MVA '92 IAPR Workshop on Machine Vision Applications Dec. 7-9,1992, Tokyo

A PARALLEL TOPOLOGICAL MAP FOR IMAGE SEGMENTATION

Dinu Scheppelmann, Jochen Frey, Manuela Schäfer, Hans-Peter Meinzer

German Cancer Research Center,
Dept. Medical and Biological Informatics,
Im Neuenheimer Feld, D-6900 Heidelberg, Germany

ABSTRACT

We present a new algorithm for topological feature mapping. It is compared to other known feature mapping algorithms like that proposed by Kohonen or Bertsch. The main difference to known algorithms is the inversion of the learning step. This makes it possible that a neural net can be seen as a cellular automaton. A neurone learns from its neighbours instead of teaching them the new value they have to adapt to. Equivalence of the algorithms is shown. The results of the algorithm are extremly high parallelization and flexibility in neighbourhood definition. This leads to new solutions for image segmentation problems.

INTRODUCTION

Artificial neural networks are providing the possibility of alternative solutions to many old problems. The unsupervised self-organizing networks with their property of learning and extracting important features of a given sensory input, has especially shown to be useful for segmentation of images and for pattern recognition. A very popular and often used self-organizing map is the Topological Feature Map (TFM) proposed by Kohonen [2].
However, TFMs are very slow in the learning phase. The reason for this is the typically sequential learning algorithm of neural nets. Each input is presented to the net one after the other and the net adapts accordingly.
There is much work done to assure convergence in the learning phase for a TFM regarding the learning factor [4] and the neighbourhood function [3]. These papers suggest that there is a lower limit for the computational steps in the learning phase of a TFM. In this paper these problems will not be discussed.
We propose an algorithm which shows the same results and which still has some of the drawbacks of the original TFM learning algorithm. However, the new algorithm offers a more 'holistic' view to the learning procedure and allows a tremendous speed up in computation due to a very easy parallelization. In addition to this, it allows for special applications some very simple but useful modifications to the features of the original TFM. In the remainder of the paper, we'll give a summary of Kohonen's [2] algorithm and the modifications proposed by Bertsch [1]. Then we will show how the neighbourhood adaption process can be inversed.

KOHONEN'S ALGORITHM

Kohonen [2] proposes a new representation of complex empirical data by an 'adaptive physical system'. It consists of an array **A** of formal neurones. The neurones or units receive random input samples from a vector space **V**. These input samples are mapped onto **A**. Each input $\mathbf{v} \varepsilon \mathbf{V}$ is represented as a vector v(s) where **s** denotes the time step. Each unit $\mathbf{a} \varepsilon \mathbf{A}$ has an internal state a(s) represented as a vector of the same dimension as v(s). It is also labelled with a unique number.

The adaptation of **A** to **V** is achieved by the following steps:

Step 1: Initialize every a(0) with small random numbers.

Step 2: A $v \varepsilon V$ is picked by random. Find the unit a_i where $a_i(s)$ is most similar to v(s). Call this unit a_i 'selected unit'.

Step 3: The internal states $a_i(s)$ of this selected unit and the units in its topological neighbourhood $N(a_i)$ are changed in a way that they become more similar to v(s) (according to (1)).

Step 4: Repeat by going to step 2.

The similarity between a units' state a(s) and input v(s) is measured by the Euclidean distance between the two vectors.
The adaptation rule for the units in the neighbourhood N(a) is as follows:

$$a_j(s+1)=a_j(s)+\alpha(s)[v(s)-a_j(s)] \quad for\ j \epsilon N(a_i)$$
$$a_j(s+1)=a_j(s) \qquad otherwise \qquad (1)$$

where $\alpha(s)$ is a slowly decreasing function of time. The optimal selection of $\alpha(s)$ and N(a) will not be discussed in this paper.

BERTSCH'S MODIFICATIONS

Bertsch [1] proposes first to find the most similar unit for every input $v \varepsilon V$ and to do the adaptation of all the units thereafter.
Instead of updating each unit by the individual input vectors one ofter the other all vectors which point to one unit are averaged and the unit is then updated by this mean value. He proves that this update-algorithm shows the same results as Kohonen's original algorithm. The adaptation of **A** to **V** is now achieved by the following steps:

Step 1: Initialize every a(0) with small random numbers.

Step 2: Initialize a counter $c_i=0$ and a sum s_i for every a_i.

Step 3: Select an input $v_j \varepsilon V$.

Step 4: Find the most similar unit a_i for every v_j.

Step 5: Increment c_i by 1 and s_i by v_j.

Step 6: Proceed with step 2 until the last input $v \varepsilon V$ is reached.

Step 7: Update the neighbourhood N(a) of every a_i according to (2).

Step 8: Repeat by going to step 2.

$$a_j(s+1)=a_j(s)+\alpha(s)[\frac{s_i}{c_i}-a_j(s)] \quad if\ j \epsilon N(a_i),\ c_i>0 \qquad (2)$$
$$a_j(s+1)=a_j(s) \quad otherwise$$

Now there is one update-step for the whole map but the actual update still is a sequential process because every unit influences its neighbours. As every input should have the same influence on the map, there are certain restrictions for $\alpha(s)$ (see [1]) to compensate this drawback.

THE NEW ALGORITHM

From teaching to learning: The new algorithm mainly gives a different view on the learning process. The algorithms mentioned above both share two main features:
1) Learning remains even after Bertsch's modifications a highly sequential task because each unit has to be updated one after the other.
2) During adaptation of a unit to new input values, the unit 'teaches' or forces the units in its topological neighbourhood to adapt to the new input (Fig.1a). This means that, for one complete update of the map, the internal state of

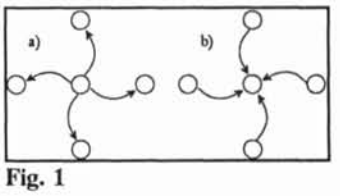

Fig. 1

each neurone has to be changed repeatedly. However, it is possible that each unit learns from its neighbours (Fig.1b). From Fig.2 you can see that the neighbourhood, the unit learns from, is the same as it used to teach. This incorporates, that there's only a single change in the neurone's internal state during a complete update of the map. This doesn't only allow a very high degree of parallelization but also takes the restrictions from $\alpha(s)$ because the weights of the input vectors are not influenced by it.

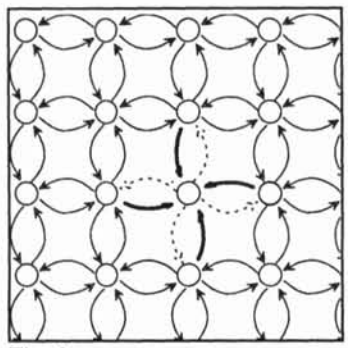

Fig. 2

The algorithm: The steps of the proposed algorithm are almost identical to Bertsch. Only step 7 is slightly modified:

Step 7: Update every $a_i \varepsilon A$ according to (3).

$$a_i(s+1)=a_i(s)+\alpha(s)[u_i(s)-a_i(s)] \qquad (3)$$

where

$$u_i(s)=\frac{\sum_{j\varepsilon N(a_i)} s_j(s)}{\sum_{j\varepsilon N(a_i)} c_j(s)} \quad for \sum c_j>0 \qquad (4)$$

$$u_i(s)=a_i(s) \quad otherwise$$

Here $\alpha(s)$ still is a slowly decreasing function of time. As the update of the whole map occurs simultaneously there are no restrictions for $\alpha(s)$ like there are in Bertsch's algorithm.

Reasons: Every state $a_i(s+1)$ is determined by the $s_j(s)$ and $c_j(s)$ of its neighbourhood (see Fig.2). For simplicity reasons we will now assume, that every neighbour of a_i has the same influence. This leads us to (5).

$$a_i(s+1)=\frac{1}{N}\sum_{j\varepsilon N(a_i)} a_i(s)+\alpha(s)[m_j(s)-a_i(s)] \qquad (5)$$

where N equals the number of units in $N(a_i)$ and $m_j(s)$ is the mean value of all $v_j(s)$ (see (6))

$$m_j(s)=\frac{s_j(s)}{c_j(s)} \qquad (6)$$

Some transformations later we get (7) and (8)

$$a_i(s+1)=a_i(s)(1-\alpha(s))+\alpha(s)\frac{\sum_{j\varepsilon N(a_i)} m_j(s)}{N} \qquad (7)$$

$$a_i(s+1)=a_i(s)+\alpha(s)[\frac{\sum_{j\varepsilon N(a_i)} m_j(s)}{N}-a_i(s)] \qquad (8)$$

As we don't want every neighbouring unit having the same weight but being weighted corresponding to its c_i, we use (4) instead of the term that represents the mean value.
We can easily extend this model by adjustable weights to satisfy other demands, e.g. to introduce a neighbourhood interaction function (9) [3] instead of a neighbourhood interaction set.

$$u_i(s)=\frac{\sum_{j\varepsilon N(a_i)} w_{ij}s_j(s)}{\sum_{j\varepsilon N(a_i)} c_j(s)\sum_{j\varepsilon N(a_i)} w_{ij}} \quad for \sum c_j>0 \qquad (9)$$

$$u_i(s)=a_i(s) \quad otherwise$$

DISCUSSION

Parallelization: Finding the best match for every $\mathbf{v}\varepsilon\mathbf{V}$ is an inherently sequential task. Still it could be done by several processors in parallel. The

following update of the map had to be done strictly sequential caused by the interference of the units with neighbouring units. This serious drawback has been removed. Now the update of each unit can be done simultaneously.
Regrettably we do not have the hardware to use all the new features of the algorithm but we found it extremly helpful for the implementation in our APL-prototyping environment.

Flexibility: The algorithm is highly flexible regarding the neighbourhood interaction. There are several possibilities to define the neighbourhood, the neighbourhood inter-action function and the mean value of the neighbouring units. As we are doing image segmentation, we found formula (7) extremly helpful, because we are not interested in the probability density distribution function of **V** but in the different objects. By using (7) we have unified all pixels of large objects in one instead of several units.

Speed: As mentioned above we do not have parallel hardware so the algorithm was running only two to four times faster, depending on the implementation and hardware environment.

Conclusion: The new algorithm has several advantages compared to Kohonen's and Bertsch's as mentioned above. However the optimal $\alpha(s)$-function and the optimal neighbourhood selection, to assure conver-gence of the map, are still unsolved problems.

REFERENCES

[1] Bertsch H., Dengler J. (1987) Klassifizierung und Segmentierung medizinischer Bilder mit Hilfe der selbstlernenden topologischen Karte. In: Paulus E. (ed) 9. DAGM-Symposium Mustererkennung. Springer Informatik Fachberichte 149. Springer, Berlin Heidelberg New York, pp 166-170

[2] Kohonen T. (1982) Clustering, taxonomy and topological maps of patterns. Sixth International Conference on Pattern Recognition, Munich, Germany, October 19-22, 1982, pp 114-128

[3] Lo Z.-P., Bavarian B. (1991) On the rate of convergence in topology preserving neural networks. Biological Cybernetics 65, Springer Verlag 1991, pp 55-63

[4] Ritter H., Schulten K. (1988) Convergence Properties of Kohonen's Topology Conserving Maps: Fluctuations, Stability and Dimension Selection. Biological Cybernetics 60, Springer Verlag 1988, pp 59-71