Split-merge Learning Vector Quantizer algorithm*

C. Kotropoulos and I. Pitas

Department of Electrical Engineering, University of Thessaloniki
Thessaloniki 540 06, Greece

1. INTRODUCTION

One of the most prominent neural networks (NN) in the literature is the Learning Vector Quantizer (LVQ) [1]. It is an associative nearest neighbor classifier which classifies p-dimensional arbitrary patterns into N-many classes using an error-correction learning procedure related to the competitive learning.

As its name suggests, LVQ is essentially a vector quantization method. Motivated by the following open questions in vector quantization methods, namely, the optimal number of code-vectors (i.e., output neurons) and the rejection of outliers in the formation of the minimum distortion partition, a split-merge LVQ algorithm is proposed that incorporates statistical hypothesis testing on mean vectors as well as additional tests that are used to determine if cluster splitting/merging is statistically significant.

2. SPLIT-MERGE LEARNING VECTOR QUANTIZER

Learning Vector Quantizer relies on the Euclidean distance in order to determine the best matching output neuron ("winner"). It is well known that in order for the Euclidean distance to be a most effective measure, input patterns must be linearly transformed to another ones that are uncorrelated and of equal variances [3]. Such a linear transformation yields the so-called Mahalanobis distance [2,3]. It can be easily seen that by simply replacing the Euclidean distance with the Mahalanobis one in the LVQ encoding algorithm, the inversion of the sample dispersion matrix having dimensions $p \times p$ would be required at each input pattern presentation. It will be seen later on that by employing tests on the mean vectors as an outlier rejection mechanism a quadratic form similar to Mahalanobis distance is needed to be evaluated and to be compared to a threshold. But in the later case, the number of times such an additional computation has to be performed is limited only to: (i) the pattern presentations during the first session (i.e., when the whole training set is presented to the input of LVQ for first time), (ii) the patterns that move from one cluster to another, and, (iii) the patterns of a cluster where a modification (i.e., insertion/removal of a pattern) has occurred during the session that modification took place.

2.1. Criteria for detecting outliers

Let us assume that an arbitrary number of output neurons exists in the output of an LVQ. Let $N(0)$ denote the number of initial neurons. Since all the statistical tests that will be employed next rely on first and second-order statistics, each output neuron evaluates the sample mean

* The research reported in this paper has been supported by ESPRIT Basic Research Action project NAT (7130).

vectors and the sample dispersion matrix associated with the cluster it represents. We shall also assume that a counter is associated with each neuron which counts the number of patterns that belong to the cluster represented by that neuron. At the beginning, the sample mean $\mathbf{m}_j$, the sample dispersion matrix $\mathbf{S}_j$ as well as the number of patterns n_j of each neuron's cluster are appropriately initialized.

Let us denote by i the index that counts the training sessions. For each pattern presentation $\mathbf{x}(k)$ $k = 1, \ldots, M$ during the i-th session, the winner neuron is found:

$$\| \mathbf{x}(k) - \mathbf{w}_c^{(i)}(k) \| = \min_{j=1}^{N^{(i)}(k)} \{ \| \mathbf{x}(k) - \mathbf{w}_j^{(i)}(k) \| \}. \tag{1}$$

$N^{(i)}(k)$ is the number of output neurons when the k-th training vector is presented in the input of LVQ during the i-th training session. Then, the number of patterns that are represented by the winner as well as the sample mean vector of the cluster associated with the winner are updated as if input pattern $\mathbf{x}(k)$ were merged into that cluster. Such an updating is required during the first training session ($i = 1$) as well as for $i \geq 2$ when $c^{(i)}(k) \neq c^{(i-1)}(k)$, where $c^{(i)}(k)$ denotes the index of the winner neuron at the presentation of the k-th pattern during i-th iteration. This is case (ii) outlined above. Case (iii) refers to the remaining patterns of a cluster which has been modified due to an insertion/removal of another pattern. Since $c^{(i)}(k) = c^{(i-1)}(k)$, for a moment, we exclude pattern $\mathbf{x}(k)$ from the cluster of patterns that is represented by the winner. Let us denote by $\mathbf{m}_\tau^{(i)}(k-1)$ the sample mean vector of the resulted cluster after the exclusion of $\mathbf{x}(k)$. Our purpose is to test if its inclusion to that cluster is still valid. A similar provision has to be made for the sample dispersion matrix of that cluster. Let

$$\mathbf{d} = \mathbf{m}_\gamma^{(i)}(k-1) - \mathbf{m}_c(k) = -\frac{1}{n_\gamma^{(i)}(k)} \mathbf{d}_\gamma^{(i)}(k) \quad \gamma \in \{c, \tau\}. \tag{2}$$

We decide that merging $\mathbf{x}(k)$ with the remaining patterns is valid, if [2]:

$$\begin{aligned} \left(\frac{n_c^{(i)}(k-1) - p}{p} \right) \mathbf{d}^T [\mathbf{S}_c^{(i)}(k-1)]^{-1} \mathbf{d} \quad &\leq \quad F_{p, n_c^{(i)}(k-1)-p; 0.05} \quad \text{for cases (i), (ii)} \\ \left(\frac{n_\tau^{(i)}(k) - p}{p} \right) \mathbf{d}^T [\mathbf{S}_\tau^{(i)}(k-1)]^{-1} \mathbf{d} \quad &\leq \quad F_{p, n_\tau^{(i)}(k)-p; 0.05} \quad \text{for case (iii).} \end{aligned} \tag{3}$$

where $F_{p,n-p;0.05}$ denotes the upper 5% level of significance for the F-distribution with n and $n-p$ degrees of freedom.

If H_0 is accepted, then the winner vector is updated as LVQ suggests [1]:

$$\mathbf{w}_c^{(i)}(k+1) = \mathbf{w}_c^{(i)}(k) + \alpha(i) \left[\mathbf{x}(k) - \mathbf{w}_c^{(i)}(k) \right] \tag{4}$$

where $\alpha(i)$ is a variable adaptation step defined as $\alpha(i) = 0.2 \left[1 - \frac{i}{1000} \right]$ [1]. The updating of $n_c^{(i)}(k-1)$ and $\mathbf{m}_c^{(i)}(k-1)$ to $n_c^{(i)}(k)$ and $\mathbf{m}_c^{(i)}(k)$ respectively are assumed valid. Furthermore, the sample dispersion matrix of the cluster associated with the winner is also updated. Moreover, in case (ii) the number of patterns, the sample mean and the sample dispersion matrix associated with the cluster of past winner are corrected. For the remaining neurons (i.e., $j = 1, \ldots, N^{(i)}(k)$, $j \neq c$ in cases (i), (iii) and $j \neq c, v$ in case (ii)), all the corresponding parameters are left intact.

2.2. Splitting criteria

If H_0 is rejected, it is reasonable to examine whether the cluster represented by the winner neuron can be split into two subclusters. Let us denote that cluster by $C_c^{(i)}(k-1)$. We shall borrow from the field of cluster analysis [4,3] a statistic that relies on the sum of squared errors $J_e(g)$, $g=1,2$ to test the validity of the following possibilities: (a) cluster $C_c^{(i)}(k-1)$ is kept united ($g=1$), and, (b) cluster $C_c^{(i)}(k-1)$ is subdivided into two clusters ($g=2$), say $C_\zeta^{(i)}(k-1)$ and $C_\eta^{(i)}(k-1)$.

Let us define first the sum of squared errors in cases (a) and (b) outlined above. We have:

$$J_e(g) = \begin{cases} \sum_{\mathbf{x}_j \in C_c^{(i)}(k-1)} \| \mathbf{x}_j - \mathbf{m}_c^{(i)}(k-1) \|^2 & \text{for } g=1 \\ \sum_{\gamma \in \{\zeta,\eta\}} \sum_{\mathbf{x}_j \in C_\gamma^{(i)}(k-1)} \| \mathbf{x}_j - \mathbf{m}_\gamma^{(i)}(k-1) \|^2 & \text{for } g=2 \end{cases} \tag{5}$$

where $\mathbf{m}_\zeta^{(i)}(k-1)$ and $\mathbf{m}_\eta^{(i)}(k-1)$ denote the sample mean vectors of the resulted subclusters. In the sequel, we shall describe how the tentative splitting is performed.

We determine the direction in which cluster $C_c^{(i)}(k-1)$ variation is greatest. This amounts to finding the principal component of the sample dispersion matrix (i.e., the eigenvector that corresponds to the largest eigenvalue of $\mathbf{S}_c^{(i)}(k-1)$). Let us denote by $\mathbf{e}_c^{(i)}(k-1)$ the principal (normalized) eigenvector of $\mathbf{S}_c^{(i)}(k-1)$. Having determined $\mathbf{e}_c^{(i)}(k-1)$, we examine the splitting of cluster $C_c^{(i)}(k-1)$ with a hyperplane which is perpendicular to the direction of $\mathbf{e}_c^{(i)}(k-1)$ and passes through the sample mean $\mathbf{m}_c^{(i)}(k-1)$.

It is well known that splitting of any cluster to two subclusters will result a lower sum of squared errors, i.e., $J_e(2) < J_e(1)$ [4]. We decide to consider as valid any splitting that yields a statistically significant improvement (i.e., decrease) in the above-mentioned criterion. Following the analysis given in [4], we accept cluster splitting at the ρ-percentage significance level, if

$$\frac{J_e(2)}{J_e(1)} < 1 - \frac{2}{\pi p} - \beta \sqrt{\frac{2(1-\frac{8}{\pi^2 p})}{n_c^{(i)}(k-1)p}} \tag{6}$$

where $\rho = 100 \int_\beta^\infty \frac{1}{\sqrt{2\pi}} \exp[-\frac{u^2}{2}]\, du$.

If cluster splitting is accepted, we proceed to the evaluation of the sample dispersion matrices for $C_\zeta^{(i)}(k-1)$ and $C_\eta^{(i)}(k-1)$. Next, it is examined if the current training pattern $\mathbf{x}(k)$ can be merged with one of the subclusters $C_\zeta^{(i)}(k-1)$ or $C_\eta^{(i)}(k-1)$ by applying the statistic described in Sect. 2.1. The winner neuron is replaced by the two newly created ones. Their weight vectors are set equal to the sample mean vectors $\mathbf{m}_\zeta^{(i)}(k)$ and $\mathbf{m}_\eta^{(i)}(k)$.

If $\mathbf{x}(k)$ cannot be merged with any of the subclusters created by splitting $C_c^{(i)}(k-1)$, a third subcluster is formed having seed $\mathbf{x}(k)$. Finally, we describe what happens when cluster splitting is not accepted, i.e., when (6) does not hold. In the later case, the winner neuron is kept united and an additional neuron is formed corresponding to a distinct cluster having seed $\mathbf{x}(k)$.

The procedure described so far is applied for each training pattern presentation. When the training set has been exhausted, the integrity of the cluster associated with each output neuron is tested once more by applying the splitting criterion described above. Having completed the later test, we compute the average distortion (e.g. MSE) at the end of session i as follows:

$$D(i) = \frac{1}{M} \sum_{k=1}^{M} \| \mathbf{x}(k) - \mathbf{w}_{c(k)}^{(i)}(M) \|^2 . \tag{7}$$

If $(D(i-1) - D(i))/D(i) > \epsilon = 0.001$, we proceed to an additional training session.

3. EXPERIMENTAL RESULTS

Four distinct bivariate normal populations have been created. Each one has 1000 2-d patterns (i.e., points). The statistical description of the created populations follows: $\mathcal{P}_1$ is a set of 2-d patterns distributed according to $\mathcal{G}(10.0, 20.0; 0.64, 1.0; 0.8)$; $\mathcal{P}_2$ is a set of 2-d patterns distributed according to $\mathcal{G}(20.0, 20.0; 1.33, 1.0; 0.5)$; $\mathcal{P}_3$ is a set of 2-d patterns distributed according to $\mathcal{G}(23.0, 16.0; 1.8, 1.2; 0.0)$ and $\mathcal{P}_4$ is a set of 2-d patterns distributed according to $\mathcal{G}(10.0, 10.0; 1.0, 1.0; 0.7)$ where $\mathcal{G}(\mu_1, \mu_2;\ \sigma_1, \sigma_2;\ r)$ denotes a bivariate normal distribution. Parameters μ_i and σ_i, $i = 1, 2$ are the expected values and the standard deviations along each dimension respectively, and r denotes the correlation coefficient.

The performance of a modified LVQ algorithm that implements the proposed split-merge criteria has been tested against that of a standard LVQ. The modified LVQ network has two output neurons initially. The number of output neurons for standard LVQ neural network is set to five. Both neural networks have been trained by the same set. The training set is formed by selecting randomly 10% of the patterns that belong to each population. Three training sessions are required for both NN to converge. At the end of the learning phase, the modified LVQ NN results in five output neurons. Each neuron is associated with a cluster where all training patterns that come from a distinct population have been included. In other words, although the training is unsupervised, we have obtained perfect classification. On the contrary standard LVQ results in three activated output neurons. In Table 1, the learning and the recall MSE are summarized.

REFERENCES

1. T.K. Kohonen, "The Self-Organizing Map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, September 1990.

2. K.V. Mardia, J.T. Kent and J.M. Bibby, *Multivariate Analysis.* London: Academic Press, 1979.

3. G.A.F. Seber, *Multivariate Observations.* New York: J. Wiley, 1984.

4. R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis.* New York: J. Wiley, 1973.

5. C. Kotropoulos, E. Augé and I. Pitas, "Two-Layer Learning Vector Quantizer for Color Image Quantization," in *Signal Processing VI: Theories and Applications* (J. Vandewalle, R. Boite, M. Moonen and A. Oosterlinck, eds.), pp. 1177–1180, Elsevier, 1992.

Table 1
Learning and recall Mean-Squared Error

Mean-Squared Error	Split-merge LVQ	Standard LVQ
Learning phase	2.530366	5.599298
Recall phase	2.659161	5.749121