LVQ Clustering and SOM Based on Inner Product by using a Kernel Function

Kiyotaka Mizutani *, Ryo Inokuchi * and Sadaaki Miyamoto **

* Graduate School of Systems and Information Enginnering,
University of Tsukuba, Ibaraki, 305-8573, Japan

** Department of Risk Enginnering, School of Systems and Information Engineering,
University of Tsukuba, Ibaraki 305-8573, Japan

E-mail: kiyotaka@odin.esys.tsukuba.ac.jp, ryo@odin.esys.tsukuba.ac.jp, miyamoto@risk.tsukuba.ac.jp

***Abstract*— This paper aims at proposing clustering algorithm based on Learning Vector Quantization (LVQ) and Self-Organizing Maps (SOM) using the inner product. Furthermore, a kernel function employed in nonlinear transformation into a high dimensional feature space in the support vector machines is considered for LVQ clustering and SOM. Two numerical examples of the clustering are shown and effects of the inner product and the kernel function are discussed by comparing to other methods of fuzzy c-means.**

I. Introduction

Clustering of numerical data forms the basis of many classification and system modeling algorithm. One of the most popular technique in cluster analysis is the Clustering by Learning Vector Quantization (LVQ) [5]. The LVQ algorithm have been considered in the framework of Self-Organizing Maps (SOM) [5].

It is well-known that the above method provide linear cluster boundaries. Namely, the boundary between two clusters obtained by these methods are linear. However, it is also known that real-world examples request us methods of separating classes having nonlinear boundaries.

To obtain nonlinear cluster boundaries, a powerful method is the use of kernel functions employed in the support vector machines [1], [10], where original data are mapped into a higher dimensional feature space. Using this technique, data are classified linearly in the feature space but in the original data space the boundary appear nonlinear.

In this paper, we propose an LVQ clustering algorithm based on the inner product, instead of the most known the Euclidean distance, using a kernel function. Moreover, we visualize how the data is separated in the feature space by SOM using the kernel.

Two numerical examples are shown and we discuss effects of the kernel and the inner product. First is an illustrative numerical example. We show effectiveness of the kernel function and we visualize how the data is separated in the feature space by SOM using the kernel in this example. Second example is based on real data. We compare effectiveness and efficiency of the present method with other algorithms in foregoing studies [2], [5], [7], [8].

II. LVQ Clustering

Clustering by learning vector quantization (LVQ) is based on competitive learning [3], [5] and this algorithm is also a standard technique of unsupervised classification as same as k-means and fuzzy c-means [2].

Objects to be clustered are denoted by $x_k = (x_k^1, \ldots, x_k^p) \in \boldsymbol{R}^p$, $k = 1, \ldots, n$, a vector in the p-dimensional vector space. Objects are also called pattern vectors. The number of clusters is taken to be c, as in many literature [2], [6]. Cluster centers are denoted by $m_i(t)$, $i = 1, \ldots, c$, $t = 1, 2, \ldots$; m_i is also called a reference vector. The symbol t shows a time variable. In the algorithms below $x(t)$ is taken from the objects $\{x_1, \ldots, x_n\}$ randomly and repeatedly.

Here, we call the following algorithm LVQC (Clustering by LVQ).

Algorithm LVQC.

LVQC1. Set initial value for $m_i, i = 1, \ldots, c$ (for example, select c objects randomly as $m_i, i = 1, \cdots, c$). Normalize x_k : $x_k \leftarrow x_k/\|x_k\|$, $k = 1, \ldots, n$. Repeat **LVQC2** and **LVQC3** for $t = 1, 2, \ldots$ until convergent.

LVQC2. Let

$$m_l(t) = \arg \max_{1 \le i \le c} \langle x(t), m_i(t) \rangle. \tag{1}$$

LVQC3. Update $m_1(t), \cdots, m_c(t)$:

$$m_l(t+1) := \frac{m_l(t) + \alpha x(t)}{\|m_l(t) + \alpha x(t)\|}, \tag{2}$$

$$m_i(t+1) := m_i(t), \ i \neq l. \tag{3}$$

Object represented by $x(t)$ is allocated to G_l.

End of LVQC.

In this algorithm, $\langle \cdot, \cdot \rangle$ indicate the inner product, $\| \cdot \|$ indicate the norm of data space and the parameter $\alpha(t)$ satisfies

$$\sum_{t=1}^{\infty} \alpha(t) = \infty, \ \sum_{t=1}^{\infty} \alpha^2(t) < \infty, \ t = 1, 2, \cdots . \tag{4}$$

For example, $\alpha(t) = Const/t$ satisfies these conditions.

III. Self-Organizing Maps

SOM [5] is a method of unsupervised learning whereby spatial relations of objects (patterns) are represented. The hexagonal array of the second layer is used in this paper. Given an input pattern, reference vector that have the maximum inner product to the input vector is the winner m_l. The nodes in the neighborhood N_c of m_l are all modified. The radius of N_c is reduced with the time. The algorithm of SOM is similar to LVQ except that the neighborhood is used.

Algorithm SOM.

SOM1. Generate initial nodes $m_i(1), i = 1, \cdots, K$. Repeat **SOM2** and **SOM3** for $t = 1, 2, \cdots$ until convergence.

SOM2. Let

$$m_l(t) = \arg \max_{1 \le i \le K} \langle x(t), m_i(t) \rangle. \tag{5}$$

SOM3. Update all nodes in the neighborhood N_c of the winner m_l.

$$m_l(t+1) := \frac{m_l(t) + \alpha x(t)}{\| m_l(t) + \alpha x(t) \|}, \tag{6}$$

$$m_i(t+1) := m_i(t), \ i \neq l. \tag{7}$$

End of SOM.

IV. Kernel Trick

Kernel trick is well known technique in the Support Vector Machines - SVM [1], [10] by which nonlinear classification is effectively realized.

Studies in SVM often employ a high dimensional feature space H for having nonlinear classification boundaries. Here H is in general an infinite dimensional inner product space. For this purpose a mapping $\Phi : \boldsymbol{R}^p \to H$ is used whereby an object x is mapped into H:

$$\Phi(x) = (\phi_1(x), \phi_2(x), \cdots). \tag{8}$$

Although x is a p-dimensional vector, $\Phi(x)$ may have infinite dimension.

In the nonlinear classification method an explicit form of $\Phi(x)$ is unavailable, but the inner product is denoted by

$$K(x, y) = \langle \Phi(x), \Phi(y) \rangle. \tag{9}$$

The function $K(x, y)$, called a kernel function, is assumed to be known.

For example,

$$K(x, y) = \exp(-cnst \parallel x - y \parallel^2), \tag{10}$$

$$K(x, y) = (1 + \langle x, y \rangle)^d \tag{11}$$

are frequently used. The first is called the Gaussian kernel, while the second is called the polynomial kernel.

In this paper the Gaussian kernel is used.

A. LVQ Clustering Using a Kernel

Instead of the measure $\langle x(t), m_i(t) \rangle$ in the data space, the next measure in a high dimensional feature space is considered:

$$s_{ik} = \langle \Phi(x_k), w_i \rangle. \tag{12}$$

It should be noted that w_i is the reference vector in the high dimensional feature space. Hence updating formula of **LVQC** is as follows.

$$w_l(t+1) = \frac{w_l(t) + \alpha \Phi(x_l)}{\| w_l(t) + \alpha \Phi(x_l) \|}. \tag{13}$$

Since we do not know a explicit form of $\Phi(x)$, we cannot explicitly use w_i. Hence the algorithm should be rewritten using s_{ik} and the kernel function instead. We therefore calculate $s_{ik}(t+1)$ using $s_{ik}(t)$.

For simplicity, put $\alpha = \alpha(t)$, $K_{kk} = K(x_k, x_k)$, and $K_{kl} = K(x_k, x_l)$. Then,

$$\begin{aligned} s_{ik}(t+1) &= \langle \Phi(x_k), w_i(t+1) \rangle \\ &= \langle \Phi(x_k), \frac{w_i(t) + \alpha \Phi(x_l)}{\| w_i(t) + \alpha \Phi(x_l \|} \rangle \\ &= \frac{\langle \Phi(x_k), w_i(t) \rangle + \alpha \langle \Phi(x_k), \Phi(x_l) \rangle}{\| w_i(t) + \alpha \Phi(x_l) \|} \end{aligned}$$

Here, using $\| w_i \| = 1$, we have

$$s_{ik}(t+1) = \frac{s_{ik}(t) + \alpha K_{kl}}{\sqrt{1 + 2\alpha s_{il}(t) + \alpha^2 K_{ll}}} \tag{14}$$

which is the updating formula for s_{ik}. Allocation of an object x_k to a cluster should use maximum of s_{ik}, $i = 1, \ldots, c$.

We now have the kernel-based LVQ clustering algorithm:

Algorithm K-LVQC.

K-LVQC1. Initialze $s_{ik}, \ i = 1, \cdots, c, \ k = 1, \cdots, n$.

K-LVQC2. Calculate

$$s_{lk}(t) = \arg \max_{1 \le i \le c} s_{ik}(t) \tag{15}$$

and allocate x_k to cluster l.

K-LVQC3. Update s_{ik} by (14).

End of K-LVQC.

B. SOM Using a Kernel

Formula of SOM using a kernel function can be derived in a similar manner to **K-LVQC**. The updating equation is

$$s_{ik}(t+1) = \frac{s_{ik}(t) + \alpha K_{kl}}{\sqrt{1 + 2\alpha s_{il}(t) + \alpha^2 K_{ll}}}. \tag{16}$$

Notice that the value of w_i is unnecessary; only the position of w_i in the grid is used.

Algorithm K-SOM.

K-SOM1. Initialize $s_{ik},\ i = 1, \cdots, c,\ k = 1, \cdots, n.$

K-SOM2. Calculate

$$s_{lk}(t) = \arg \max_{1 \le i \le c} s_{ik}(t). \tag{17}$$

K-SOM3. Update s_{ik} for all nodes i in the neighborhood N_c of the winner w_l using (16).

End of K-SOM.

V. Numerical Examples

Two examples were tested, of which first is an illustrative and capability of the present algorithm to handle nonlinearity was tested; the second is based on real data on which different algorithms were compared.

A. An Artificial Data

An artificially generated 150 points on three dimensional space was analyzed. The result by the algorithm **LVQC** is shown in Fig. 1 and the result by the algorithm **K-LVQC** is shown in Fig. 2, where two clusters are represented by $\square$ and $+$. In Fig. 1, it shows that the original data were not divided into the two classes. It is moreover obvious that the methods of ordinary k-means and fuzzy c-means cannot separate the ring around the ball and the ball within the ring. In contrast, Fig. 2 shows the effectiveness of the kernel-based method for having nonlinear boundary between clusters. Unlike the original data space, the two classes can be linearly separated in the high dimensional feature space, although we cannot observe the feature space directly.

A way of visualizing configuration of points in the high dimensional feature space is to use SOM. Fig. 3 shows the result of applying algorithm **K-SOM** to this example. In this figure $\square$ and $+$ are nodes nearest to the corresponding objects. As expected, the two classes are almost linearly separated in this figure. Thus visualization of the objects in the high dimensional feature space is successful in this figure.

It has been shown that such 'a ring around a ball' data can be successfully separated by a kernel-based crisp k-means algorithm [4] and a kernel-based fuzzy c-means algorithm [8]. The result in Fig. 2 implies that the **K-LVQC** algorithm is also successful in separating such data classes.

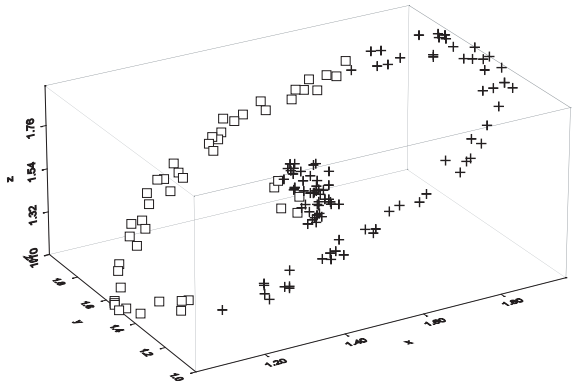

Fig. 1. Result by LVQC to 'a ring around a ball' data. Parameters are $(c = 2, \alpha = 0.5)$

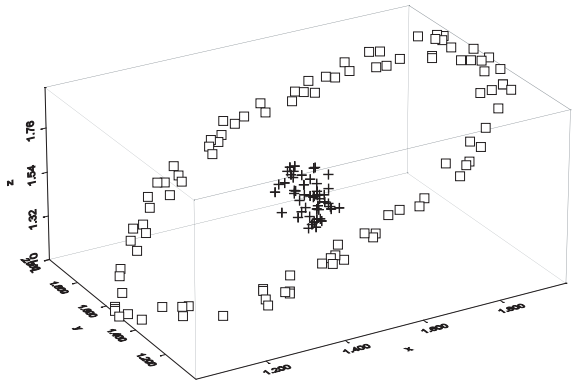

Fig. 2. Result by K-LVQC to 'a ring around a ball' data. Parameters are $(c = 2, \alpha = 0.05, cnst = 20)$

B. A Real Data

Iris data is well-known data set in statistics. We used this data set to compare different clustering algorithms with proposed algorithms. The data set has been downloaded from [9].

The Iris data set has 4 dimensional vectors of 150 objects in which 3 different types of Iris are included. In addition to the **LVQC** and **K-LVQC** algorithms based on the Euclidean distance and the inner product considered here, four other methods of fuzzy c-means, i.e., the standard fuzzy c-means **sFCM** [2] and entropy-based fuzzy c-means **eFCM** [7] without a kernel function, the standard fuzzy c-means with the kernel **K-sFCM** [8] and the entropy-based fuzzy c-means with the kernel **K-eFCM** [8] were tested.

Table I shows the number of misclassified objects and processor time until convergence by these eight methods. The processor time is average of 100 trials with different initial values in each methods. It is seen that the proposed method is better than the other methods and the use of the kernel is effective in the real data set. Furthermore, the processor time of **K-LVQ** is much less than **K-sFCM** or **K-eFCM**.

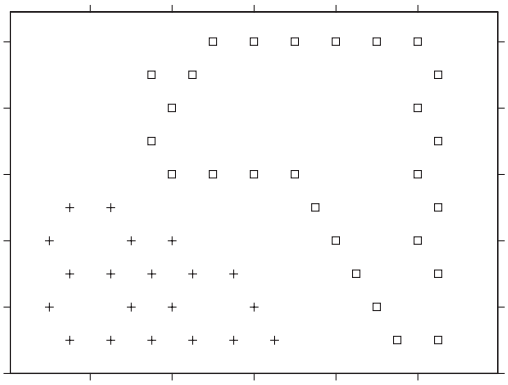

Fig. 3. Result by K-SOM to 'a ring around a ball' data. Parameters are $(c = 2, \alpha = 0.05, cnst = 20)$

TABLE I

NUMBER OF CLASSIFICATION ERRORS AND PROCESSOR TIME IN IRIS DATA BY DIFFERENT METHODS

algorithm	*num. of errors*	*processor time (sec)*
LVQC (I.P)	9	1.5×10^{-3}
LVQC (E.D)	14	1.4×10^{-3}
sFCM	15	1.4×10^{-2}
eFCM	16	7.8×10^{-3}
K-LVQC (I.P)	5	4.2×10^{-2}
K-LVQC (E.D)	9	2.6×10^{-2}
K-sFCM	11	37.3
K-eFCM	10	12.5

I.P : Inner Product, E.D : Euclidean Distance

Fig. 4 shows the result of applying algorithm **K-SOM** using the inner product to the Iris data. In this figure, $\square$ is Iris-setosa, $+$ is Iris-verginica and $\triangle$ is Iris-versicolor. It is seen that the three classes are almost linearly separated in this figure. We can't get such a result in the method which **K-SOM** using the Euclidean distance (Fig. 5).

VI. CONCLUSION

In this paper we have proposed LVQ clustering algorithm and SOM algorithm based on the inner product. Moreover nonlinearities in unsupervised automatic classification have been dealt with by employing kernel trick in the support vector machines. In the first numerical example, capability of the present algorithm **K-LVQ** in separating nonlinear classes has been shown; **K-SOM** moreover visualizes the data configuration in the high dimensional feature space. In the second numerical example, effectiveness of the inner product and kernel-based method has been shown.

Future studies include consideration of variations of the LVQ and other algorithms based on competitive learning [5] where kernel-based algorithms can be derived. Various data sets should be tested to examine effectiveness of nonlinear separation of clusters.

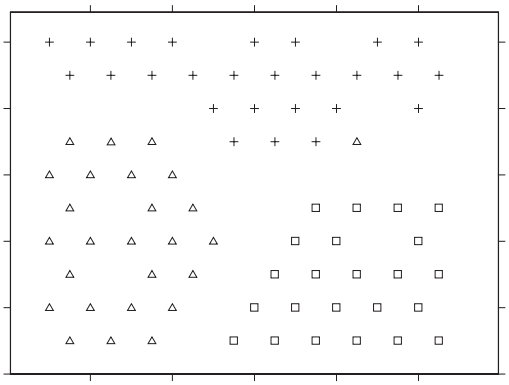

Fig. 4. Result by K-SOM using the inner product to the 'Iris data'. Parameters are $(c = 3, \alpha = 0.4, cnst = 20)$

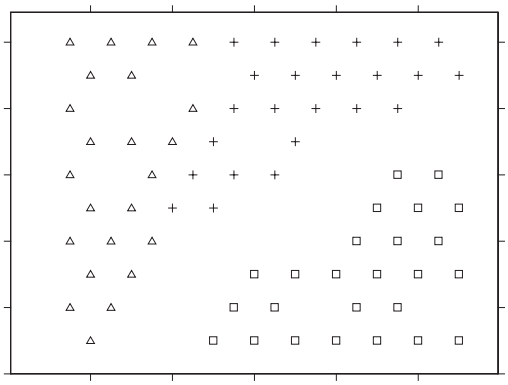

Fig. 5. Result by K-SOM using the Euclidean distance to the 'Iris data'. Parameters are $(c = 3, \alpha = 0.05, cnst = 0.8)$

REFERENCES

[1] A.Ben-Hur, D.Horn, H.T.Siegelmann, V.N.Vapnik, Support vector clustering, *Journal of Machine Learning Research*, Vol.2, pp.125–137, 2001.

[2] J.C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum, New York, 1981.

[3] R.O. Duda, P.E. Hart, D.G. Stork, *Pattern Classification, 2nd Ed.*, Wiley, New York, 2001.

[4] M. Girolami, Mercer kernel based clustering in feature space, *IEEE Trans. on Neural Networks*, Vol.13, No.13, pp. 780-784, 2002.

[5] T. Kohonen, *Self-Organizing Maps*, Springer-Verlag, Heidelberg, 1995.

[6] Z.Q. Liu, S. Miyamoto (eds.), *Soft Computing and Human-Centered Machines*, Springer, Tokyo, 2000.

[7] S. Miyamoto, M. Mukaidono, Fuzzy c - means as a regularization and maximum entropy approach, *Proc. of the 7th International Fuzzy Systems Association World Congress (IFSA'97)*, June 25-30, 1997, Prague, Chech, Vol.II, pp. 86–92, 1997.

[8] S. Miyamoto and D. Suizu, Fuzzy c-means clustering using kernel functions in support vector machines, *J. of Advanced Computational Intelligence and Intelligent Informatics*, Vol.7, No.1, pp. 25–30, 2003.

[9] "UCI repository of machine learning databases and domain theories", FTP address:ftp://ftp.ics.uci.edu/pub/machine-learning-databases

[10] V. Vapnik, *Statistical Learning Theory*, Wiley, New York, 1998