Computing OWA weights as relevance factors

Angel Caţaron
Department of Electronics and Computers, TRANSILVANIA University of Braşov, Romania
e-mail: cataron@vega.unitbv.ro

Răzvan Andonie
Computer Science Department, Central Washington University, Ellensburg, USA*
e-mail: andonie@cwu.edu

Abstract **– Ordered Weighted Aggregation (OWA) operators represent a distinct family of aggregation operators and were introduced by Yager in [1]. They compute a weighted sum of a number of criteria that must be satisfied. The central element of the OWA operators is that the criteria are reordered before aggregation and therefore a particular weight is associated to a position.**

Relevance Learning Vector Quantization (RLVQ) [2] is an extension of the Learning Vector Quantization (LVQ) algorithm [3] and performs a heuristic determination of the relevance factors of the input dimensions. This method is based on Hebbian learning and associates a weight factor to each dimension of the input vectors.

We present a LVQ method for on-line computing of the OWA weights as relevance factors. The method uses a weighted metric based on OWA restrictions. The principal benefit of our algorithm is that it connects two distinct topics: RLVQ algorithm and the consistent mathematical model of the OWA operators.

Index Terms **– ordered weighted aggregation operators, learning vector quantization, relevance factors, machine learning, neural networks.**

I. INTRODUCTION

OWA operators represent a class of aggregation operators that provide an aggregated value based on a reordering of the criteria that must be satisfied. The weights of the OWA operator are associated to a position of the reordered arguments and not to a specific value.

LVQ is a method of classification based on a number of patterns. The vector quantization defines a mapping from a space of *n*-dimensional vectors into a finite set of *n*-dimensional vectors referred to as codebook. The vectors from the codebook are the prototypes and each of them is assigned to a particular class. LVQ implements an algorithm that iteratively adapts the codebook vectors by optimizing global criteria based on the Euclidian distance. A number of modifications of standard LVQ algorithm were proposed in order to ensure a faster convergence (OLVQ) or for a better adaptation along the borders (LVQ2, LVQ3) [3].

The standard LVQ algorithm does not make a distinction between the more or less influent features of the input vectors. The Distinction Sensitive Learning Vector Quantization (DSLVQ) algorithm introduced in [18] employs a weight for each feature and uses a weighted distance for classification. It uses a heuristically iterative algorithm to adapt the weights to problems' requirements: reduces the influence of the features that frequently lead to a wrong classification and amplify the influence of the features that have a large contribution to a correct classification. RLVQ is a variation of LVQ, similar to DSLVQ, that introduces the relevance factors for each feature.

We present our OWA-RLVQ algorithm as a method to compute the OWA weights as relevance factors in parallel with defining a classification based on a modified LVQ algorithm. This method connects two different approaches: RLVQ and OWA. We have obtained good recognition rates on several standard datasets. OWA-RLVQ can also be used as a technique for ranking the input vector's features. A preliminary version of the algorithm appeared in [10], however without the OWA details and without some of the experiments described below.

---
* On leave of absence from the Department of Electronics and Computers, Transylvania University of Braşov.

II. OWA OPERATORS

We aim to describe first the general aggregation problem. Then we will introduce the OWA operators and their fundamental properties.

Let us consider that $C_1,...,C_n$ are n criteria which define a multicriterial problem. We denote by X the domain of the values of these criteria and by I the interval [0,1]. The aggregation problem means to formulate a global decision function D. It has the property that, for any alternative $x \in X$, the value $D(x) \in I$ reflects the degree that x meets the required conditions in respect with the n criteria, when $C_i(x) \in I$, $i = 1,...,n$ is the degree that x satisfies the criteria C_i. We can therefore write the following relation:

$$D(x) = F(C_1(x),...,C_n(x)) .$$

F represents the aggregation operator which must be [1]:

a) Monoton, that is the more an individual criteria satisfied, the bigger the global decision function's value:
$$C_i(x) \geq C_i(y) \; \forall i = 1,...,n \text{ and } x, y \in X \Rightarrow D(x) \geq D(y)$$

b) Symmetric, that is the order of the criteria is not important for the global decision function's value.

The analysis of an aggregation operator means to study the relations between the criteria that describe the problem. An extreme is when x must satisfy all the $C_1,...,C_n$ criteria and we have an *anding* applied to the n values. An obvious example of such an operator is $D(x) = Min(C_1(x),...,C_n(x))$. Another extreme is when at least a criterion must be satisfied and this is an *oring* of the n values. An operator who belongs to this category is $D(x) = Max(C_1(x),...,C_n(x))$. The usual aggregation operators are between these two extreme cases.

Special categories of aggregation operators are the Ordered Weighted Aggregation (OWA) operators. An *OWA operator* is an n-dimensional function:

$$F: R^n \to R$$
$$F(a_1,...,a_n) = \sum_{i=1}^{n} w_i a_i^*$$

where $\mathbf{W} = [w_1 ... w_n]^t$ is the weights vector associated to the operator, with $w_i \in [0,1]$, $\sum_{i=1}^{n} w_i = 1$ and $\mathbf{A}^* = [a_1^* ... a_n^*]^t$ an ascending reordering of the arguments of $\mathbf{A} = [a_1 ... a_n]^t$ such as a_i^* is the largest i^{th} a_i.

The central element in using the OWA operators is the reordering step. A particular argument a_i will no longer be associated to a particular w_i, but to the value from the position *i* resulted after reordering.

It can be shown that an OWA operator is commutative, monotone and idempotent [1]. It also has the bounding property:

$$Min(a_1,...,a_n) \leq F(a_1,...,a_n) \leq Max(a_1,...,a_n) .$$

Based on this property, Yager [1] introduced a measure called *orness* which is close to 1 if the operator has an *or* character:

$$O(\mathbf{W}) = \frac{1}{n-1} \sum_{i=1}^{n} (n-i) w_i .$$

A number of methods were proposed to choose the OWA operator's weights. O'Hagan [4] introduced a technique based on a given *orness* value. Torra [5], [6] used a particular aggregation operator named Weighted OWA and determined its parameters by using a procedure that computes an ideal output for each training pattern. Karayiannis [7] used two OWA weights families containing a set of equal weights and a set of linear descending values. Beliakov [8] approximated the OWA operators by solving a problem named Least Squares with Equality and Inequality (LSEI). Filev and Yager [4], [9] computed the OWA operator's weights by using a descending gradient method.

We will present a method to determinate the OWA weights by considering them as relevance factors.

III. RELEVANCE LVQ

It is known that sometimes not all features of the input vector have the same influence in the decision of a classification or a recognition system. The Relevance Learning Vector Quantization (RLVQ) algorithm computes a set of relevances associated to each feature of the input vector. This is an iterative method based on Hebbian learning. It is a heuristically algorithm and a number of improvements were proposed in order to avoid unstable behavior in some particular situations.

The RLVQ algorithm reinforces the relevance factors of the features that have the highest influence for the correct classification. This algorithm decreases the weights of the features that have a negative influence over the recognition process. The clustering is realized by a set of prototypes that are tuned by the incoming feature vector 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:

$$X = \left\{ (\mathbf{x}_i, \mathbf{y}_i) \subset R^n \times \{1,...,C\} \middle| i = 1,..., M \right\}.$$

The components of a vector $\mathbf{x}_i$ are $[x_{i1},...,x_{in}]^t$. LVQ chooses prototype vectors in R^n for each class, so called *codebook vectors*. Denote the set of all codebook vectors by $\{\mathbf{w}_1,...,\mathbf{w}_K\}$. The components of a vector $\mathbf{w}_j$ are $[w_{j1},...,w_{jn}]^t$. The training algorithm adapts the codebook vectors 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 vector $\mathbf{w}_j$, 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:

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

where $\lambda = [\lambda_1,...,\lambda_n]^t$ is the relevance vector and $\sum_{k=1}^{n} \lambda_k = 1$. Following a similar rule, the weighting factors are iteratively adapted [2]:

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 $k = 1,...,n$. $\alpha > 0$ is the learning rate for the weighting factors.
2. Normalize the weight vectors.

Relevance determination can be used after LVQ learning or simultaneously, this second version is yielding an on-line algorithm. Reported results [2] proved a better recognition accuracy of RLVQ compared to the standard LVQ.

IV. OWA – RELEVANCE LVQ ALGORITHM

We present now the OWA-RLVQ algorithm for computing the OWA weights as relevance factors simultaneously with the update of the codebook vectors.

We redefine first the LVQ algorithm by replacing the Euclidian distance with a weighted distance:

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

The relevances vector, λ, has the properties of the OWA operator's weights:

$$\sum_{k=1}^{n} \lambda_k = 1, \forall \lambda_1,...,\lambda_n \in [0,1].$$

By the expression:

$$|x_{ik} - w_{jk}|^*$$

we denoted the k^{th} largest difference between corresponding components of the input vector $\mathbf{x}_i$ and the codebook vector $\mathbf{w}_j$.

The distance D_{ij}^* is a particular case of the ordered weighted generalized mean [7]:

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

where p is a real, positive number. By replacing $p = 2$ we obtain our modified distance where a_k^* corresponds to the difference $|x_{ik} - w_{jk}|^*$ and:

$$|x_{x1} - w_{j1}|^* \geq |x_{i2} - w_{j2}|^* \geq ... \geq |x_{in} - w_{jn}|^*.$$

We can reformulate the LVQ algorithm by minimizing an objective function based on the modified distance. The codebook vectors can be updated by computing

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

if $\mathbf{x}_i$ was correctly classified according to the D_{ij}^* distance and $\mathbf{I}$ is the unit diagonal matrix. In the case when the input vector $\mathbf{x}_i$ was not correctly classified, according to the same distance we use the following update formula:

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

In these update relations we denoted with $(\mathbf{x}_i - \mathbf{w}_j)^*$ a vector obtained by reordering of its components. The relevances are used after the reordering step. As OWA operator's properties suggest, the components of the vectors $\mathbf{w}_j^*$ will correspond to the resulting positional weight components.

We established the method to adapt the prototype vectors considering the modified distance D_{ij}^*. We will see now how we can update the relevance factors.

The modified distance D_{ij}^* is a criterion to decide the correct classification of the input vectors. We consider that $\mathbf{x}_i$ is correctly classified if our modified distance to the codebook $\mathbf{w}_j$ is minim and the two vectors belong to the same class:

$$D_{ij}^* < D_{il}^*, \forall l \neq j .$$

Denote

$$\mathbf{d} = \left[d_1, \ldots, d_n\right]^t$$

where

$$d_k = x_{ik} - w_{jk}, k = 1, \ldots, n .$$

In the case when $\mathbf{x}_i$ is correctly classified according to the modified distance D_{ij}^*, a small value of $\left|d_k\right|^*$ should lead us to a large value of $\Delta\lambda_k$. On the other hand, a large value of the distance $\left|d_k\right|^*$ should have a small influence for the relevance values and the magnitude $\Delta\lambda_k$ is minimal. Therefore,

$$\left|d_k\right|^* < \left|d_{k'}\right|^* \Leftrightarrow \Delta\lambda_k > \Delta\lambda_{k'}$$

that is

$$-\left|d_k\right|^* > -\left|d_{k'}\right|^* \Leftrightarrow \Delta\lambda_k > \Delta\lambda_{k'} .$$

If we consider:

$$\Delta\lambda_k = -\alpha\left|d_k\right|^*$$

then we can write:

$$\Delta\lambda_k = -\alpha\left|x_{ik} - w_{jk}\right|^* .$$

When the classification of the input vector $\mathbf{x}_i$ is not correct according to the modified distance D_{ij}^*, the update formula can be developed with a similar method. A small value of $\left|d_k\right|^*$ induces a small value of $\Delta\lambda_k$ and a large value of $\left|d_k\right|^*$ corresponds to a large value of $\Delta\lambda_k$. This means that

$$\left|d_k\right|^* > \left|d_{k'}\right|^* \Leftrightarrow \Delta\lambda_k > \Delta\lambda_{k'}$$

and

$$\Delta\lambda_k = \alpha\left|d_k\right|^*$$

or

$$\Delta\lambda_k = \alpha\left|x_{ik} - w_{jk}\right|^* .$$

Because the relevance factors are weights of an OWA operator, we finally use the following transform:

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

which ensure us that $\sum_{k=1}^{n} \lambda_k = 1$ and $\lambda_k \in \left[0,1\right]$.

We are ready now to write the procedure that simultaneously adapts the codebook vectors and the relevance factors.

1. Initialize the learning rates η and α. Assigning the initial values to the relevance vector: $\lambda_k = \frac{1}{n}, k = 1, \ldots, n$.
2. Initialize the codebook vectors.
3. Update the codebook vectors using the modified LVQ algorithm which uses the distance D_{ij}^*:
$$\mathbf{w}_j^* = \begin{cases} \mathbf{w}_j^* + \eta\lambda\mathbf{I}\left(\mathbf{x}_i - \mathbf{w}_j\right)^*, & \text{if } \mathbf{x}_i \text{ was correctly classified} \\ \mathbf{w}_j^* - \eta\lambda\mathbf{I}\left(\mathbf{x}_i - \mathbf{w}_j\right)^*, & \text{otherwise} \end{cases} .$$
4. Update the relevance factors:
$$\lambda_k = \begin{cases} \lambda_k - \alpha\left|x_{ik} - w_{jk}\right|^*, & \text{if } \mathbf{x}_i \text{ was correctly classified} \\ \lambda_k + \alpha\left|x_{ik} - w_{jk}\right|^*, & \text{otherwise} \end{cases}$$
for all $k = 1, \ldots, n$.
5. Normalize relevances:
$$\lambda_k = \frac{e^{\lambda_k}}{\sum_{i=1}^{n} e^{\lambda_i}}, k = 1, \ldots, n$$
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.

This algorithm computes the relevance factors that are attached to a specific position in the ordered vector of distances to the codebook vectors. It also computes the rank of each feature and theses values have a different meaning. It is attached to a specific feature and this explains the step 6.

V. EXPERIMENTS

We used standard benchmarks [13] to test the OWA-RLVQ algorithm: Iris database, Vowel Recognition database (Deterding data) and Ionosphere dataset. We studied the recognition rates and the resulted relevance vectors that can also be interpreted as OWA operator's weights.

The Iris database contains 3 classes of patterns with 50 vectors each. Two classes are not linearly separable. The problem is to detect the classes considering the 4 features of each vector. We used in our training procedure 6 codebook vectors and we finally obtained a recognition rate of 96.60% and the following relevance vector: $[0.15\ 0.21\ 0.23\ 0.38]^t$. For the learning constants we used the values $\eta = 0.3$ and $\alpha = 2$. The feature ranking, depicted in Table 1, reflected the same results as obtained in [1] for RLVQ, with the last feature considered as the most important.

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

The Vowel Recognition database contains vectors extracted from 15 individual speakers pronouncing vowels in 11 contexts, 6 times each. The problem is to use the pronunciations of the first 8 speakers for training and the pronunciations of the last 7 speakers for recognition tests. We used 59 codebook vectors and the accuracy rate that we obtained in this experiment was 46.75% comparing to 46.32% obtained with RLVQ and 44.8% with LVQ. The values of the learning rates were $\eta = 1.7$ and $\alpha = 1.9$. We obtained the following relevance vector: $[0.032\ 0.039\ 0.043\ 0.044\ 0.077\ .0136\ .0137\ 0.150\ .0163\ .0.173]^t$. The feature number 2 was ranked as the second most important by OWA-RLVQ and as most important by RLVQ, as described in Table 2.

The Ionosphere dataset consists of 351 instances of radar collected data, with 34 continuous features each. The vectors, balanced between positive and negative examples, are labeled with "bad" or "good", this yielding a binary classification task. The first 200 patterns were used for training and the remaining 151 we used for the recognition tests. By training 8 codebook vectors, we obtained a recognition rate of 93.37% with OWA-RLVQ, of 92.71% with RLVQ and of 90.06% with LVQ. In Table 3 we present the ranking of the most important 5 features as resulted from our experiments. We used the values $\eta = 3.3$ and $\alpha = 3.5$ for the learning parameters.

Table 2. Feature ranking for the Vowel Recognition database.

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	5.20	5.20	5.20	5.17	5.16
RLVQ Feature	6	7	8	9	10
OWA-RLVQ Feature	3	4	8	7	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 were 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	19.1	19.1	19	19

A comparison of the recognition rates between LVQ, RLVQ and OWA-RLVQ is provided in Table 4, reflecting that our algorithm performed better in all experiments reported here.

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

Database	LVQ	RLVQ	OWA-RLVQ
Iris	91.33%	95.33%	96.60%
Vowel	44.80%	46.32%	46.75%
Ionosphere	90.06%	92.71%	93.37%

VI. CONCLUSIONS

We have presented a method to compute the OWA operator's weights as relevance factors of the input features. The OWA-RLVQ algorithm uses a modified weighted metric. The relevance vector is updated on-line, giving the possibility of dynamical adaptation to the incoming data.

We have obtained good recognition rates when applying our algorithm on standard benchmarks.

The relevance factors can be used as OWA weights. They can also be used for on-line feature ranking and for feature selection.

REFERENCES

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

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

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

[4] R.R. Yager and D. Filev, *Induced Ordered Weighted Averaging Operators*. IEEE Trans. on System, Man, and Cybernetics, 29, 141-150, 1999.

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

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

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

[8] G. Beliakov, *How to Build Aggregation Operators From Data*. International Journal of Intelligent Systems, 18, 903-923, 2003.

[9] D. Filev and R.R. Yager, *On the Issue of Obtaining OWA Operator Weights*, .Fuzzy Sets and Systems, 94, 157-169, 1998.

[10] A. Caţaron and R. Andonie, *RLVQ Determination Using OWA Operators*. In M. Hamza (Ed.) Proceedings of the 3rd IASTED International Conference on Artificial Intelligence and Applications (ACTA Press) 434-438, 2003.

[11] B. Hammer and T. Villmann, *Batch-RLVQ*. In M. Verleysen (Ed.). European Symposyum on Artificial Neural Networks'2002 (D-side publications), 295-300, 2002.

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

[13] C.L. Blake and C.J. Merz, *UCI Repository of Machine Learning Database*, University of California, Irvine, Dept. of Information and Computer Sciences, http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998.

[14] D. Filev and R.R. Yager, *Learning OWA Operator Weights from Data*, Proceedings of the Third IEEE International Conference on Fuzzy Systems, Orlando, IEEE Press, 468-473, 1994.

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

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

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

[18] M. Pregenzer, D. Flotzinger and G. Pfurtscheller, *Distinction Sensitive Learning Vector Quantization – A Noise-Insensitive Classification Method*, Proceedings of the IEEE International Conference on Neural Networks (IJCNN 1994), Orlando, Florida, 2890-2894, 1994.