TDT4171 Ex5

Harald Husum

May 2016

1 Framework Completion

To complete the framework, several additions had to be made. The back propagation function was an empty skeleton in the handed out framework. Functionality for delta calculation and weight updating had to be added. This was a simple matter of implementing the equations specified in the exercise. The train function was also in need of implementing. This was done by propagating and back propagating every pattern pair in the training set once for every specified iteration. An error counting function was implemented. It propagates pairs from a test set without back propagating (and thus not training). It then counts the rate of wrongly ranked pairs.

The data loader lacked some functionality as well. pattern pairings were generated as specified: unequal ratings from the same query ID. They were then added to the training and test lists. The function was modified to train 10 networks on the same patterns to generate the required data.

2 Data Discussion

For my run, I chose to halve the learning rate from the default in the framework. When I experimented with increasing it tenfold, it severely impacted the quality of the learning, so I figured the default value was close to its upper feasible limit. I judged that lowering it slightly would improve quality without hampering performance significantly. I also ran 25 epochs ten times because I had time to waste.

The graphs show a rapid improvement from the initial performance in the first ten epochs. The random initial weights have a lot of room for improvement, and this is exploited here. As the weights come closer to an optimum, progress slows down. The increase in performance on the training set comes to a crawl by the end of epoch 25, but never stops completely. Post epoch 20 something interesting happens on the test set. The success rate stops increasing and slowly starts deteriorating. This continued performance increase on the training set while the opposite is happening to the test set is actually the beginning of over fitting. If we were to let the training run for many more epochs, we can expect the test performance to continue to decrease.

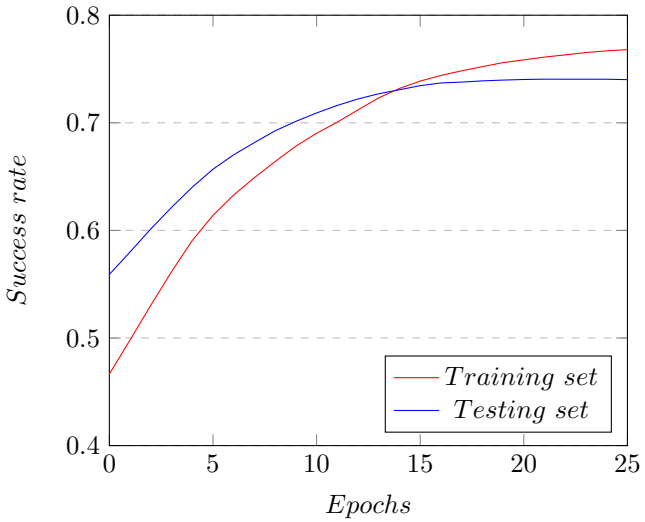

Figure 1: Training set and test set error averages over 10 runs

3 Difficulties

This was a fun exercise, but my progress and enjoyment was slightly hampered by a framework that is neither very pretty, nor well documented. It is not critically bad, but it has lots of room for improvement. For instance, using TODOs both as general specifications for method functionality and as instructions for code to be implemented on specific lines, creates confusion. TODOs is almost always interpreted as "insert code at this exact line", and documentation should be rewritten to comply with this. For general specifications, like the one at the top of runRanker, a simple comment will suffice.

To fit the exercise specification and avoid confusion, the general "previous/current" naming scheme should be changed to "a/b". In backprop_skeleton, "inputActivation" should get an "s" added to the end of it to make clear that it is a list. As should the "dataInstance" lists in dataLoaderSkeleton.