Case-based reasoning
TDT4173- Assignment 4

Øyvind Aukrust Rones, NTNU

April 13, 2018

1 Theory

1.1

The idea of deep learning is that learning is done through a long cascade of composite nonlinear processing units. What separates this from shallow learning is the length of this cascade, where no clear threshold between these have ever been set universally. In the case of artificial neural networks this would amount to a model with several hidden layers, letting more features be learned. For example, in section 2, the XOR-network could be considered a shallow learner, while the digit-learner could be considered a deep learner.

1.2

K-NN classifies samples by looking at their k nearest neighbors according to a given distance metric, while SVM's try to find a decision border that separates clusters maximally. The latter works similar to decision trees in that they both split a set of clusters, but instead of finding a single (or more) multidimensional boundary, decision trees split the clusters by a single feature at a time. Deep learning seeks to approximate the decision boundary as a mathematical function.

K-NN automatically implements non-linear behavior, and given that non-linear units is used, this is also the case for deep learning. SVM's on the other hand have to rely on kernel functions for this, which might require more knowledge of the process in question. Decision trees are very sensitive to nonlinear problems and should be used with care in such cases. As SVM's use support vectors they handle outliers quite well as opposed to k-NN and deep learning. K-NN and SVM's also scales badly. Decision trees however is very effective for well clustered data with well chosen features.

1.3

Ensemble methods should be used when the amount of available data is low or the data tends to be noisy. Additionally, it could also be used when modeling all the aspects of a process with a single model is difficult. Examples of ensemble learning techniques:

Bagging implements a set of similar models and takes a mean of all their predictions.

Boosting weights the observations depending on their earlier classifications.

Stacking adds a seconds learner that combines the outputs of several different learners.

2 Programming

2.4

Figure 1 shows the smallest possible MLP that can solve the XOR-problem. There has to be two neurons in the hidden layer to capture the asymmetrical nature of the problem. A single neuron in place of this would not be able to differentiate between two high inputs and a single high input.

Figure 2 shows the average loss per epoch over a total of 10000 epochs.

2.5

The confusion matrix for the handwritten digit classification is here shown for a test set of 450 samples. The network does a pretty good job of classifying the digits with a success rate of 0.9. The network was in this case composed of two hidden layers with 64 and 32 nodes respectively. A hot one encoding scheme was implemented as well so an output layer of 10 nodes were used. The learning rate was set to 0.01 and the network was trained for 100 epochs.

$$\begin{pmatrix}
39 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 42 & 1 & 0 & 2 & 0 & 0 & 0 & 2 & 0 \\
0 & 2 & 33 & 2 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 37 & 0 & 0 & 0 & 0 & 3 & 2 \\
0 & 0 & 0 & 0 & 46 & 0 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 & 41 & 1 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 & 0 & 43 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 3 & 0 & 0 & 47 & 0 & 0 \\
0 & 3 & 2 & 1 & 0 & 0 & 0 & 0 & 36 & 0 \\
0 & 0 & 0 & 3 & 0 & 0 & 0 & 2 & 2 & 41
\end{pmatrix}$$

2.6

After implementing dropout, the network did marginally better, correctly classifying a single sample, with a success rate of 0.902, yet certain digits were more prone to being misclassified in this case. Unexpectedly, no significant improvement was found from dropout. The dropout rate used here was 0.1. All other

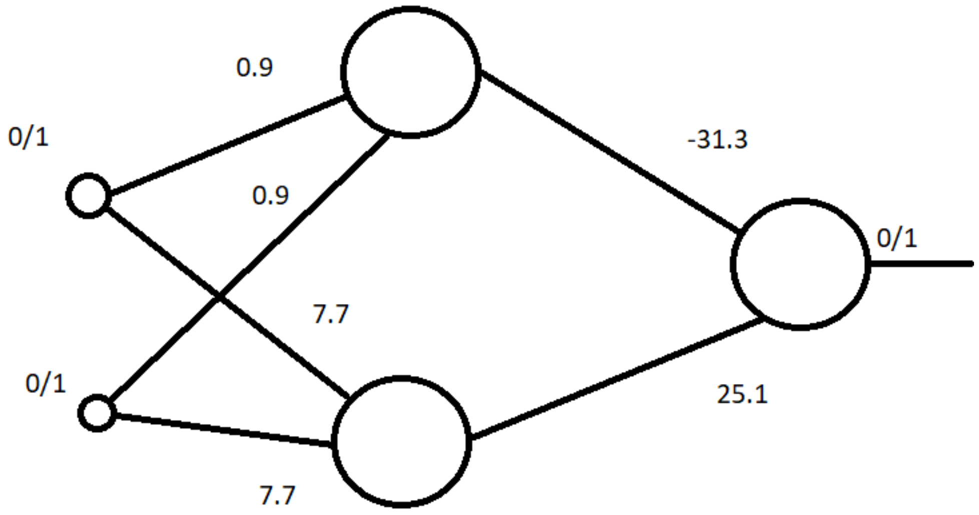

Figure 1: The minimal XOR-network found by my program.

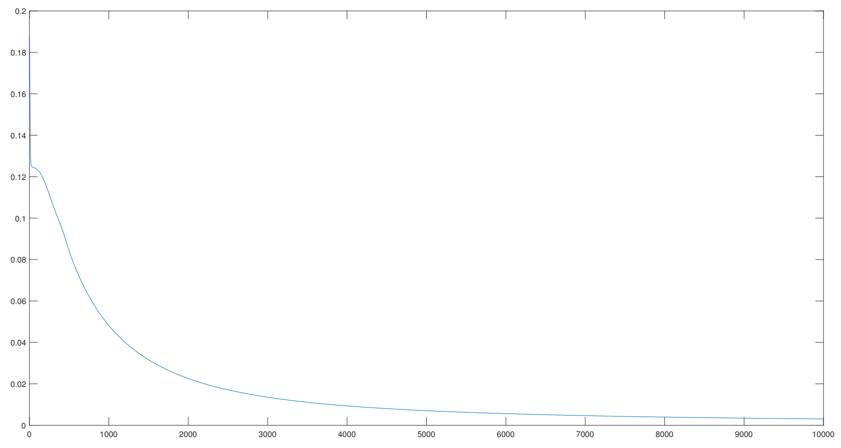

Figure 2: The minimal XOR-network found by my program.

parameters were as before.

$$\begin{pmatrix}
39 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 43 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
0 & 1 & 35 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 2 & 37 & 0 & 0 & 0 & 0 & 1 & 3 \\
0 & 0 & 0 & 0 & 42 & 0 & 0 & 5 & 2 & 0 \\
0 & 0 & 0 & 2 & 0 & 42 & 0 & 0 & 1 & 1 \\
0 & 2 & 0 & 0 & 0 & 0 & 43 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 2 & 0 & 0 & 49 & 0 & 0 \\
0 & 1 & 3 & 0 & 0 & 0 & 0 & 0 & 37 & 1 \\
0 & 0 & 0 & 2 & 0 & 2 & 0 & 2 & 3 & 39
\end{pmatrix}$$