ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 471-476

An Informational Energy LVQ Approach for Feature Ranking

Răzvan Andonie [1] and Angel Caţaron [2]

[1]Computer Science Department, Central Washington University, USA

[2]Electronics and Computers Department, Transylvania University, Romania

Abstract. Input feature ranking and selection represent a necessary preprocessing stage in classification, especially when one is required to manage large quantities of data. We introduce a weighted LVQ algorithm, called Energy Relevance LVQ (ERLVQ), based on Onicescu's informational energy [10]. ERLVQ is an incremental learning algorithm for supervised classification and feature ranking.

1 Introduction

Standard Learning Vector Quantization (LVQ) [9] does not discriminate between more or less informative features: their influence to the distance function is equal. On the contrary, the Distinction Sensitive Learning Vector Quantizer (DSLVQ), introduced by Pregenzer et al. [11], holds a changeable weight (relevance) value for every feature and employs a weighted distance function for classification. This function is based on the Euclidean approach, but other weighted distance functions could also be applied. An iterative heuristic training process is used to tune the weight values for a specific problem: the influence of features which frequently contribute to miss classifications of the system is reduced while the influence of the very reliable features is increased. This facilitates class discrimination and makes the system less sensitive to noise. A DSLVQ variation is Relevance LVQ (RLVQ), introduced in [3].

In DSLVQ and RLVQ, the updating of the relevances is only heuristically motivated. For this reason, an improved RLVQ has been proposed by Hammer et al. [7], which obeys a stochastic gradient descent on an energy function. Another approach is OWA-RLVQ [4], with the relevances computed as Ordered Weighted Aggregation (OWA) weights.

We will include these LVQ variations in a general LVQ algorithm class, called "weighted LVQ". A *weighted LVQ* algorithm is characterized by the following features: *a)* the input features are weighted and the weights are used in the distance function; *b)* relevances and codebook vectors are updated in

ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 471-476

parallel, during the learning phase. The updating can be performed incrementally, after processing each learning sample; *c)* the resulted relevances may be regarded as feature ranks and used for dimensionality reduction of the input space (i.e., feature selection).

Our aim is to bring weighted LVQ type algorithms into a different theoretical framework. We introduce ERLVQ, a weighted LVQ algorithm based on mutual information (MI) optimization. A sensible part of this approach is the estimation of MI because of its high computational complexity. In order to avoid this drawback, Principe et al. [12], and Torkkola [13] have recently developed expressions for approximating MI based on measures similar to Renyi's quadratic entropy. A similar approach can be found in [8]. Rather than using Renyi's entropy, we estimate MI using Onicescu's informational energy [10]. Our estimation is used in conjunction with a weighted LVQ algorithm. It updates incrementally both the codebook vectors and the feature relevances. The relevances can be used for feature ranking.

After introducing the basic notations in Section 2, we describe in Section 3 our MI estimation procedure and the ERLVQ algorithm. In Section 4 we compare ERLVQ to other weighted LVQ algorithms and finally, in Section 5, we conclude with some closing remarks.

2 The weighted LVQ algorithm

We aim to introduce firstly the basic notations used in LVQ, and to describe the general weighted 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 *codebook vectors* in $\mathbf{R}^n$ for each class. Denote the set of all codebook vectors by $\{\mathbf{w}_1, \ldots, \mathbf{w}_K\}$. The components of a vector $\mathbf{w}_j$ are $[w_{j1}, \ldots, w_{jn}]$.

The distance between $\mathbf{x}_i$ and $\mathbf{w}_j$ is determined by a weighted distance function $dist(\mathbf{x}_i, \mathbf{w}_j, \lambda)$, employing the feature weights vector $\lambda = [\lambda_1, \ldots, \lambda_n]$, $\sum_{k=1}^{n} \lambda_k = 1$. The weight, or *relevance*, λ_k measures the importance of the k-th feature in the supervised classification of the input vector $\mathbf{x}_i$. The *(overall) rank* of the k-th feature at step i of the LVQ training is an average of all λ_k, for $k = 1, 2, \ldots, i$.

A weighted LVQ algorithm learns $\mathbf{x}_i$ by performing the following steps:

1. Find the closest codebook, $\mathbf{w}_j$, the winner, which provides the minimum value of $dist(\mathbf{x}_i, \mathbf{w}_j, \lambda)$, for $j = 1, 2, \ldots, K$.
2. Update the winner codebook according to the LVQ algorithm [9].
3. Update the relevance vector λ according to some rule, and normalize it.

ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 471-476

4. Update the rank of each feature as an average over the previous steps.

This steps can be used in an incremental supervised LVQ learning algorithm for the set X.

3 The Energy Relevance LVQ algorithm

In feature selection algorithms, the MI can be used for evaluating the "information content" of each individual feature with regard to the output class. The feature selection method is searching for a subset of relevant features from an initial set of available features. The subset should maximize MI.

The MI, $I(Y,X) = H(Y) - H(Y|X)$, measures the dependence between two random variables X and Y using Shannon's entropy. Rather than using this formula, we should look at some more convenient ways to express it, according to [12]. We use Onicescu's informational energy [10]. For a discrete random variable X with probabilities p_k, the informational energy was defined as $E(X) = \sum_{k=1}^{n} p_k^2$.

We will first show that there is a similarity between $I(Y,X)$ and the dependence measure $o(Y,X) = E(Y|X) - E(Y)$, introduced in [1]. The application o has the following properties: *i)* o is not symmetrical with respect to its arguments; *ii)* $o(X,Y) \geq 0$ and equality holds iff X and Y are independent; *iii)* $o(X,Y) \leq 1 - E(X)$ and equality holds iff X is completely dependent on Y.

The value $o(X,Y)$ measures the unilateral dependence of X relative to Y, whereas $I(Y,X)$ measures the interdependence of X and Y (it is symmetrical). Generally, both unilateral and bilateral measures are useful for characterizing stochastic systems [1]. We will use the unilateral measure o.

Let us consider the transform $\mathbf{y}_i = \lambda(\mathbf{x}_i - \mathbf{w}_j)$, where $\mathbf{y}_i$, $(i = 1, \ldots, N)$, are samples of the random variable Y. Each input vector $\mathbf{x}_i$, $(i = 1, \ldots, N)$, belongs to one of the classes $c_1, c_2, \ldots, c_M$, and the prototypes $\mathbf{w}_j$, $(j = 1, \ldots, P)$, are determined by the LVQ algorithm. We consider the class labels as samples of a discrete random variable C. The vector λ stores the relevances.

We iteratively compute the relevances by gradient ascent on $o(Y,C)$:

$$\lambda^{(t+1)} = \lambda^{(t)} + \eta \sum_{i=1}^{N} \frac{\partial o(Y,C)}{\partial \mathbf{y}_i} \mathbf{I} \left(\mathbf{x}_i - \mathbf{w}_j \right), \tag{1}$$

where η is the learning rate.

For the continuous informational energy [6], we have:

$$o(Y,C) = \sum_{p=1}^{M} \frac{1}{p(c_p)} \int_{\mathbf{y}} p^2(\mathbf{y}, c_p) d\mathbf{y} - \int_{\mathbf{y}} p^2(\mathbf{y}) d\mathbf{y}. \tag{2}$$

Using the Parzen windows approximation [12], this can be rewritten:

ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 471-476

$$o(Y,C) = \frac{1}{N}\left(\sum_{p=1}^{M}\frac{1}{M_p}\right)\sum_{k,l=1}^{M_p} G(\mathbf{y}_{pk}-\mathbf{y}_{pl},2\sigma^2\mathbf{I}) - \frac{1}{N^2}\sum_{k,l=1}^{N} G(\mathbf{y}_k-\mathbf{y}_l,2\sigma^2\mathbf{I}) \tag{3}$$

where $\mathbf{I}$ is the unit diagonal matrix, $G(\mathbf{y},\sigma)$ is the Gaussian kernel, and M_p is the number of training samples from class c_p.

Using two consecutive samples $\mathbf{y}_1$ and $\mathbf{y}_2$, as suggested in [13], and obtain:

$$o(Y,C) = G(0,2\sigma^2\mathbf{I}) - \frac{1}{2}G(\mathbf{y}_1-\mathbf{y}_2,2\sigma^2\mathbf{I}). \tag{4}$$

We compute the partial derivatives of $o(Y,C)$ in (4), introduce them in (1), and finally obtain:

$$\lambda^{(t+1)} = \lambda^{(t)} - \eta\frac{1}{4\sigma^2}G(\mathbf{y}_1-\mathbf{y}_2,2\sigma^2\mathbf{I})(\mathbf{y}_2-\mathbf{y}_1)\mathbf{I}(\mathbf{x}_1-\mathbf{w}_{j(1)}-\mathbf{x}_2+\mathbf{w}_{j(2)}) \tag{5}$$

where $\mathbf{w}_{j(1)}$ and $\mathbf{w}_{j(2)}$ are the winner codebooks for $\mathbf{x}_1$, respectively for $\mathbf{x}_2$.

Using the relevances computed in (5), we define the weighted distance between an input vector $\mathbf{x}_i$ and a codebook vector $\mathbf{w}_j$:

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

We are ready to introduce the ERLVQ algorithm. After initializing the relevance vector $\lambda_k = 1/n$, $(k = 1,\ldots,n)$, and the codebook vectors, the following procedure updates incrementally the codebook vectors, the relevances and the feature ranks, for a given input $\mathbf{x}_i$:

1. Find the closest codebook, $\mathbf{w}_j$, the winner, which provides the minimum value of $|\mathbf{x}_i-\mathbf{w}_j|_\lambda$, for $j = 1,2,\ldots,K$.

2. Update the winner codebook:

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

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

3. Update the relevances according to (5). Normalize:

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

   $(k = 1,\ldots,n)$, to ensure that $\sum_{k=1}^{n}\lambda_k = 1$ and all $\lambda_k \in [0,1]$. (Note that λ_k can become negative in Step 3.)

4. Update the overall rank of each feature as an average over the previous steps.

ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 471-476

Table 1: Comparative recognition rates obtained with LVQ, RLVQ, OWA-RLVQ, and ERLVQ.

Database	LVQ	RLVQ	OWA-RLVQ	ERLVQ
Iris	91.33%	95.33%	96.66%	97.33%
Vowel	44.80%	46.32%	46.75%	47.18%
Ionosphere	90.06%	92.71%	93.37%	94.03%

Table 2: Feature ranking for the Iris database.

Rank	1	2	3	4
RLVQ Feature	4	2	3	1
OWA-RLVQ Feature	4	3	2	1
ERLVQ Feature	1	2	3	4

4 Experiments

We have tested the ERLVQ algorithm on the following standard datasets [2]: Iris, Vowel Recognition, and Ionosphere. Table 1 shows the recognition rates obtained for LVQ, RLVQ, OWA-RLVQ, and ERLVQ, using each time the same set of initial codebooks. We have generally obtained better results with ERLVQ.

Feature ranking is hard to compare because of the differences in data interpretation. For the Iris database, Table 2 presents the ranking of the four features, resulted after the RLVQ, OWA-RLVQ and ERLVQ training, using the same initial set of codebook vectors. LVQ can not calculate feature ranks. The last feature is the most important one for algorithms which try to find well defined classes based on their centroid, such as RLVQ and OWA-RLVQ. For ERLVQ, feature ranking is focused on discriminating as much as possible the borders of classes. The first feature is in this case the most important one.

The ERLVQ ranking is similar to the results reported in [5], where the bidimensional projection of patterns from classes "1" and "2", using only the first two features, leads to two well delimited clusters with relatively close centroids. On the other hand, the bidimensional projection of the same patterns based on the last two features creates two clusters with quite distanced centroids, but the classes are not well delimited.

5 Conclusions

ERLVQ is a computational attractive neural model for applications with large and redundant data sets. Compared to RLVQ and OWA-RLVQ, ERLVQ has

better recognition rates. For feature ranking, ERLVQ has a better class discrimination capability by comparison with RLVQ and OWA-RLVQ, but the resulted classes are not necessarily well distributed in the pattern space.

References

[1] R. Andonie and F. Petrescu. Interacting systems and informational energy. *Foundation of Control Engineering*, 11:53–59, 1986.

[2] K. Blacke, E. Keogh, and C. J. Merz. UCI Repository of Machine Learning Databases, www.ics.uci.edu/˜mlearn/MLSummary.html, 1998.

[3] T. Bojer, B. Hammer, D. Schunk, and K. T. von Toschanowitz. Relevance determination in learning vector quantization. In M. Verleysen, editor, *Proceedings of the European Symposium on Artificial Neural Networks (ESANN 2001)*, pages 271–276, D-side publications, 2001.

[4] A. Cataron and R. Andonie. RLVQ determination using OWA operators. In M. Hamza, editor, *Proceedings of the Third IASTED International Conference on Artificial Intelligence and Applications (AIA 2003), Benalmadena, Spain, Sept. 8-10*, pages 434–438, ACTA Press, 2003.

[5] R. K. De, N. R. Pal, and S. K. Pal. Feature analysis: Neural network and fuzzy set theoretic approaches. *Pattern Recognition*, 30:1579–1590, 1997.

[6] S. Guiasu. *Information Theory with Applications*. McGraw-Hill, 1977.

[7] B. Hammer and T. Villmann. Generalized relevance learning vector quantization. *Neural Networks*, 15:1059–1068, 2002.

[8] D. Huang and T. W. S. Chow. Searching optimal feature subset using mutual information. In M. Verleysen, editor, *Proceedings of the European Symposium on Artificial Neural Networks (ESANN 2003)*, pages 161–166, D-side publications, 2003.

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

[10] O. Onicescu. Theorie de l'information. Energie informationelle. *C. R. Acad. Sci., Ser. A-B*, Tome 263:841–842, 1966.

[11] M. Pregenzer, G. Pfurtscheller, and D. Flotzinger. Automated feature selection with a distinction sensitive learning vector quantizer. *Neurocomputing*, 11:19–29, 1996.

[12] J. C. Principe, D. Xu, and J. W. Fisher III. Information-theoretic learning. In S. Haykin, editor, *Unsupervised Adaptive Filtering*. Wiley, New York, NY, 2000.

[13] K. Torkkola. Feature extraction by non-parametric mutual information maximization. *Journal of Machine Learning Research*, 3:1415–1438, 2003.