Fast Learning Approach Using Self Organizing Map

Sagheer, Alaa
Department of Intelligent Systems, Kyushu University

Tsuruta, Naoyuki
Department of Electronics Engineering and Computer Science, Fukuoka University

Taniguchi, Rin-ichiro
Department of Intelligent Systems, Kyushu University

Arita, Daisaku
Department of Intelligent Systems, Kyushu University

他

出版情報：IEICE Technical Report, Pattern Recognition and Media Understanding. 105 (613), pp.25-30, 2006-02. 電子情報通信学会
バージョン：
権利関係：

Fast Learning Approach Using Self Organizing Map

Alaa SAGHEER†, Nayouki TSURUTA‡, Rin-Ichiro TANIGUCHI†
Daisaku ARITA†, Sakashi MAEDA‡

†Department of Intelligent Systems, Kyushu University
6-1 Kasuga-Koean, Kasuga, Fukuoka 816-8085, JAPAN

‡Department of Electronics Engineering and Computer Science, Fukuoka University
8-19-1, Nanakuma, Jonan, Fukuoka 814-0180, JAPAN

E-mail: †{alaa, rin, arita}@limu.is.kyushu-u.ac.jp, ‡{tsuruta, maeda}@tl.fukuoka-u.ac.jp.co.jp

Abstract The Self Organizing Map (SOM) is one of the most widely used neural network paradigm based on unsupervised competitive learning. However, the learning algorithm introduced by Kohonen is very slow when the size of the map is large. This slowness is caused by seeking about the best node among "all" the map nodes which tunes to "each" input sample. In this paper, a novel fast learning SOM algorithm is proposed. Exploiting a new strategy, the new algorithm runs by concerning "only" about the nodes which are aligned around principal components and neglects the rest of nodes which already include less information [1]. Experimental results are reported at the end of this paper. Two data sets are utilized to illustrate the proposed algorithm. Under same experiment conditions, it is shown here that the computation time is reduced to $O(\log N)$ instead of $O(N)$. Also our method computation time is less than that of FDCT by 6 times under same experimental conditions.

Key words Self Organizing Map, Lip-Reading Systems, Principal Components Theory

1. Introduction

In many situations in pattern recognition, machine intelligence, and computer vision, it is necessary to achieve fast learning for large size-multivariate data sets using a low cost tool in order to handle the information contained in these data easily. Fast unsupervised linear techniques more or less all rely on *Principal Component Analysis* (PCA) [1]. It yields a linear mapping (or representation) with the most minimum amount of information loss. In addition, PCA is enough stable and viewed as the least cost approach in the linear analysis domain.

However, in some situations, there is a possibility that the feature space is winding. Since PCA summarizes the data by the mean and the standard deviation (the covariance matrix), the linear representation is accurate only if the data distribution is Gaussian. In other words, PCA is inappropriate tool for modeling nonlinear effects such as data bending or shape rotation [2].

The Kohonen *self-organizing map* (SOM) [3] can be viewed as a non-linear extension of PCA. It replaces the linear subspace of PCA by a *nonlinear manifold* that can represent even the winding data distributions. The manifold is constructed by an iterative learning procedure and can be viewed as a non-linear, 'topology-preserving map' of the original data space.

However, the discrete nature of the standard SOM can be a limitation when the construction of smooth, higher-dimensional map manifolds is desired by the more powerful learning algorithms such as face and robot applications. On the other side, increasing the feature space dimensions will increase the computational load of conventional SOM.

In summarizing, a powerful learning algorithm preserves the properties of conventional SOM and, in mean time, avoids higher computational cost, like PCA, is desired. This paper presents a much Fast SOM (FSOM) which takes the search complexity of $\boldsymbol{O}(\log N)$**,** where M is the number of nodes. It is a non-parametric simple method does not need any pre-calculation steps and consists of a piecewise one dimension SOM network. Its idea is based on the concept that the most relevant features are naturally aligned around the Principal Components *PCs* [1] [4]. Therefore, in *N*-dimensional hyperplane spanned by the *N-PCs* of input, it starts by deciding the first *PC* and pick up the first *winner*. Then, deciding the second *PC* and concern only about its neurons which pass on the first *winner*, and also pick up the second *winner*, and so on until the *N-winner*. By this strategy, the search algorithm consumes the minimum time. In conclusion, FSOM network is enough stable, simple and low cost alternative to the original SOM network and viewed as a non linear extension to PCA.

The outline of this paper is as follows: Section 2 gives overview about original SOM. Then, the proposed algorithm is provided in section 3. Experimental results are shown in section 4. Finally, a conclusion and future work.

2. SOM & Computation Complexity

2.1 Overview

In the context of image processing, the conventional SOM provides a good quantization of the image samples into a topological low-dimensional space such that inputs which are nearby in the original space are also nearby in the output space, thereby providing dimensionality reduction and invariance to minor changes in the image sample. This topological preservation of SOM makes it so useful in the classification of data which includes large number of classes.

Consider the input data $\mathcal{X} = \{\boldsymbol{x}_i, 1 < i < M\}$ belongs to a high dimensional space: $\boldsymbol{x}_i = (\boldsymbol{x}^{(l)}{}_i)_{1<l<n} \in \mathbb{R}^n$. SOM is usually represented as a neural network sheet or map whose units, usually called nodes or neurons, become tuned to different input vectors $\boldsymbol{x}_i$. A weight vector $\boldsymbol{w}_k$ (sometimes called reference) is associated with each neuron k and the map weight vectors are given by $\mathcal{W} = \{\boldsymbol{w}_j, 1 < j < N\}$, where $N < M$.

In each training step, the following two steps are repeated for each input sample $\boldsymbol{x}_i$.

1. Using a similarity measure between input and all the map's neurons, find the best matching neuron, called winner, c which satisfies:

$$\|\boldsymbol{x}_i - \boldsymbol{w}_c\| = \min_u(\|\boldsymbol{x}_i - \boldsymbol{w}_u\|) \tag{1}$$

2. Update the weigh vector of the winner c and also all its topological neighborhood in the map towards the prevailing input according to the rule:

$$\boldsymbol{w}_u(t+1) = \boldsymbol{w}_u(t) + h_{cu}(t)[\boldsymbol{x}_i(t) - \boldsymbol{w}_u(t)] \tag{2}$$

$$h_{cu}(t) = \alpha(t).\exp\left(\frac{\|r_c - r_u\|}{2\sigma^2(t)}\right) \tag{3}$$

$h_{cu}(t)$ is the neighborhood kernel function around the *winner c* at time t, $\alpha(t)$ is the learning rate and is decreased gradually toward zero and $\sigma^2(t)$ is a factor used to control the width of the neighborhood kernel. The term $\|r_c - r_u\|$ is referring to the distance between the *winner* neuron c and neuron u. After the training data is exhausted, the neurons sheet is automatically organized, without external supervision, into a meaningful two-dimensional order denoted by feature map (or codebooks).

2.2 Motivation

From the computation complexity point of view, SOM is expensive approach comparing to PCA when a large size map is needed. This is because "each" learning pass requires a computation of the distance of the current sample to "all" nodes in the map; as in equation 1, which is **O(*N*M*)** [5].

It has been noted that a manageably sized lattices with, in most works now, two dimensions admit only very few nodes along each axis direction and can, therefore, be not sufficiently smooth for many purposes where continuity is very important, as e.g. in control tasks or in robotics or face applications [6]. On the other side, as the number of nodes grows exponentially with the number of map dimensions, then using up to 2 dimensions will let performance is slow. In conclusion, original SOM search algorithm is not easily affordable for most of dynamic image recognition problems.

The authors are already observed this phenomenon and they already developed a lip-reading system based on conventional SOM [7-9]. Under same conditions, Table 1 shows the recognition accuracy for a lip-reading data set in case of training and testing phases. The left column shows the number of dimensions of SOM. Each dimension includes 8 neuron; which means that the feature map (FM) has 8x8 neurons in case of 2-Dim and 8x8x8 neurons in case of 3-Dim, and so on. As it is shown, if we increase the number of FM dimensions the recognition accuracy is becoming better and, in main time, the recognition time becomes longer. Pleas pay attention to that the resolution of input image is 160x120 and recognition time is measured by second.

Table 1. Relation between: Number of FM dimensions, Time and Accuracy

# Dim	Time "sec"	Training Data		Testing Data	
		Word	Sent	Word	Sent
2-Dim	92.4	76.9	61.1	51.6	40.1
3-Dim	656.7	88.5	74.1	60.3	44.4
4-Dim	5312.8	92.3	85.2	76.9	51.8

Motivated by higher computation cost problem of SOM and our previous work using SOM for lip-reading applications [7-9], we propose a new fast search rule can enhance the learning SOM algorithm.

2.3 Feature Map and Principal Components Extraction

The issue now is: How can the topological order of SOM can carry out the analysis of non-linear *PCs*. As we explained in the previous section that SOM forms a discrete space (map) such that each point (node) j has a reference vector $\boldsymbol{w}_j$ indicates the corresponding point in the original space, and has a neighborhood range $\boldsymbol{w}_j^k$. When a training data sample $\boldsymbol{x}_i$ is given, then SOM tries to find the best codebook that can:

1. Minimize the quantization error $\sum_i \|\boldsymbol{x}_i - \boldsymbol{w}_c\|$, included in (1), where c is the winner and given by the winner-take-all rule: $\boldsymbol{c} = \min_j \sum_i \|\boldsymbol{x}_i - \boldsymbol{w}_j\|$,
2. Minimize the total distance among the neighborhoods measured in the original space $\sum_{jk} \|\boldsymbol{w}_j - \boldsymbol{w}_j^k\|$.

So, to fulfill first factor and minimize the quantization error, the codebooks should be aligned along the first *PC*. Then, to fulfill the second factor and minimize the total distance between neighborhoods, they should be aligned in order of the lateral neighborhood which constructs the second *PC*. In that sense SOM is able to extract the *PC*s such that each dimension matches a *PC*.

Now, the original N-dimensional SOM is denoted by:

$$\boldsymbol{u} = \left\{ \boldsymbol{u}_{d_1,d_2,\ldots,d_j} \mid d_j = 1,2,..,D_j,\ j = 1,..,N \right\} \tag{4}$$

where $\boldsymbol{u}_{d_1,d_2,\ldots,d_N}$ terms to the neurons aligned through the dimensions $d_1,d_2,..,d_N$. Each dimension $d_j = 1,2,..,D_j$ where D_s refers to the maximum size of dimension s such that $D_1 > D_2 > \cdots > D_N$. For simplicity we will denote the term $\boldsymbol{u}_{d_1,d_2,\ldots,d_N}$ by the word "neuron". Each neuron $\boldsymbol{u}_{d_1,d_2,\ldots,d_N}$ has codebook vector as $\boldsymbol{w}_{d_1,d_2,\ldots,d_N}$. According to equation (1) the *winner* neuron is decided according to the winner-take-all rule:

$$\left\|\boldsymbol{x_i} - \boldsymbol{w}_{c_1,c_2,\ldots,c_N}\right\| = \min_{d_j}\left(\left\|\boldsymbol{x_i} - \boldsymbol{w}_{d_1,d_2,\ldots d_N}\right\|\right) \tag{5}$$

It is clear that, the computational complexity to get the winner neurons list $c_1,c_2,..,c_N$ during the N-dimension SOM through (5) is $O\left(D_1 \cdot D_2 \cdot \ldots \cdot D_N\right)$. The following section addresses the proposed algorithm.

3. Fast SOM (FSOM) Algorithm

3.1 FSOM Structure

The new SOM views to each dimension from the N-dimensions as one-dimension SOM such that each one-dimension SOM matches a *PC* in the feature space. In other words, the FSOM structure consists of the following N-sequence (piecewise) of one-dimension SOM:

$$\boldsymbol{u}_1 = \left\{ u_{1,d_1} \mid d_1 = 1,2,..,D_1 \right\} \tag{6}$$

$$\boldsymbol{u}_2 = \left\{ u_{2,d_1} = \left\{ u_{2,d_1,d_2} \mid d_2 = 1,2,..,D_2 \right\} \right\} \tag{7}$$

…

$$\boldsymbol{u}_N = \left\{ u_{N,d_1,\ldots,d_{N-1}} = \left\{ u_{N,d_1,\ldots,d_N} \mid d_N = 1,2,..,D_N \right\} \right\} \tag{8}$$

Such that, the neuron u_{1,d_1} refers to the neurons aligned through the first "one dimension SOM" (or the first *PC*), u_{1,d_2} refers to the neurons aligned through second "one dimension SOM" (or the second *PC*) and so on. The codebook for each *PC* u_{n,d_n} is denoted by w_{n,d_n}.

3.2 Learning Phase

The learning process is described as a recursive call for the function $\text{Learn}\left(1, u_{1,d_1}\right)$, where 1 is the order of the extracted component, and u_{1,d_1} terms to the neuron aligned around this component. This means that we will start to extract "piecewisely" the first *PC* then second *PC* and so on. Thus, the *N*-dimension function $\text{Learn}\left(n, u_{n,d_n}, C\right)$ can be generated as given in Table 2.

Table 2. Fast SOM learning algorithm

{ **If** (n=*N*)

{ - For each input sample $\boldsymbol{x_i}$, "train" each neuron u_{N,d_n,c_N}. That is for each $\left(d_N = 1,2,..,D_N\right)$ apply the winner-take-all rule: $\left\| x_i - w_{N,d_N}(t) \right\|$.

- Then "update" the codebook according to $w_{N,d_N,c_N}(t+1) = w_{N,d_N,c_N}(t) + \alpha(t).\left\{x_i - w_{N,d_N,c_N}(t)\right\}$ }

else {

1- "Train" $(N-n+1)$-dimensional SOM $u = \left\{u_{d_n,d_{n+1},...,d_N} \mid d_j = 1,2,..,D_j, j = n, n+1,.., N\right\}$ where $D_n > D_{n+1} > \cdots > D_N$,

2- Regard to the central column of the current codebook units; $PC_n = \left\{u_{d_n,\frac{D_{n+1}}{2},...,\frac{D_N}{2}} \mid d_n = 1,2,..,D_n\right\}$ as the *n*-th *PC* at the *winner* list *C*, and "copy" it onto u_{n,d_n}.

3- For each d_n, *Do*: $\text{Learn}\left(n+1, u_{n+1,d_n,c_n}, C\right)$

} //end of else

} //end of algorithm

In step 2 above we decide the *PC* by using the central column of current map and copy to first *PC*, then for second *PC* and so on until getting all *PCs*. The simplicity of the proposed approach is obvious as this tail recursive function can easily translate to a one For-loop statement. In this For-loop, the function $\text{Learn}\left(n, u_{n,d_n}, C\right)$ calls the function $\text{Learn}\left(n+1, u_{n+1,d_{n+1}}, C\right)$ D_n times. Now, as the sizes of $u_{n+1,d_{n+1}}$ is $\frac{1}{D_n}$ of u_{n,d_n}, therefore, the computational complexity to train u_{n+1} is $\frac{1}{D_n}$ of the complexity for u_n. In terms, if the computational complexity to train u_n (or conventional *N*-dimension SOM) is $\boldsymbol{C}$, then, overall complexity of the new approach results in

$$\boldsymbol{C}\left(1 + \frac{1}{D_1} + \frac{1}{D_1 D_2} + \cdots + \frac{1}{\prod_i D_i}\right) \approx \boldsymbol{C} \tag{9}$$

This means that the time to train the FSOM is the same of that to train conventional SOM.

3.3 Recognition phase

In image recognition domain, the most challenge is to achieve recognition in a real time or near to real time. In FSOM, after getting an ordered feature map during learning phase, the final *winner* $u_{N,d_N,C}$ is selected from u_N by using the following *N*-steps, consequently.

- First, winner-take-all rule is applied to select first *winner* u_{1,c_1} from first *PC* according to:

$$u_{1,c_1} = \min_j \left(\left\| x_i - w_{1,j} \right\| \right) \tag{10}$$

- Second *winner* u_{2,c_1,c_2} is picked from second *PC* and selected by

$$u_{2,c_1,c_2} = \min_j \left(\left\| x_i - w_{2,c,j} \right\| \right) \tag{11}$$

- Finally, the *N-winner* is picked up according to:

$$u_{N,WinList} = \min_j \left(\left\| x_i - w_{N,Winlist,j} \right\| \right) \tag{12}$$

Obviously, computational complexity during FSOM recognition phase is $O(D_1 + \cdots + D_N)$. Of course it is less than that of the conventional *N*-dimension SOM concluded from 5; which is $O\left(D_1 \cdot D_2 \cdot .... \cdot D_N\right)$.

3.4 Example

Exploiting the above strategy, let us show, using simple example, how the new method runs. For simplicity, consider the number of dimensions is 3 (i.e. n=3); such that each dimension includes 10 neurons.

(1) The conventional SOM in 3-dimendions can be given as in (5):

$$u_3 = \left\{ u_{3,d_1,d_2,d_3} \mid d_i = 1,2,..,10 \right\} \quad (13)$$

where $D_1 > D_2 > D_3$. From (5), it seems that computational load of recognition phase is included (10×10×10) steps or 1000 steps.

(2) The recognition phase of FSOM in 3 dimensions also runs as follows: Apply winner-take-all rule directly to (10-12) as follows:

- First, winner-take-all rule is applied to first *PC*. Then first *winner* is selected by:

$$u_{c,} = \min_{l} \left(\left\| \mathbf{x}_i - \boldsymbol{w}_{1,l} \right\| \right) \quad (14)$$

- Second, we will concern, ONLY, about the neurons of second *PC*, red color in Figure 2, which already passing through first *winner c*. Then second *winner* is selected by:

$$u_{c,d} = \min_{j} \left(\left\| \mathbf{x}_i - \mathbf{w}_{2,c,j} \right\| \right) \quad (15)$$

Similarly we will concern, ONLY, about the neurons, u_3, of third *PC* which passing through first and second *winners c* and *d* respectively. Then the third *winner* is selected by:

$$u_{c,d,m} = \min_{k} \left(\left\| \mathbf{x}_i - \mathbf{w}_{3,c,d,k} \right\| \right) \quad (16)$$

Obviously, computational load during FSOM recognition phase is (10+10+10) steps or 30 steps, which is much less than the 1000 steps which required for running SOM.

4. Experimental Results

The authors already presented lip-reading systems for Japanese [7] and Arabic [8] data sets. The number of image is 5670 gray image for Japanese set and 7200 image for Arabic data. Each image has resolution as 160x120 pixels. Each set of image is captured from 9 native subjects such that each one of them uttered 9 different sentences using his language. Some examples for our images are shown in Fig. 1. For more details about the two data sets please refer to [7-9].

Fig.1. Examples for lip reading images

4.1. Comparison between SOM and FSOM

Exploiting same data sets and same environment we provide here a comparison between SOM and FSOM. We achieved many experiments using different number of dimensions for each data set. Since our motivation is computation time, the column "Rate" in the tables given here represents the ratio of FSOM recognition time to SOM recognition time in order to realize the rate of acceleration. In all table given here, the measure unit is second. First for Arabic data set shown in Table 3, the experiments are done in 2, 3 and 4 dimensions using 8x8, 8x8x8 and 8x8x8x8 feature map sizes, respectively. As it is shown, FSOM starts to be faster than the original SOM by 7.6 times. As we increase the dimensions (or map size), the convergence rate of SOM drops drastically while FSOM rate convergences quickly.

Table 3. Arabic Data set: Recognition Time "Second"

# Dim	SOM	FSOM	Rate
2	92.4	12.1	7.6
3	656.7	36	18
4	5312.8	101.9	52

Table 4. Japanese Data set: Recognition Time "Second"

# Dim	SOM	FSOM	Rate
2	144.7	25.2	5.7
3	713.6	51.4	14
4	6652.1	154.7	43

Similarly, Table 4 shows the recognition time and rate during Japanese experiments, which are achieved using 11 x 11, 11 x 9 x 7, and 11 x 9 x 7 x 7 sizes for the feature map, respectively. Therefore, we expect that the impact of FSOM becomes clearer when large data sets are used such as in face applications, which usually include more than 10000 images require wider feature map.

4.2 Comparison between FSOM and FDCT

It is demonstrated that, the FDCT has the ability to concentrate the information contained in an image to few coefficients only. Therefore, unlike original SOM, the calculation of FDCT coefficients requires much less computational effort in a similar way to the Fast Fourier Transform (FFT) [10]. As it is shown in Table 5, using same data set and same experiment environment, that fast SOM is faster than FDCT by about 6.5 times; in Table 5 the measure unit is second.

Table 5. Recognition Time “Second” Comparison: FSOM and FDCT

Data	FDCT	FSOM
Japanese	302.8	47.2
Arabic	374.4	58

5. Conclusion

In this paper we presented a new SOM search algorithm reduces the computational complexity of conventional SOM while preserving the basic quality of SOM. The new search algorithm based on the fact that most of important information are given by the neurons which aligned around principal components. The proposed algorithm consists of *N* one-dimension SOM such that each one-dimension SOM matches a principal component. Exploiting two lip-reading data sets, we showed that the new SOM needs computation efforts less than the conventional SOM. Until the time of preparing this manuscript, Recognition accuracies did not improve and showing same as those given in [7-9]. Currently, we do our best to improve the recognition accuracies. Also we plan to use the new method in one of face recognition problems.

References

[1] I.T. Jolliffe, “Principal Component Analysis”, 2nd Ed, Springer, 2002.

[2] B. Chalmond and S.C. Girard, “Nonlinear Modeling of Scattered Multivariate Data and Its Application to Shape Change”, the IEEE Transaction on Pattern Analysis and Machine Intelligence, IEEE, Vol 21 (5), May 1999, pp 422-432.

[3] T. Kohonen, “Self Organizing Maps,” 3rd edition, Springer series in information sciences, 2001.

[4] B. Lu and W. W. Hsieh, “Simplified nonlinear principal component analysis”, Proceedings of the International Joint Conference on Neural Networks, Vol 1, July 2003, pp. 759 – 763.

[5] S. Lawrence, C. Giles, A. Tsoi and A Back, “ Face Recognition: A Convolutional Neural Network Approach,” the IEEE Transactions on Neural Networks, Special Issue on Neural Networks and Pattern Recognition, Vol 8, No. 1, 1997,pp. 98–113.

[6] J. Walter and H. Ritter,” Rapid learning with parametrized self-organizing maps,” Neurocomputing, 12 (1996),pp 131-153.

[7] A. Sagheer, N. Tsuruta, R. Taniguchi, S. Maeda, “Visual Speech Features Representation for Automatic Lip-Reading,” Proceedings of the 30th IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP05, Vol 2, 2005, pp: 781 – 784.

[8] A. Sagheer, N. Tsuruta, R. Taniguchi, S. Maeda, “Hyper-Column Model vs. Fast DCT for Feature extraction in Visual Arabic Speech Recognition,” The IEEE international Symposium in Signal Processing and Information Technology, ISSPIT05, Dec. 2005.

[9] A. Sagheer, N. Tsuruta, R. Taniguchi, S. Maeda, “Appearance Features Extraction vs. Image Transform for Visual Speech Recognition,” Submitted to the Journal of Japanese Society of Information Processing Systems (IPSJ).

[10] K. Rao and P. Yip,”Discrete cosine transform : algorithms, advantages, applications,” Academic Press, 1990.