RLVQ determination using OWA operators

Angel Caţaron
Department of Electronics and Computers
Transylvania University of Braşov, Romania
email: cataron@vega.unitbv.ro

Răzvan Andonie
Computer Science Department
Central Washington University, Ellensburg, USA [*]
email: andonie@deltanet.ro

ABSTRACT
Relevance Learning Vector Quantization (RLVQ) (introduced in [1]) is a variation of Learning Vector Quantization (LVQ) which allows a heuristic determination of relevance factors for the input dimensions. The method is based on Hebbian learning and defines weighting factors of the input dimensions which are automatically adapted to the specific problem. These relevance factors increase the overall performance of the LVQ algorithm. At the same time, relevances can be used for feature ranking and input dimensionality reduction.

We introduce a different method for computing the relevance of the input dimensions in RLVQ. The relevances are computed on-line as Ordered Weighted Aggregation (OWA) weights. OWA operators are a family of mean type aggregation operators [2]. The principal benefit of our OWA-RLVQ algorithm is that it connects RLVQ to the mathematically consistent OWA models.

KEY WORDS
machine learning, learning vector quantization, ordered weighted aggregation, feature ranking

1 Introduction

Kohonen introduced LVQ [3] as a method to define a classification based on a number of patterns from the trainig set. The vector quantization is a mapping from the n-dimensional space of the feature vectors into a finite space of n-dimensional vectors referred to as codebook. The vectors of the codebook are the prototypes and are labeled with the label of the class they represent. LVQ iteratively adapts codebook vectors to implement desired classification optimizing a global criteria based on the Euclidian distance. Various modifications to the basic algorithm where proposed [3] to ensure a faster convergence (OLVQ) or for better adaptation of the borders (LVQ2, LVQ3).

RLVQ is a modified version of LVQ which introduces relevance factors for features. It uses a modified metric by assigning a weight to each dimension of the vectors. Thus, an additional procedure iteratively adapts the scaling factors with a Hebbian learning technique, giving an overview of the influence of individual features in the classification process.

[*]On leave of absence from the Department of Electronics and Computers, Transylvania University, Brasov, Romania

OWA operators are a class of aggregation operators that provide an aggregation based on the reordering of the criteria that must be satisfied. The weights of the OWA operator are associated to the ordered position of these criteria, not to a particular one [2].

We introduce the OWA-RLVQ algorithm as a new method for computing the relevances of the input features considering them as OWA weights. The relevances are adapted on-line, in common with prototypes training. The metric we use is a particular case of the ordered weighted generalized mean [4] and we use it as an alternative to the Euclidean metric in the LVQ algorithm. Our method makes an interesting connection between two different approaches: RLVQ and OWA. We have obtained a good recognition accuracy on several standard datasets. Meanwhile, the technique can be used for ranking the input features and reducing the dimensionality of the input space.

2 Relevance learning vector quantization

Sometimes, not all the features of the input vectors have the same importance in the decision of a classification system. Some of them can prove to be more influent than others, therefore a feature relevance vector can be computed. RLVQ is an iterative method based on Hebbian learning that introduces weighting factors of the inputs. This is a heuristical algorithm and a number of improvements were proposed [5], [6] in order to avoid unstable behaviour in particular situations.

RLVQ is a supervised algorithm that stronger reinforces the weights of the features that most contribute to the correct classification than the weights of the features having a negative influence. The clustering is realized by a set of prototypes that are tuned by the incoming feature vectors and a standard LVQ algorithm.

Assume that a clustering of data into C classes is to be learned and a set of training data is given:

$$X = \{(\mathbf{x}_i, \mathbf{y}_i) \subset \mathbf{R}^n \times \{1, \ldots, C\} \mid i = 1, \ldots, M\}.$$

The components of a vector $\mathbf{x}_i$ are $[x_{i1}, \ldots, x_{in}]$.

LVQ chooses prototype vectors in $\mathbf{R}^n$ for each class, so called *codebooks*. Denote the set of all codebooks by $\{\mathbf{w}_1, \ldots, \mathbf{w}_K\}$. The components of a vector $\mathbf{w}_j$ are $[w_{j1}, \ldots, w_{jn}]$. The training algorithm adapts the codebooks for as least as possible quantization error on all feature vectors, as follows [3]:

1. For a given input $\mathbf{x}_i$, find the closest codebook, $\mathbf{w}_j$, the winner, which provides the least value of the distance $\|\mathbf{x}_i - \mathbf{w}_j\|$. If $\mathbf{x}_i$ and $\mathbf{w}_j$ have the same class label, the feature vector is correctly classified.

2. Update the winner codebook:

$$\mathbf{w}_j = \begin{cases} \mathbf{w}_j + \eta(\mathbf{x}_i - \mathbf{w}_j) & \text{if } \mathbf{x}_i \text{ was correctly classified} \\ \mathbf{w}_j - \eta(\mathbf{x}_i - \mathbf{w}_j) & \text{otherwise} \end{cases}$$

   where $\eta > 0$ is the learning rate.

RLVQ uses a modified weighted metric in this algorithm:

$$\|\mathbf{x}_i - \mathbf{w}_j\|_\lambda = \sqrt{\sum_{k=1}^{n} \lambda_k (x_{ik} - w_{jk})^2}$$

where $\lambda = [\lambda_1, \ldots, \lambda_n]$ is the relevance vector and $\sum_{k=1}^{n} \lambda_k = 1$. Following a similar rule, the weighting factors are iteratively adapted [1]:

1. $\lambda_k = \begin{cases} \max\{\lambda_k - \alpha|x_{ik} - w_{jk}|, 0\} & \text{if } \mathbf{x}_i \text{ was correctly classified} \\ \lambda_k + \alpha|x_{ik} - w_{jk}| & \text{otherwise} \end{cases}$

   for $k = 1, \ldots, n$. $\alpha > 0$ is the learning rate for the weighting factors.

2. Normalize the weights vector.

Relevance determination can be used after LVQ learning, or simultaneously, this second version yielding an online algorithm. Reported results [1] proved a better classification accuracy of RLVQ compared to the standard LVQ.

3 Ordered Weighted Aggregation Operators

An OWA operator [2] of dimension n is a function

$$F : \mathbf{R}^n \to \mathbf{R}$$

such that

$$F(a_1, a_2, \ldots, a_n) = \sum_{i=1}^{n} v_i a_i^*,$$

where $\mathbf{V} = [v_1, v_2, \ldots, v_n]$ is a weighting vector associated to the operator, $v_i \in [0, 1]$, $\sum_{i=1}^{n} v_i = 1$, and $\mathbf{A}^* = [a_1^*, a_2^*, \ldots, a_n^*]$ is the descending reordering of the arguments $\mathbf{A} = [a_1, a_2, \ldots, a_n]$ such that a_i^* is the i^{th} largest of the a_i, $a_1^* \geq a_2^* \geq \ldots \geq a_n^*$.

The OWA operators are aggregation operators, satisfying the commutativity, monotonicity, bounding and idempotency properties [2].

A number of methods for setting the weight vector components' were proposed. Yager [2] suggested a method based on the concept of linguistic quantifiers. O'Hagan [8] introduced a technique based on providing an orness degree for the weighting vector. Based on this information, the values can be obtained by maximizing the dispersion. In [7], [8], Filev and Yager computed the aggregation operator weights by gradient descending method, using a target vector of aggregated values. Torra [9], [10] used a particular aggregation operator named Weighted OWA (WOWA) and determined its parameters in a two steps optimization procedure, providing an ideal outcome for each training pattern. In a recent work, Karayiannis [4] introduced an OWA-based family of learning vector quantization and clustering algorithms.

4 OWA-RLVQ

We aim to introduce an algorithm for computing the OWA weights as relevance factors for supervised clustering problems.

First, we will define the distance between an input vector $\mathbf{x}_i$ and a codebook vector $\mathbf{w}_j$ as an aggregation operator:

$$D_{ij}^* = \sqrt{\sum_{k=1}^{n} \lambda_k (|x_{ik} - w_{jk}|^*)^2},$$

where $\sum_{k=1}^{n} \lambda_k = 1$, and $|x_{ik} - w_{jk}|^*$ is the k^{th} largest difference between corresponding components of the input vector and the codebook vector.

This distance is a special case of the ordered weighted generalized mean [4]:

$$M = \left(\sum_{k=1}^{n} \lambda_k a_k^{*p} \right)^{1/p}$$

where $p \in \mathbf{R}^*$. For $p = 2$, this reduces to our modified distance, where $a_k^* = |x_{ik} - w_{jk}|^*$ and $|x_{i1} - w_{j1}|^* \geq |x_{i2} - w_{j2}|^* \geq \ldots \geq |x_{in} - w_{jn}|^*$.

The LVQ algorithm must be reformulated to minimize an objective function based on this modified distance. We compute

$$\Delta \mathbf{w}_j^* = \eta \lambda (\mathbf{x}_i - \mathbf{w}_j)^* \quad (1)$$

if $\mathbf{x}_i$ was correctly classified according to the D_{ij}^* distance, and

$$\Delta \mathbf{w}_j^* = -\eta \lambda (\mathbf{x}_i - \mathbf{w}_j)^* \quad (2)$$

if $\mathbf{x}_i$ was not correctly classified. We note that $(\mathbf{x}_i - \mathbf{w}_j)^*$ is a reordered vector and the relevances will apply after reordering. Thus, the components of $\mathbf{w}_j^*$ from (1) and (2) will correspond to the resulting positional weight components. The resulting trained codebooks will generally be different than the codebooks obtained by the regular LVQ algorithm, because they reflect a different training approach.

We consider that $\mathbf{x}_i$ is correctly classified if the distance D^*_{ij} to codebook $\mathbf{w}_j$ is minimum for the entire set of codebooks, where the two vectors have the same class label:

$$D^*_{ij} < D^*_{il} \text{ for any } l \neq j.$$

We denote $\mathbf{d} = [d_1, \ldots, d_n]$, where $d_k = x_{ik} - w_{jk}$, $k = 1, \ldots, n$. If $\mathbf{x}_i$ was correctly classified according to D^*_{ij}, a small $|d_k|^*$ should induce a large $\Delta\lambda_k$ and the reverse. Thus, $|d_k|^* < |d_{k'}|^*$ leads to $\Delta\lambda_k > \Delta\lambda_{k'}$, that is $-|d_k|^* > -|d_{k'}|^*$ with $\Delta\lambda_k > \Delta\lambda_{k'}$. We may consider now $\Delta\lambda_k = -\alpha|d_k|^*$, and $\Delta\lambda_k = -\alpha|x_{ik} - w_{jk}|^*$. When the classification is not correct, we can follow a similar path and obtain $\Delta\lambda_k = \alpha|x_{ik} - w_{jk}|^*$.

We use the transform:

$$\lambda_k = \frac{e^{\lambda_k}}{\sum_{i=1}^{n} e^{\lambda_i}}$$

for $k = 1, \ldots, n$, to ensure that $\sum_{k=1}^{n} \lambda_k = 1$ and $\lambda_k \in [0, 1]$.

The following procedure updates on-line both the relevances and the feature ranks.

1. Initialize η and α. Initialize the relevance vector: $\lambda_k = \frac{1}{n}$, $k = 1, \ldots, n$.

2. Initialize the codebooks.

3. Update the codebooks according to the modified LVQ rule, using the D^*_{ij} distance:

$$\mathbf{w}^*_j = \begin{cases} \mathbf{w}^*_j + \eta\lambda(\mathbf{x}_i - \mathbf{w}_j)^* & \text{if } \mathbf{x}_i \text{ was correctly classified} \\ \mathbf{w}_{j^*} - \eta\lambda(\mathbf{x}_i - \mathbf{w}_j)^* & \text{otherwise} \end{cases}$$

4. Update the relevances:

$$\lambda_k = \begin{cases} \lambda_k - \alpha|x_{ik} - w_{jk}|^* & \text{if } \mathbf{x}_i \text{ was correctly classified} \\ \lambda_k + \alpha|x_{ik} - w_{jk}|^* & \text{otherwise} \end{cases}$$

   for $k = 1, \ldots, n$.

5. Transform the relevances:

$$\lambda_k = \frac{e^{\lambda_k}}{\sum_{i=1}^{n} e^{\lambda_i}}$$

   for $k = 1, \ldots, n$.

6. Compute the weight of each feature as an average of its before ordering position index in the input vector, for all previous steps.

7. Repeat steps 3-6 for each training pattern.

Table 1. Feature ranking for the Iris database.

Rank	1	2	3	4
RLVQ Feature	4	2	3	1
OWA-RLVQ Feature	4	3	2	1
OWA-RLVQ Feature Weight	1.86	1.44	1.24	1.17

This algorithm computes relevance OWA weights by minimizing the modified distance which we consider as an aggregated value. We have to distinguish between the relevance and the rank of a feature. The rank is attached to a specific feature, whereas the relevance is attached to a specific position in the ordered vector of distances to the codebooks. This explaines why we have to perform step 6.

5 Experiments

We tested the OWA-RLVQ algorithm on standard benchmarks in order to study the resulted relevance vectors and recognition rates. The Iris database [11] contains 3 classes, 50 vectors each. Two of them are not linearly separable. The problem is to detect the classes based on 4 features. We trained OWA-RLVQ with 6 codebooks, obtaining a recognition rate of 96.6%. The obtained relevance vector was [0.15 0.21 0.23 0.38]. We used $\eta = 0.3$ and $\alpha = 2$. The experiments showed that the most important feature is the last one and the least important is the first one, the same ranking as reported in [1]. In Table 1 we present the ranking resulted after the RLVQ and OWA-RLVQ training, using the same initial set of codebooks.

The Vowel Recognition database (Deterding data) [11] contains vectors extracted from 15 individual speakers pronouncing vowels in 11 contexts, 6 times each. The problem is to use the pronounciations of the first 8 speakers for training and of the last 7 speakers for tests. We obtained the following relevance vector: [0.032 0.039 0.043 0.044 0.077 0.136 0.137 0.15 0.163 0.173], and a recognition rate of 46.75%. The LVQ recognition rate for this experiment was 44.8%, and the RLVQ recognition rate was 46.32%. The problem is known as difficult and we used 59 codebooks. In the OWA-RLVQ experiments we set $\eta = 1.7$ and $\alpha = 1.9$. Table 2 describes the ranking resulted for the Vowel Recognition database. Feature 2 is ranked as most important by RLVQ and the second most important by OWA-RLVQ. Feature 10 is ranked as the least important by both models.

The Ionosphere dataset [11] consists of 351 instances of radar collected data, with 34 continuous attributes each. The vectors are labeled with ”bad” or ”good”, being a binary classification task. We used the first 200 instances, balanced between positive and negative examples, for train-

Table 2. Feature ranking for the Vowel Recognition dataset.

Rank	1	2	3	4	5
RLVQ Feature	2	5	1	9	6
OWA-RLVQ Feature	8	2	4	5	6
OWA-RLVQ Feature Weight	5.209	5.209	5.2	5.17	5.16
Rank	6	7	8	9	10
RLVQ Feature	3	4	8	7	10
OWA-RLVQ Feature	9	3	7	1	10
OWA-RLVQ Feature Weight	5.15	5.15	5.14	5.14	5.13

Table 3. Feature ranking for the Ionosphere database. Only the five most important features are represented.

Rank	1	2	3	4	5
RLVQ Feature	20	28	26	12	6
OWA-RLVQ Feature	14	12	1	3	28
OWA-RLVQ Feature Weight	19.2	19.16	19.1	19.07	19.06

ing, and the remaining 151 for tests. We used 8 codebooks, and the values of the OWA-RLVQ training parameters were $\eta = 3.3$ and $\alpha = 3.5$. The obtained recognition rate (93.37%) was better than the LVQ and the RLVQ rates (90.06%, respectively 92.71%). The typical rates reported in [11] start from 0.9. Table 3 presents the ranking of the most important 5 features.

Table 4 compares the recognition rates obtained with LVQ, RLVQ, and OWA-RLVQ, using each time the same set of initial codebooks. We have generally obtained better results with our OWA-RLVQ algorithm.

Constant feature values over all the training set may lead to an undesired reinforcement of the corresponding relevance and this is a known drawback of RLVQ too. Therefore, some preprocessing is necessary to eliminate constant features. As the training progresses, the relevances adapt to the data.

Table 4. Comparative recognition rates obtained with LVQ, RLVQ and OWA-RLVQ.

Database	LVQ	RLVQ	OWA-RLVQ
Iris	91.33%	95.33%	96.6%
Vowel	44.8%	46.32%	46.75%
Ionosphere	90.06%	92.71%	93.37%

6 Conclusions and Future Work

We introduced a method for computing the relevances of the input dimensions as OWA weights. The proposed algorithm computes on-line the relevances, giving the possibility of dynamic adaptation to the incoming data. The experimental results showed that the supplementary functionality introduced by OWA-RLVQ offers a good recognition rate. The relevance factors can be used for on-line feature ranking. A straightforward application of feature ranking is feature selection in order to reduce input space dimensionality.

Our OWA-RLVQ is different than the OWA-based LVQ algorithms introduced by Karayiannis [4], since we modify the LVQ metric following the RLVQ strategy.

It is interesting to note that our approach is a new method for computing OWA weights. Compared to the gradient descending approach described in [7], [8], this is performed in a dynamic way. We aim to investigate further this aspect.

References

[1] T. Bojer, B. Hammer, D. Schunk, K.T. von Toschanowitz, Relevance Determination in Learning Vector Quantization, in M.Verleysen (Ed.) *European Symposium on Artificial Neural Networks'2001* (D-side publications, 2001) 271–276.

[2] R.R. Yager, On Ordered Weighted Averaging Aggregation Operators in Multicriteria Decisionmaking, *IEEE Trans. Systems, Man, and Cybernetics*, 18, 1988, 188–190.

[3] T. Kohonen, *Self-Organizing Maps* (Springer-Verlag, 1997).

[4] N.B. Karayiannis, Soft Learning Vector Quantization and Clustering Algorithms Based on Ordered Weighted Aggregation Operators, *IEEE Trans. Neural Networks*, 11, 2000, 1093–1105.

[5] B. Hammer, T. Villmann, Batch-RLVQ, in M.Verleysen (Ed.) *European Symposium on Artificial Neural Networks'2002* (D-side publications, 2002) 295–300.

[6] B. Hammer, T. Villmann, Generalized Relevance Learning Vector Quantization, *Neural Networks*, 15, 2002, 1059–1068.

[7] D. Filev, R.R. Yager, On the issue of obtaining OWA operator weights, *Fuzzy Sets and Systems*, 94, 1998, 157–169.

[8] R.R. Yager, D. Filev, Induced Ordered Weighted Averaging Operators, *IEEE Trans. Systems, Man, and Cybernetics*, 29, 1999, 141–150.

[9] V. Torra, Learning Weights for Weighted OWA Operators, *Proc. of the IEEE Int. Conference on Industrial Electronics, Control and Instrumentation (IECON 2000)*, Nagoya, Japan, 2000.

[10] V. Torra, Weighted OWA Operators for Synthesis of Information, *Proc. of the 5th IEEE Int. Conference on Fuzzy Systems*, New Orleans, USA, 2000, 996–971.

[11] C.L. Blake, C.J. Merz, *UCI Repository of machine learning databases*, University of California, Irvine, Dept. of Information and Computer Sciences, http://www.ics.uci.edu/∼mlearn/MLRepository.html, 1998.

[12] D. Filev, R.R. Yager, Learning OWA operator weights from data, *Proceedings of the Third IEEE International Conference on Fuzzy Systems*, Orlando, IEEE Press, 1994, 468–473.

[13] N.B. Karayiannis, An Axiomatic Approach to Soft Learning Vector Quantization and Clustering, *IEEE Trans. Neural Networks*, 10, 1999, 1153–1165.

[14] N.B. Karayiannis, M.M. Randolph-Gips, Soft Learning Vector Quantization and Clustering Algorithms Based on Non-Euclidian Norms: Multinorm Algorithms, *IEEE Trans. Neural Networks*, 14, 2003, 89–102.

[15] R.K. De, N.R. Pal, S.K. Pal, Feature Analysis: Neural Network and Fuzzy Set Theoretic Approaches, *Pattern Recognition*, 10(30), 1997, 1579–1590.