ESANN'1993 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 7-8-9 April 1993, D-Facto public., ISBN 2-9600049-0-6, pp. 229-234

An Intuitive Characterization for the Reference Vectors of a Kohonen Map

Varfis A. and Versino C.

CEC - Joint Research Centre, Ispra Establishment
Institute for System Engineering and Informatics (ISEI).
Neural Network Laboratory - TP 361, 21020 ISPRA (VA), Italy.

Abstract. The weight updating formula of Kohonen's algorithm may be viewed as the derivative of an objective error function when the input data are generated by a discrete probability distribution. We have derived from this interpretation a simple appealing characterization for the reference vectors layout at the end of the learning phase. That characterization is used for defining a convergence index for Kohonen's algorithm, and for proposing a batch version of it. A first set of experiments with the batch approach is discussed.

1. Introduction

Although the practical effectiveness of the Kohonen map connectionist model [1] has been demonstrated across a wide spectrum of applications [2][3], this approach may be criticised for its lack of theoretical foundations. There have been a few endeavours for relating the algorithm to stochastic approximation theory [2][4], but they have yielded only partial results. One of these establishes a kind of potential function *if we assume the input data to be generated by a discrete probability distribution.* The restriction about the functional accounts for the fact that it has discontinuities in its derivatives, thereby preventing the use of most results about the convergence of stochastic approximation methods. Nevertheless, since the functional is still almost surely differentiable, it is natural to look at the zero values of its derivatives. In this paper we investigate how the related equations might be exploited to gain more insight into the mechanisms of Kohonen's algorithm.

2. An Objective Function

In this section the connectionist algorithm will be briefly described and the candidate potential function will be derived. We will use the same notations as Kohonen [1][4] for easier cross-reference.

2.1. Kohonen Map

Let us consider a fully connected feedforward network with two layers of units. The input layer is fed with a feature vectors $x = x(t) \in S \subset \Re^n$. The formal neurons in the output layer are organized in a two-dimensional *lattice* (or *grid*) and are referred to by their position vector $r_{i=(i1,i2)}$, $1 \le i \le N$. A topology is defined on the grid by

means of some distance $d(\boldsymbol{r}_i,\boldsymbol{r}_j) = d_{ij}$, like e.g. the Euclidean or Manhattan distance ($d_{ij} = \max\{|i_1 - j_1|, |i_2 - j_2|\}$). Each output cell also represents, through its fan-in weight vector, a variable *reference* (or *codebook*) vector $\boldsymbol{m}_i(t) \in \Re^n$.
At step t, either during the learning or the retrieval phase, the squared Euclidean distance d between current pattern $\boldsymbol{x}(t)$ and each reference vector $\boldsymbol{m}_i(t)$ is computed:

$$d(\boldsymbol{x}(t),\boldsymbol{m}_i(t)) = \| \boldsymbol{x}(t) - \boldsymbol{m}_i(t) \|^2 = \sum_{l=1}^{n}[x_l(t) - m_{il}(t)]^2 \tag{1}$$

Let us denote by $c \in \{1,..,N\}$ the index of the best matching reference vector:

$$c(\boldsymbol{x}) = \arg\min_i \| \boldsymbol{x}(t) - \boldsymbol{m}_i(t) \|^2 \tag{2}$$

A step of the learning algorithm consists of presenting an input pattern $\boldsymbol{x}(t)$ and updating every reference vector $\boldsymbol{m}_i(t)$ proportionally to the *interactivity* $h(\boldsymbol{r}_i,\boldsymbol{r}_c) = h_{ic}$ between unit $\boldsymbol{r}_i(t)$ and the *winner* unit $\boldsymbol{r}_c(t)$:

$$\boldsymbol{m}_i(t) \longleftarrow \boldsymbol{m}_i(t) + \varepsilon(t) h_{ic}[\boldsymbol{x}(t) - \boldsymbol{m}_i(t)] \tag{3}$$

The interactivity h is often a positive decreasing function of the grid's distance d. Normally, the *learning rate* $\varepsilon(t)$ decreases during the learning process. This holds true for h as well, but in the following sections *we will only consider a stabilized h function corresponding to the final steps of the learning phase.*

2.2. An Error Functional

In order to interpret Kohonen's algorithm (3) in terms of derivative of an objective error function, *we will assume that the input data have been generated by a discrete probability distribution.* Typical examples are problems where generalization over unseen cases is not included, so that the available finite training sample S - of size T - supports the whole probability distribution.
Assuming - for simplicity - equiprobability of cases $\boldsymbol{x}$, we can deduce from (3) that the *average change* of $\boldsymbol{m}_i$ on a single learning step is:

$$\overline{\Delta \boldsymbol{m}_i} = \frac{\varepsilon}{T} \sum_{\boldsymbol{x}\in S} h_{ic(\boldsymbol{x})}[\boldsymbol{x} - \boldsymbol{m}_i] \tag{4}$$

For the complete set of weights $\boldsymbol{m}$, let us define the *functional* $V^h(\boldsymbol{m})$:

$$V^h(\boldsymbol{m}) = \frac{1}{2T} \sum_{i=1}^{N} \sum_{\boldsymbol{x}\in S} h_{ic(\boldsymbol{x})}[\boldsymbol{x} - \boldsymbol{m}_i]^2 \tag{5}$$

Almost surely (with respect to the Lebesgue measure in $\Re^n$), we can compute:

$$\frac{\partial V^h(\boldsymbol{m})}{\partial \boldsymbol{m}_i} = -\frac{1}{T} \sum_{\boldsymbol{x}\in S} h_{ic(\boldsymbol{x})}[\boldsymbol{x} - \boldsymbol{m}_i] \tag{6}$$

A straightforward computation of the $V^h(\boldsymbol{m})$ derivatives has been possible because, for a finite set S, it is (almost surely) possible to define an open sphere O_i about each $\boldsymbol{m}_i$, such that for every $\boldsymbol{x} \in S$, $c(\boldsymbol{x})$ does not change when $\boldsymbol{m}_i$ moves in O_i. As a consequence, the usual implicit dependency between $c(\boldsymbol{x})$ and the $\boldsymbol{m}_i$'s has not to be taken into account for the derivatives' computation.

Since $\overline{\Delta \boldsymbol{m}_i} = -\varepsilon \frac{\partial V^h(\boldsymbol{m})}{\partial \boldsymbol{m}_i}$, it appears that Kohonen's learning formula (3) may be

seen as a stochastic gradient descent method to minimize $V^h(\boldsymbol{m})$. Unfortunately, this does not mean that a successful stochastic approximation method for minimizing (5) has been derived. Beyond the small problem of $h(t)$ values that decrease during the learning phase - we could be content with a true potential $V^{h_\tau}(\boldsymbol{m})$ for the *target* interactivity h_τ - the dependency between $c(\boldsymbol{x})$ and the $\boldsymbol{m}_i$'s keeps from applying the available theorems relative to the convergence of stochastic approximation methods.
In spite of these shortcomings, the fact is that many users tend to feel more comfortable when numeric criteria are made available to them. The main goal of the following sections consists of showing how equations (5) and (6) may be exploited for gaining more insight into Kohonen Map's data representation.

3. Reference Vectors Layout

We will show that a Kohonen map should organize in such a way that any reference vector is a weighted mean of the elements of $\boldsymbol{S}$, with weighting coefficients that correspond to the target interactivities h_τ.

3.1. A "Characteristic" Equation

Local minima for equation (5) correspond to zero values for the derivatives (6). Thus it is possible to obtain an intuitive characterization of "ideal" Kohonen map's reference vectors at the end of the learning phase:

$$\mu_i = \frac{1}{\sum\limits_{\boldsymbol{x}\in \boldsymbol{S}} h_{ic(\boldsymbol{x})}} \sum_{\boldsymbol{x}\in \boldsymbol{S}} h_{ic(\boldsymbol{x})}\, x \tag{7}$$

For the final layout of a Kohonen map, *the codebook vector associated to a neuron $\boldsymbol{r}_i$ should be close to the barycentre of the input set $\boldsymbol{S}$, weighted by the interactivity coefficients $h_{ic(\boldsymbol{x})}$*. Two examples follow:

- K-means: $h_{ic} = \delta_{ic}$, and $V^{\delta}(m) = \frac{1}{2T}\sum\limits_{i=1}^{N}\sum\limits_{c(\boldsymbol{x})=i}[\boldsymbol{x}-\boldsymbol{m}_i]^2$. For the traditional reconstruction error criterion used for vector quantization, we recover the known fact that the K-means vectors $\boldsymbol{m}_i$ should be close to the centroid of their *receptive fields* $\mathrm{RF}_i = \{\boldsymbol{x}, c(\boldsymbol{x}) = i\}$.

$$\mu_i = \frac{1}{\mathrm{Card}(\mathrm{RF}_i)} \sum_{\boldsymbol{x}\in \mathrm{RF}_i} \boldsymbol{x} \tag{8}$$

- $h_{ij} = \mathbf{I}(d_{ij} \le \Delta)$, where $\mathbf{I}$ is the indicator function: Let us call *extended receptive field* ERF_i for neuron $\boldsymbol{r}_i$ the subset of $\boldsymbol{x} \in \boldsymbol{S}$ leading to the selection of a neuron $\boldsymbol{r}_j$ within a distance Δ of $\boldsymbol{r}_i$:

$$\mathrm{ERF}_i = \{\boldsymbol{x}, \exists j, c(\boldsymbol{x}) = j \text{ and } d_{ij} \le \Delta\} = \bigcup_{d_{ij}\le\Delta} \mathrm{RF}_j . \tag{9}$$

Here again reference vectors $\boldsymbol{m}_i$ should end up close to the centroid of their ERF_i.

$$\mu_i = \frac{1}{\mathrm{Card}(\mathrm{ERF}_i)} \sum_{\boldsymbol{x}\in \mathrm{ERF}_i} \boldsymbol{x} \tag{10}$$

3.2. A Convergence Index

The characteristic equation (7) may be used for defining in a natural way convergence indexes for Kohonen's algorithm. For example, one may compute the mean value, over the complete set of codebook vectors, of the square distance between actual $\boldsymbol{m}_i$ and ideal μ_i reference vectors:

$$I(t) = \frac{1}{N} \sum_{i=1}^{N} \left\| \mu_i - \boldsymbol{m}_i(t) \right\|^2 \qquad (11)$$

That index I has been successfully tested on a real problem [3], thereby supporting the interpretation of Kohonen map's organization according to formula (7). In [3], 228 14-dimensional input patterns were mapped onto an 8 by 8 grid (Card(S)=228, n=14, N=64). d_{ij} corresponded to the Manhattan distance, and $\mathrm{h}_{ij} = \mathbf{I}(\mathrm{d}_{ij} \leq \Delta)$, like for the second example of section 3.1. For ten experiments differing only by the initial random weight settings, and at the end of the learning phase, the index I has been computed for Δ=0 (K-means), for Δ=1 (the actual target interactivity) and for Δ=2. The averages over these ten experiments are displayed in table 1, where it appears that the indexes corresponding to the target interactivity $\mathrm{h}_{ij} = \mathbf{I}(\mathrm{d}_{ij} \leq 1)$ are smaller by about one order of magnitude.

$\mathrm{h}_{ij} = \mathbf{I}(\mathrm{d}_{ij} \leq 0)$	$\mathrm{h}_{ij} = \mathbf{I}(\mathrm{d}_{ij} \leq 1)$	$\mathrm{h}_{ij} = \mathbf{I}(\mathrm{d}_{ij} \leq 2)$
0.177	0.015	0.112

Table 1.

4. Batch Kohonen Map Learning

One of the intriguing consequences of equation (7) stems from the possibility of deriving a kind of "batch" version of Kohonen's algorithm, in a way similar to the LBG algorithm [5] for computing K-means. One proceeds as follows:

```
1) Initialization of the reference vectors $m_i(t_0)$
2) $\forall$i, compute $\mu_i(t)$ from (7).          These values depend of the $m_i(t)$!
3) If $\forall$i, $\mu_i(t) = m_i(t)$, then stop.
4) $\forall$i, $m_i(t+1) \leftarrow \mu_i(t)$.          Weighted means become reference vectors
5) Goto 2.
```

If such a process eventually converges, its index value (11) is obviously $I = 0$. In other words, the partial derivatives (6) are all equal to zero, and we are in presence of a (local) minimum for the functional $\mathrm{V}^h(\boldsymbol{m})$.

The first experiments with the proposed algorithm have yielded mixed results. On one hand, some tests on a real clustering task [3] have shown that the algorithm may prove to be extremely sensitive to the initial organization of the reference vectors. On the other hand, the batch version of Kohonen's algorithm performed well for the traditional toy problem were the elements of the (finite) training sample S have been drawn from the uniform distribution on the unit square of $\Re^2$.

4.1. Real-life Data

The above algorithm essentially failed for the real dataset, even for initial reference vectors $\boldsymbol{m}_i(t_0)$ issued from a fairly pushed on traditional Kohonen map training session. Either no convergence was observed, or the codebook set collapsed into a few multiple points. However, when the "initial" $\boldsymbol{m}_i(t_0)$ vectors were taken from a "completed" Kohonen map training session, then in many cases a single ensuing loop let the algorithm converge. As a consequence, and quite generally, our suggestion is to *try for a fine tuning of the reference vectors of a trained Kohonen map, by inspecting whether the substitution of the* $\boldsymbol{m}_i$ *by the* μ_i *modifies any of the N receptive fields* RF_i. When it leads to a fixed reference vector set, computing the single batch loop constitutes a straightforward way for removing residual random fluctuations from a given solution.

4.2. Synthetic Data

For the experiments with synthetic data, 100 vectors uniformly distributed on the unit square of $\Re^2$ were to be mapped onto an 8 by 8 grid (Card(S)=100, n=2, N=64). Here again we used $h_{ij} = \mathbf{I}(d_{ij} \leq \Delta)$, with a target range Δ=1 and d_{ij} corresponding to the Manhattan distance. Since for n=2 it is possible to watch the map's organization by plotting the weights on the plane (the "Magic TV"), we could verify for the following experimental settings that the proposed algorithm was capable of providing a suitable organization of the reference vectors:

- The initial codebook $\boldsymbol{m}(t_0)$ comes from a very short Kohonen map training session, that is a couple of epochs with Δ = 4 or 3. From the resulting rough global organization the batch version with Δ = 1 quickly produces the expected approximately regular grid.
- The batch algorithm may also be used from scratch (with the customary initialization $\boldsymbol{m}_i(t_0)$ = (0.5, 0.5) + random noise). For Kohonen's algorithm, it is advisable to foster the reference vector expansion with a high Δ range during the early learning phase, and then let Δ decrease towards its target value. Since the new algorithm is well defined for any fixed Δ value, it has been possible to proceed in a similar way. Accordingly, we successively computed two epochs of the batch version for Δ = 5, 4, 3, 2 and 1, constraining the reference vectors to centroids of extended receptive fields of narrower and narrower range, to finally reach an organized state.

While the above-described experiments show the viability of the proposed algorithm for the simple task at hand, they do not provide clear indicators for comparing the new algorithm with the traditional one. For that purpose some numeric criterion is needed - not $V^h(\boldsymbol{m})$ or I (11)! - for measuring the quality of a Kohonen map. We have used the *topographic product P*, which is a "quantitative measure for the preservation of neighbourhood relations in maps" [6]. For the ensuing discussion, we just need to know that decreasing $|P|$ values correspond to improving performances.

We trained a Kohonen map over ten epochs. The first five, with initially high Δ values, were aimed at spreading rapidly the reference vectors over the unit square and reaching a partial organization. The last five epochs were computed with Δ = 1 (and a decreasing learning rate ε). At the beginning of each of these five epochs, the set of current reference vectors was saved and fed to the batch algorithm for initializing it. The third column of table 2 displays the topographic product values that have been obtained for a single batch cycle after the five successive initializations. They are about two times smaller than the corresponding *P* values, epoch after epoch, for Kohonen's algorithm (second column of table 2).

Epoch	*P: Kohonen*	*P: Batch*
6	-0.00838	-0.00434
7	-0.00676	-0.00359
8	-0.00569	-0.00327
9	-0.00530	-0.00315
10	-0.00513	-0.00293

Table 2.

5. Conclusion

The functional $V^h(\boldsymbol{m})$ cannot strictly speaking be used in the frame of stochastic approximation theory, even for a fixed $h(t)$, because its derivatives have discontinuities along the borders of adjacent receptive fields. Nevertheless, the "characteristic" equation (7) which is obtained by zeroing the derivatives of $V^h(\boldsymbol{m})$ (6) has an appealing simple interpretation in terms of weighted means (7) or barycentres of extended receptive fields (9)(10). Furthermore, the convergence index (section 3.2.) and the batch version of Kohonen's algorithm (section 4.) which are derived from equation (7) have confirmed the relevance of the weighted mean interpretation for Kohonen's map reference vectors.

6. References

1. T. Kohonen: The Self-Organizing Map. Proceedings of the IEEE, 78, N° 9, 1464-1480 (1990).
2. H. Ritter, T. Martinetz, K. Schulten: Neural Computation and Self-Organizing Maps. An Introduction. Addison-Wesley, Reading (1992).
3. A. Varfis, C. Versino: Clustering of socio-economic data with Kohonen Maps. Neural Network World, 2, N° 6, 813-834 (1992)
4. T. Kohonen : Self-Organizing Maps: Optimization Approaches. In: Artificial Neural Networks, Kohonen T., Mäkisara K., Simula O., Kangas J. (Eds), Elsevier Science Publishers B.V. (North-Holland), 981-990 (1991).
5. Y. Linde, A. Buzo, R.M. Gray: An algorithm for vector quantizer design. IEEE Trans. Comm., 28, 84-95 (1980).
6. H-U. Bauer, K. Pawelzik, T. Geisel: A Topographic Product for the Optimization of Self-Organizing Feature Maps. In: Advances in NIPS 4, J. Moody, S. Hanson, R. Lippman (Eds), Morgan Kaufmann, 1141-1147 (1992).