prediction, are recommended to have a go at the RNN tutorial early on and experiment with the use of RNNs on their choice of data set in comparison with using standard backprop with a "sliding window" approach (more on sliding windows below). This way you can reduce the risks of having problems later on, too close to the submission deadline.

For those using time series (item (3) above), consider backprop through time (BPTT) compared with standard backprop with a sliding window approach: given a window size on the time series data and an offset, which are hyperparameters, the time series can be transformed into a standard data set for use with backprop. The extra effort of using RNNs, LSTMs and Deep Nets/Transformers should pay off because learning from sequence data is an interesting and relevant problem to tackle and your work has scope to continue as part of your MSc dissertation. The same is true for deep networks based on a stack of RBMs with a softmax layer or an SVM on top). This can be compared with backprop applied to multiple hidden layers, as done in the tutorial using the MNIST dataset, or to a simple SVM. Here, you can investigate the effect of the vanishing gradients problem on having deeper networks, or the benefit of the RBMs in comparison with having only an SVM.

In relation to items (4) and (5), you can use cross-validation (with or without early-stopping) to improve your estimate of the model's generalization performance. In this case, report the average training set and validation set performance and use a network ensemble as your final model. If you are doing model selection, the model hyper-parameters giving the lowest average validation performance should be chosen. It is common in this case to re-train a single network with those hyper-parameters on the entire data set instead of using an ensemble. Training can be stopped when this network's training set performance reaches the best average performance from earlier. With larger data sets, instead of using cross-validation, it is common that the data will be split already into a training set, a validation set also known as a development (dev) set, and a test set. In this case, you'd simply report training set performance and validation set performance when choosing the best model, and finally for the comparison with the other learning method, you'd report test set performance. It is also possible to use in this case early-stopping and boostrapping for model selection, and to compare results with the case where early-stopping is not used.