T-61.181 Time in self-organizing maps
Kangas models

Jussi Pakkanen
jussi.pakkanen@hut.fi

11th October 2002

1 Introduction

This paper describes several methods used to analyze time-dependant data with the SOM. These methods, such as trajectories and data concatenation are explained in some detail. These explanations are then followed by results and examples of usage. The results show that the SOM can be used for time-dependant data analysis, although it was not originally designed for this task.

2 Trajectories

The basic self-organizing map does not use any time information on the data points. It sees the data vectors merely as independent data points in space. One could therefore assume, that the basic SOM is unsuitable for time-dependant data analysis. However this is not the case.

When measuring time-dependant variables, the vectors measured at different times differ from each other, in some cases dramatically. The SOM is quite good in finding differences and similarities in data. Therefore it is quite probable that the samples which depict similar points in time become clustered on the map. Suppose you have a periodic system with a period of t seconds. Now the samples that are separated by multiples of t should map to SOM nodes that are close to each other.

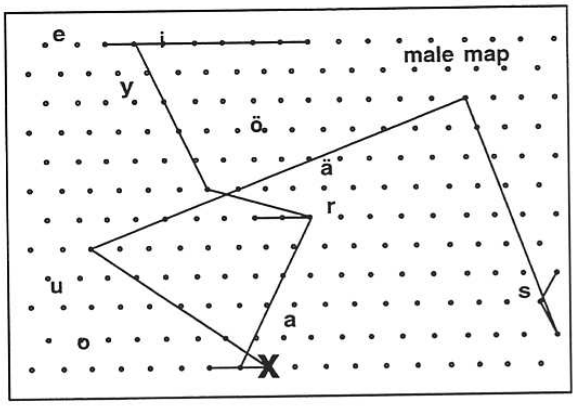

Figure 1: A SOM of phonemes and a corresponding trajectory. The word said in this figure was /sa:ri/.

This insight offers an effective way of using the basic SOM to analyze time-dependant data. First you train the SOM normally. Then you feed your data that you want to analyze to the SOM one by one. Then you check how the BMU moves around on the map. An example can be seen in Figure 1. It can be seen that the trajectory goes through the letters needed to form the word in question. Unfortunately it also wanders on other areas.

2.1 Diagnosing speech disorders

In his Ph.D. thesis [1], Jari Kangas describes a system which can be used to diagnose phonation disorders. The system consists of a SOM which has been trained with speech samples from various speakers. This was then used to analyze new samples. Examples can be seen in Figure 2.

The normal speakers' trajectories remain neatly in one or two nodes the whole time. In contrast the trajectories of the people that have speech problems wander a lot around the map. It is quite easy, even for a non-specialist to look at the trajectories and diagnose possible speech disorders.

It is possible to make an even more precise diagnosis by more careful analysis of the trajectory. In this case it was found that the mean length of the shifts was a good indication of the severity of the speech problem. Shift length means in this case how far from each other successive samples are on

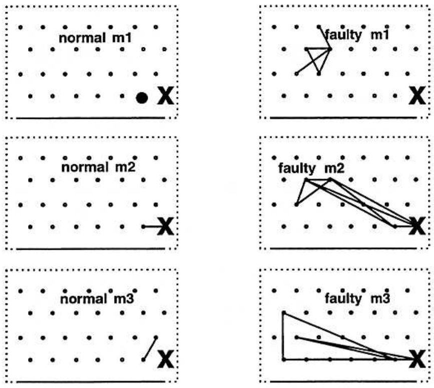

Figure 2: Trajectories of normal speakers (left) and those with speech problems (right).

the map. The distance between two adjacent nodes is defined to be one.

The mean shift length for normal people was 0.35, which implies very little movement. For mildly dysphonic people the mean shift was 0.93 and for severely dysphonic people it was 1.31. Therefore the severity of the speech disorder can be determined from this behaviour. Kangas describes other similar experiments for speech classification but they are not examined here.

3 Time-aware SOMs

Since the basic SOM algorithm does not take time information into account, it must be modified to get better results with time-dependant data. Since modifying the self-organizing algorithm itself would most likely be very hard, most systems use the unaltered SOM as one part and build the time-dependency part around it with various tricks. Here are listed some of these from [2].

3.1 Data averaging

Data averaging works on a simple idea: the network is given a combination of present and past values. This time average is considered to better represent

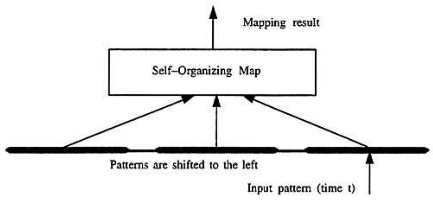

Figure 3: Concatenating data vectors to get a single larger input vector.

the time characteristic of the data.

This method is extremely simple to realize. If you have a time series input that consists of $x[t]$, you don't feed that to the network directly. Instead, you feed it $x_w[t]$, which is calculated like this:

$$x_w[t] = w \cdot x[t] + (1 - w)x_w[t-1], \quad \text{where} \quad 0 < w < 1.$$

As can be seen from the formula, the weight of the past values decreases as time goes on. How fast the system forgets the past depends on the value of w. If it is small, the system changes states very slowly. If it is large, the past plays only a very small role in the input.

3.2 Data concatenation

In data concatenation the SOM is given as input a concatenation of several data vectors. This is illustrated in figure 3. A new data vector is put to the rightmost slot. Its contents are moved one slot to the left as are the next one's and so on. The last value is simply discarded.

When the SOM is trained with this kind of data, it does not know that it consists of several concurrent data vectors. It just tries to find some explanation of the high-dimensional data. The explanation it finds should contain information about the time behaviour of the system in question.

3.3 Averaging responses

This method is a refinement of the data averaging method. First the data is fed into a normal SOM. The output of this map is then fed into a data averager and after this to an another SOM. In effect the second SOM is fed data that describes the first SOM's trajectory.

3.4 Results

The various systems were tested with artificial data. This data consisted of an object moving in a figure-of-eight shape. The task was to correctly determine which way the object was moving in the middle part (the crossing of the figure of eight). This test was done with various amount of noise in the measurement data.

When using a normal SOM the classification error was 89% without noise and decreasing to 72% with maximum noise. Data averaging added very high tolerance against noise. The accuracy was 97% without noise and 96% with maximum noise. The downsize was that the system was very sensitive to the value of w. Very small changes from the optimal value of 0.05 degraded the performance considerably.

Data concatenation increased the the percentages dramatically. Without noise the accuracy was 100% (with 4 concatenated vectors) and with maximum noise it was 99% (with 8 vectors). Response averaging worked also very well. The accuracies were 100% for no noise and 98% for maximum noise.

4 Conclusions

In this paper several methods of analyzing time-dependent data with the SOM was discussed. It was found that the SOM, although not inherently time-oriented, can be used succesfully to analyze time-dependent data. This is achieved by working around and complementing the basic SOM. Results indicate that these methods are quite succesfull and worthy of further examination.

References

[1] Kangas, J., On the analysis of pattern sequences by self-organizing maps, *Ph.D. thesis*, Helsinki University of Technology, 1994

[2] Kangas J., Time-delayed self-organizing maps, is *Proc. IEEE/INNS Int. Joint Conf. on Neural Networks (IJCNN'90)*, 1990, Vol 2., pp. 331-336