Relevance Determination for Learning Vector Quantization using the Fisher Criterion Score *

Barry Ridge, Aleš Leonardis, and Danijel Skočaj.
Faculty of Computer and Information Science,
University of Ljubljana, Slovenia
{barry.ridge, danijel.skocaj, ales.leonardis}@fri.uni-lj.si

Abstract. *Two new feature relevance determination algorithms are proposed for learning vector quantization. The algorithms exploit the positioning of the prototype vectors in the input feature space to estimate Fisher criterion scores for the input dimensions during training. These scores are used to form online estimates of weighting factors for an adaptive metric that accounts for dimensional relevance with respect to classifier output. The methods offer theoretical advantages over previously proposed LVQ relevance determination techniques based on gradient descent, as well as performance advantages as demonstrated in experiments on various datasets including a visual dataset from a cognitive robotics object affordance learning experiment.*

1. Introduction

Learning vector quantization (LVQ) [9] provides an intuitive, and often highly effective, means for discriminative learning where prototype vectors are used to quantize the input feature space and given labels to form piecewise-linear classifiers using the nearest neighbour rule. Since their introduction, LVQ algorithms have undergone various analyses and seen various improvements to their design. The original formulations *(LVQ1, LVQ2, LVQ3)* [9] have been shown to be divergent, inspiring the *generalized learning vector quantization (GLVQ)* algorithm [14] where prototypes are updated such that a stochastic gradient descent is performed over an error function. LVQ algorithms have also been shown to be a family of maximum margin classifiers [3], thus providing excellent generalization for novel data with high-dimensional inputs. More recently, the nearest neighbour rule of LVQ has been modified to a k-nearest neighbours rule using a local subspace classifier [7].

Perhaps just as significantly, much attention has also been paid in recent years to the role that the distance metric plays in the effectiveness of LVQ methods. LVQ ordinarily relies on the Euclidean metric to measure the distance between data points, which provides equal weighting to all input dimensions. Many of the input dimensions, however, may have little relevance when considering the desired output function and may even have a detrimental effect on the output if considered with equal weighting in the metric to the more important dimensions. One standard approach to this issue is to pre-process the data using some form of feature selection or dimensionality reduction, but this can be infeasible in many learning scenarios where the training data are not available in advance, e.g. autonomous robotics. Various reformulations of LVQ have been proposed that can adjust the metric during training such that the impact of the individual input dimensions are dynamically re-weighted during training in accordance with the data under consideration. This can make a crucial difference, both during training for more efficient adjustment of the prototypes, and when classifying test samples where the undue consideration of irrelevant dimensions can mean the difference between a correct and incorrect classification.

One early adaptation of LVQ3 known as *distinction sensitive learning vector quantization (DSLVQ)* [11] achieves this by using a heuristic to adjust weights along each of the input dimensions to modify the Euclidean metric. An adaptation of LVQ1 known as *relevance learning vector quantization (RLVQ)* [1] uses Hebbian learning to do similar, by adjusting weights for each of the input dimensions at every

*This research has been supported by: EU FP7 project CogX (ICT-215181), and Research program P2-0214 Computer Vision (Republic of Slovenia).

training step depending on whether they contributed to the correct or incorrect classification of a training sample. RLVQ was subsequently adapted for use with GLVQ producing a method known as *generalized relevance learning vector quantization (GRLVQ)* [6] such that the dimensional weight updates also adhere to gradient descent dynamics in a similar way to the prototype updates. Another modified version of GLVQ [15] uses Fisher's discriminant analysis to create an alternative metric to the weighted Euclidean distance that employs a matrix transformation to reduce the feature space dimensionality. More recently, an adaptive metric was used in combination with training data selection for LVQ [10].

In this paper, two new algorithms for LVQ-based relevance determination are presented. Both methods exploit the positioning of the prototype vectors in the input feature space to inform estimates of the Fisher criterion score along the input dimensions, which are then used to form online estimates of the relevance of the input dimensions with respect to the classifier output. Both methods provide online updates that may be used alongside regular LVQ updates and neither method requires the specification of a learning rate, as in stochastic gradient descent. The remainder of the paper is organized as follows. In Section 2 the background theory and related algorithms are outlined. The new algorithms are described in Section 3. Experimental results are provided in Section 4 and concluding remarks are provided in Section 5.

2. Related Algorithms

Let $X = \{(\mathbf{x}^i, y^i) \subset \mathbb{R}^n \times \{1, \ldots, C\} \,|\, i = 1, \ldots, N\}$ be a training set of n-dimensional vectors and corresponding class labels. Let $X^c = \left\{(\mathbf{x}^i, y^i) \in X \,\middle|\, y^i = c\right\}$ and $N^c = |X^c|$. Similarly, let $\mathcal{W} = \{(\mathbf{w}^i, c^i) \subset \mathbb{R}^n \times \{1, \ldots, C\} \,|\, i = 1, \ldots, M\}$ be a set of prototype vectors with corresponding class labels, and let $\mathcal{W}^c = \left\{(\mathbf{w}^i, c^i) \in \mathcal{W} \,\middle|\, c^i = c\right\}$ and $M^c = |\mathcal{W}^c|$. Given a vector $\mathbf{x} \in \mathbb{R}^n$, denote its components as $(x_1, \ldots, x_n)$. Letting $\mathbf{x}$ be an n-dimensional data vector and $\mathbf{w}$ be an n-dimensional prototype vector, then a weighted squared Euclidean distance between both vectors may be defined as

$$d^2(\mathbf{x}, \mathbf{w}) = \sum_{i=1}^{n} \lambda_i (x_i - w_i)^2, \tag{1}$$

where the λ_i are weighting factors for each dimension. Adding such weights to the Euclidean metric allows for the possibility of re-scaling each of the input dimensions depending on their respective influences on the classification output. Moreover, it enables the metric to be made *adaptive* such that the weights are adjusted dynamically during training depending on the data.

Prototype vectors have associated receptive fields based on the metric and classification of samples is performed by determining which receptive fields those samples lie in, or alternatively, which prototype vectors are closest to the samples. The receptive field of prototype $\mathbf{w}^i$ is defined as: $R^i = \{\mathbf{x} \in X \,|\, \forall(\mathbf{w}^j, c^j) \in \mathcal{W},\, d^2(\mathbf{x}, \mathbf{w}^i) \leq d^2(\mathbf{x}, \mathbf{w}^j)\}$. Given a sample $(\mathbf{x}, y) \in X$, we denote by $g(\mathbf{x})$ a function that is negative if $\mathbf{x}$ is classified correctly, i.e. $\mathbf{x} \in R^i$ with $c^i = y$, and is positive if $\mathbf{x}$ is classified incorrectly, i.e. $\mathbf{x} \in R^i$ with $c^i \neq y$. We also let f be some monotonically increasing function.

The goal of GLVQ [14] is to minimize

$$E = \sum_{i=1}^{m} f(g(\mathbf{x}^i)) \tag{2}$$

via stochastic gradient descent. The update rules for GLVQ and many other LVQ algorithms can be derived using the above notation. In the following, the LVQ1 [9], RLVQ [1], GLVQ [14] and GRLVQ [6] algorithms will be reviewed, before introducing the proposed relevance determination methods.

2.1. LVQ1

Given a training sample $(\mathbf{x}, y) \in X$, by letting $f(x) = x$ and $g(\mathbf{x}) = \eta d_j$ where $d_j = d^2(\mathbf{x}, \mathbf{w}^j)$ with $\mathbf{w}^j$ being the closest prototype to $\mathbf{x}$ and $\{\lambda_i = 1\}_{i=1}^m$ (i.e. equal weights for regular Euclidean distance), with $\eta = 1$ if $\mathbf{x}$ is classified correctly (i.e. $c^j = y$) and $\eta = -1$ if $\mathbf{x}$ is classified incorrectly (i.e. $c^j \neq y$), the following stochastic gradient descent update rule may be derived for LVQ1 [9]:

$$\mathbf{w}_{t+1}^j = \begin{cases} \mathbf{w}_t^j + \alpha(\mathbf{x} - \mathbf{w}_t^j), & \text{if } c^j = y \\ \mathbf{w}_t^j - \alpha(\mathbf{x} - \mathbf{w}_t^j), & \textit{otherwise}, \end{cases} \tag{3}$$

where α is the learning rate and the t subscripts denote prototype states at different training steps. However, it should be noted that the error function as defined here is highly discontinuous, and thus can lead to instabilities in the algorithm. GLVQ, discussed next, was designed to resolve this issue.

2.2. GLVQ

Here, $d_j = d^2(\mathbf{x}, \mathbf{w}^j)$ is defined where $\mathbf{w}^j$ is the closest prototype to $\mathbf{x}$ with label $c^j = y$ and $d_k = d^2(\mathbf{x}, \mathbf{w}^k)$ where $\mathbf{w}^k$ is the closest prototype to $\mathbf{x}$ with some other label. By letting

$$g(\mathbf{x}) = \frac{d_j - d_k}{d_j + d_k} \tag{4}$$

and

$$f_t(g(\mathbf{x})) = \frac{1}{1 + \exp^{-g(\mathbf{x})t}}, \tag{5}$$

which is a sigmoidal function that redefines the error function (Eq. 2) such that it is continuous over borders between the receptive fields for $\mathbf{w}^j$ and $\mathbf{w}^k$. When minimized, the error function yields the following update rules for $\mathbf{w}^j$ and $\mathbf{w}^k$ [14]:

$$\mathbf{w}^j_{t+1} := \mathbf{w}^j_t + \alpha\nu \frac{d_k}{(d_j + d_k)^2}(\mathbf{x} - \mathbf{w}^j_t) \tag{6}$$

$$\mathbf{w}^k_{t+1} := \mathbf{w}^k_t + \alpha\nu \frac{d_j}{(d_j + d_k)^2}(\mathbf{x} - \mathbf{w}^k_t) \tag{7}$$

where

$$\nu = f'_t(g(\mathbf{x})) = f_t(g(\mathbf{x}))(1 - f_t(g(\mathbf{x})). \tag{8}$$

GLVQ, unlike LVQ1 or the rest of Kohonen's original LVQ formulations, has been shown to be convergent [14, 6], although it is sensitive to the initialization of the prototype vectors. This is demonstrated in the experimental results of Section 4.

2.3. RLVQ and GRLVQ

The LVQ prototype update equations can be accompanied by updates that also alter the λ_i in Eq. (1) dynamically during training, hence allowing for an adaptive Euclidean metric. In RLVQ [1], LVQ1 training is adjusted such that the following weighting factor update rule is applied alongside Eq. (3):

$$\lambda_l := \begin{cases} \lambda_l - \beta(x_l - w^j_l)^2 & \text{if } c^j = y \\ \lambda_l + \beta(x_l - w^j_l)^2 & \textit{otherwise}, \end{cases} \tag{9}$$

for each l-th dimension where $\beta \in (0, 1)$ is a learning rate for the weighting factor adjustments. The weights are normalized at each update such that $\sum_{i=1}^{n} \lambda_i = 1$. The motivation for the above comes from Hebbian learning, the idea being that when $\mathbf{w}^j$ classifies the sample $\mathbf{x}$ correctly, the weights for the dimensions that contributed to the classification the most are increased, whereas the weights of those that contributed the least are decreased. When $\mathbf{w}^j$ incorrectly classifies $\mathbf{x}$, the weights for dimensions that contributed most are decreased, whereas the weights for dimensions that contributed the least are increased. GRLVQ [6] is an application of the above idea to GLVQ, such that the updates for the weights for the metric also follow a stochastic gradient descent on the error function defined by GLVQ.

One disadvantage of both RLVQ and GRLVQ is that they require the specification of an additional learning rate, β, which can be difficult to specify appropriately with respect to its α counterpart in the prototype updates. Another disadvantage is that they fail to take into consideration the additional statistical information provided by the remaining prototypes other than the ones currently being updated at a given training step when making relevance estimates. These issues are addressed with the following two proposed LVQ relevance determination algorithms.

3. Proposed Algorithms

The Fisher criterion, while ordinarily associated with Fisher's discriminant analysis [4], can also serve as an effective means for relevance determination when applied across individual data dimensions. Letting $\overline{x}^A = \frac{1}{N}\sum_{x^i \in A} x^i$ be the mean of a set of points A with cardinality N, the Fisher criterion score for a given individual dimension l is defined as

$$F(l) = \frac{S_B(l)}{S_W(l)}, \tag{10}$$

where

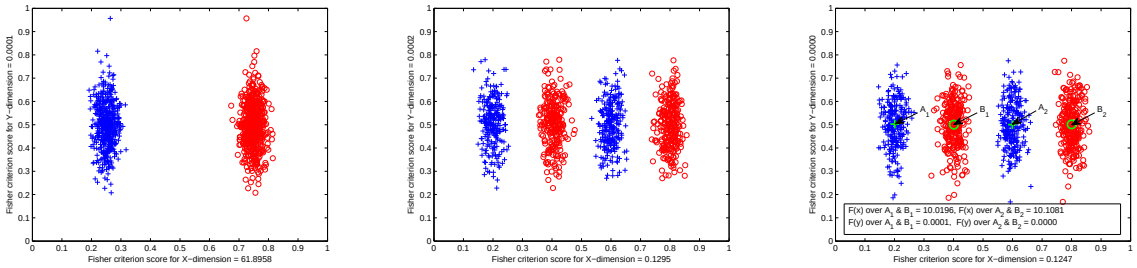
$$S_B(l) = \sum_{c=1}^{C} N^c \left(\overline{x}^{X^c}_l - \overline{x}^X_l\right)^2 \tag{11}$$

is the between-class variance and

$$S_W(l) = \sum_{c=1}^{C} \sum_{\mathbf{x} \in X^c} \left(x_l - \overline{x}^{X^c}_l\right)^2 \tag{12}$$

is the within-class variance over the l-th dimension.

With regard to relevance determination for LVQ, $F(l)$ could be calculated for each dimension over the entire training set X in advance of LVQ training and applied to the weighting factors in Eq. (1) by setting $\lambda_l = F(l)$ for all l to form a weighted metric. However, for many applications it is more desirable to have an online feature relevance training mechanism that is not reliant on having access to the entire training set at once. Two such online algorithms where estimation of the Fisher criterion score is integrated into the training scheme are presented next.



(a) (b) (c)

Figure 1. A simple 2D, 2-class example of how the Fisher criterion score (see Eq. (10)) can fail as a feature relevance metric over multi-modal distributions. (a) shows uni-modal class data distributions, linearly separable in the x-dimension, but with large overlap in the y-dimension. The score reflects the relevance of each dimension to class discrimination. (b) by comparison, shows the same number of data points, but with a multi-modal distribution (yet still linearly separable in x). The score is significantly lower for the x-dimension in this case. (c) shows the improvement provided by calculating the score between pairs of clusters with centers at points A_1, B_1, A_2 and B_2. See Section 3 for more details.

3.1. Algorithm 1

With the first algorithm, rather than calculating $F(l)$ over the data in X, at a given timestep t the score is estimated over the values of the prototype vectors in W. This is plausible since the distribution of the prototype vectors should approximate the distribution of the data over time. During training, certain prototypes will quantize more significant modes of the distribution than others, thus to account for this, weighted means and variances are calculated for each class based on the classification accuracy of each of the prototypes of that class, then the Fisher criterion score is calculated over the weighted means and variances for all classes. Firstly, the definition of $\mathcal{W}$ is altered to $\mathcal{W} = \{(\mathbf{w}^i, c^i, p^i) \subset \mathbb{R}^n \times \{1, \ldots, C\} \times \mathbb{R} \mid i = 1, \ldots, M\}$ where, given random variable $(\mathbf{x}, y)$, $p^i = p(\mathbf{x} \in R^i | y = c^i)$ is the conditional probability of x lying in receptive field R^i of prototype $\mathbf{w}^i$ given that $\mathbf{w}^i$ correctly classifies x. The p^i form probability distributions over class prototypes such that $\sum_{p^i \in \mathcal{W}^c} p^i = 1$ for each class c. A definition of the estimated Fisher criterion score may now be formed as

$$F(l) \simeq \hat{F}(l) = \frac{\hat{S}_B(l)}{\hat{S}_W(l)}, \tag{13}$$

where

$$S_B(l) \simeq \hat{S}_B(l) = \sum_{c=1}^{C} \frac{N^c}{N} \left(\hat{w}_l^{\mathcal{W}^c} - \hat{w}_l^{\mathcal{W}}\right)^2 \tag{14}$$

is the estimated between-class variance over the l-th dimension,

$$S_W(l) \simeq \tag{15}$$

$$\hat{S}_W(l) = \sum_{c=1}^{C} \frac{N^c}{N} \sum_{(\mathbf{w}^i, c^i, p^i) \in \mathcal{W}^c} p^i \left(w_l - \hat{w}_l^{\mathcal{W}^c}\right)^2 \tag{16}$$

is the estimated within-class variance over the l-th dimension, and

$$\hat{w}_l^{\mathcal{W}^c} = \sum_{(\mathbf{w}^i, c^i, p^i) \in \mathcal{W}^c} p^i w_l^i \tag{17}$$

is a weighted mean over the l-th dimension of prototypes in a given set $\mathcal{W}^c \subseteq \mathcal{W}$.

The λ_m relevance factors may then be updated at each timestep by taking a running mean of the normalized estimated Fisher criterion score:

$$\lambda_{l,t+1} := \lambda_{l,t} + \frac{\frac{\hat{F}(l)}{\sum_{l=1}^{n} \hat{F}(l)} - \lambda_{l,t}}{t+1}. \tag{18}$$

While the Fisher criterion score is suitable for feature relevance determination in many cases, its main drawback is that it does not cope well with multi-modal feature distributions. An example of this is shown in Figure 1. This problem remains in the estimation proposed above, since Eq. (14) and Eq. (16) are calculated over all class prototypes. The second proposed algorithm was designed to account for this.

3.2. Algorithm 2

The second proposed algorithm is based on the idea of calculating the Fisher criterion score between

single prototype vectors of opposing classes, where the assumption is made that each class prototype vector may be quantizing different modes of the underlying class distribution. During training, Gaussian kernels are used to maintain estimates of the accuracies of each of the prototypes over the parts the data distribution accounted for by each of their receptive fields. At a given training step, the nearest single prototypes of each class to the training sample are found, and their Gaussian kernels are used to calculate an estimate of the Fisher criterion score for that local portion of the distribution, which is subsequently averaged over the entire training period.

The definition of $\mathcal{W}$ is this time altered to accommodate a Gaussian estimate of the accurate portion of the receptive field for each prototype, such that

$$\mathcal{W} = \{ \left(\mathbf{w}^i, c^i, \mathcal{N}(\mathbf{x}; \mu^i, \Sigma^i)\right) \subset \mathbb{R}^n \times \{1, \ldots, C\} \times (\mathbb{R}^n \times \mathbb{R}^{n \times n}) \mid i = 1, \ldots, M \}, \quad (19)$$

where $\mathcal{N}$ approximates $\tilde{R}^i = \{\mathbf{x} \in R^i | y = c\}$ with mean μ^i and covariance matrix $\Sigma^i = \text{diag}([s_1^i, \ldots, s_n^i])$ where the $\{s_l^i\}_{l=1}^n$ are variances along each l-th dimension. During LVQ training, given a random sample $(\mathbf{x}, y) \in X$ at training step t, if the closest prototype $\mathbf{w}^j$ classifies x correctly, i.e. $c^j = y$, then μ_l^j and s_l^j are updated in each l-th dimension as follows [8]:

$$\mu_{l,t}^j := \mu_{l,t-1}^j + \frac{x_l - \mu_{l,t-1}^j}{t} \quad (20)$$

$$\hat{s}_{l,t}^j := \hat{s}_{l,t-1}^j + (x_l - \mu_{l,t-1}^j)(x_l - \mu_{l,t}^j) \quad (21)$$

where $\mu_{l,t}^j$ is the running mean estimate and $s_{l,t}^j = \frac{\hat{s}_{l,t}^j}{t-1}$ is the running variance estimate for the l-th dimension at training step t. If $c^j \neq y$, then the above updates are not performed. Assuming a sufficient number of updates have been performed on the relevant prototypes up until step t, a Fisher criterion score estimate may be calculated between

$$\mathcal{W}' = \{\omega^k = \left(\mathbf{w}^k, c^k, \mathcal{N}(\mathbf{x}; \mu^k, \Sigma^k)\right) \in \mathcal{W} \mid \forall \mathbf{w}^i, c^i = c^k, d(\mathbf{x}, \mathbf{w}^k) \leq d(\mathbf{x}, \mathbf{w}^i) \}, \quad (22)$$

the closest prototypes of different classes (including $\mathbf{w}^j$), as follows:

$$F(l) \simeq \tilde{F}(l) = \frac{\tilde{S}_B(l)}{\tilde{S}_W(l)}, \quad (23)$$

where

$$S_B(l) \simeq \tilde{S}_B(l) = \frac{1}{C} \sum_{c=1}^{C} (\mu_l^c - \overline{\mu}_l)^2 \quad (24)$$

is the between-class variance estimate in the l-th dimension with

$$\overline{\mu}_l = \frac{1}{C} \sum_{\omega^k \in \mathcal{W}'} \mu_l^k, \quad (25)$$

and

$$S_W(l) \simeq \tilde{S}_W(l) = \sum_{\omega^k \in \mathcal{W}'} s_l^k \quad (26)$$

is the within-class variance estimate in the l-th dimension. The relevance factors may then be updated in a similar way to Eq. (18), this time using the new estimates:

$$\lambda_{l,t+1} := \lambda_{l,t} + \frac{\frac{\tilde{F}(l)}{\sum_{l=1}^n \tilde{F}(l)} - \lambda_{l,t}}{t+1}. \quad (27)$$

Since each prototype carries an accompanying Gaussian kernel that estimates its accuracy, it is now possible to estimate the Fisher criterion score using only single prototypes from each class, as opposed to the previous algorithm where multiple prototypes in each class have to be considered to achieve variance estimates. Though the model is made more complex, it is more capable of successfully handling the multi-modal distribution issue described in Fig. 1 as shown by the experimental results in the next section.

4. Experiments

The proposed algorithms were evaluated over simulated data, datasets from the UCI repository, and a real-world dataset from a cognitive robotics object affordance learning experiment. In the following, the datasets are described in more detail and experimental results are provided in Section 4.1. Two simulated datasets were proposed in [1, 6], the first of which was replicated for the experiments here. The data is composed of three classes, each separated into two clusters with some small overlap to form multi-modal class data distributions in the first two dimensions. Eight further dimensions are generated from the first two dimensions as follows: assuming (x_1, x_2) is one data point, $x_3 = x_1 + \eta_1, \ldots, x_6 = x_1 + \eta_4$ is chosen where η_i comprises normally-distributed noise with variances $0.05, 0.1, 0.2$, and 0.5 respectively. The remaining $x_7, \ldots, x_{10}$ components contain pure noise uniformly distributed in

$[-0.5, 0.5]$ and $[-0.2, 0.2]$. This dataset is multimodal for each class in the two relevant dimensions and thus provides a good test for the potential difference between the two proposed algorithms.

Dataset	# Features	# Samples	# Classes
Simulated	10	90	3
Iris	4	150	3
Ionosphere	34	351	2
Wine	13	178	3
Soybean	35	47	4
WBC	30	569	2
Affordance	11	160	2

Table 1. An attribute list for the datasets in Section 4.

Five different datasets from the UCI repository [5] were tested: Fisher's Iris dataset, the ionosphere dataset, the wine dataset, the soybean dataset (small), and the Wisconsin breast cancer (WBC) dataset. A dataset from a cognitive robotics object affordance learning experiment [13] was also tested. It consists of eight household objects separated into two classes, four rolling objects and four non-rolling objects, and labeled as such, accompanied by eleven different shape features, two of which measure the curvature of 3D points from stereo images of the objects and the remainder of which were derived from 2D silhouettes of the objects.

4.1. Results

The primary goal of the investigation was to evaluate whether or not the new algorithms when applied to standard LVQ methods such as LVQ1 and GLVQ offer performance improvements over those methods in their original form, as well as over other relevance determination techniques for LVQ, such as RLVQ and GRLVQ. The results of these comparisons are outlined in Table 2 and are discussed in more detail in the following. In the results, the proposed Fisher criterion score-based relevance determination algorithms are referred to as FC1LVQ1 and FC2LVQ1 respectively when applied to LVQ1, and FC1GLVQ and FC2GLVQ when applied to GLVQ.

A secondary consideration was to test the methods under the duress of various different conditions. GLVQ, for example is known to perform poorly if the prototype vectors are not initialized within the data distribution [12], thus in our evaluations, both random prototype initializations as well as initializations where the prototypes are placed at the mean points of class clusters were considered. Note that random prototype initialization in this case refers to selecting random values for each prototype dimension scaled within the data range. K-means clustering was used to determine class clusters in the latter case.

The performance of LVQ algorithms over short training periods is not often considered in the literature, which tends to favour evaluations of the algorithms over several hundred training epochs until convergence is reached. Given that LVQ algorithms have online training mechanisms, and that the relevance determination techniques proposed above were explicitly developed to also function online, sample-by-sample without access to the rest of the training set, such short-term training evaluations are important if the methods are to be considered useful in real-world online settings, e.g. cognitive robotics [13], where the entire training set is often unavailable at any given point during training.

Thus, the results in Table 2 are divided into four main evaluations: both 1 epoch and 300 epochs of training from random initialization, and both 1 epoch and 300 epochs of training from class cluster mean initialization. The 300 epoch sessions used the relatively slow learning rates of $\alpha = 0.1$ for the prototype updates (cf. Eq. (3), Eq. (6) & Eq. (7)) and $\beta = 0.01$ for the dimensional relevance updates where required (cf. Eq. (9)), whereas the 1 epoch training sessions used the faster rates of $\alpha = 0.3$ and $\beta = 0.1$. Note that the FC1 and FC2 methods do not require the additional β learning rate. In each of the 1 epoch evaluations, 20 trials of ten-fold cross validation were performed with random data orderings in each trial, and results were averaged over test data performance, whereas in the 300 epoch evaluations, 5 trials were performed. 10 prototypes were used for every dataset and the data dimensions were scaled prior to training.

The results in Table 2 show that when trained over a single epoch from random initialization, of the algorithms tested FC2LVQ1 and FC2GLVQ achieved higher mean classification scores than their counterparts in many cases. Over long-term training of 300 epochs from random initialization, the results for all algorithms aside from GLVQ, tend to improve with FC2LVQ1 and FC2GLVQ again tending to be competitive with their counterparts. It is worth noting here the impact relevance determination has on improving the results of GLVQ when exposed to poor prototype initialization. When the prototypes are initialized optimally at the class cluster mean points the results tend to improve dramatically across all of the

Dataset	LVQ1	RLVQ1	**FC1LVQ1**	**FC2LVQ1**	GLVQ	GRLVQ	**FC1GLVQ**	**FC2GLVQ**
Random Initialization, 1 Epoch of Training, 20 Trials								
Sim	53± 18%	64± 22%	54± 19%	**69± 18%**	37± 17%	63± 22%	51± 20%	**70± 19%**
Iris	90± 8%	91± 9%	93± 9%	**95± 5%**	63± 24%	**89± 13%**	83± 19%	88± 15%
Iono	81± 8%	75± 11%	**85± 6%**	84± 7%	66± 13%	80± 9%	82± 7%	**84± 7%**
Wine	93± 6%	79± 13%	92± 9%	**94± 6%**	52± 19%	92± 8%	85± 14%	**94± 7%**
Soy	**89± 17%**	83± 24%	89± 18%	85± 21%	34± 27%	84± 22%	83± 21%	**85± 20%**
WBC	92± 4%	86± 8%	93± 4%	**93± 3%**	71± 19%	93± 5%	90± 10%	**94± 3%**
Afford	97± 7%	93± 10%	98± 4%	**99± 3%**	78± 22%	96± 9%	84± 20%	**98± 6%**
Random Initialization, 300 Epochs of Training, 5 Trials								
Sim	79± 14%	79± 13%	77± 16%	**87± 12%**	38± 17%	**96± 7%**	90± 12%	94± 9%
Iris	92± 7%	92± 8%	95± 5%	**96± 5%**	47± 24%	96± 5%	91± 16%	**96± 4%**
Iono	85± 7%	80± 10%	**86± 8%**	85± 7%	60± 16%	**90± 5%**	90± 6%	89± 6%
Wine	95± 5%	77± 11%	95± 5%	**96± 5%**	42± 18%	96± 5%	97± 4%	**98± 3%**
Soy	99± 6%	97± 10%	**100± 4%**	98± 7%	33± 26%	97± 8%	**97± 7%**	96± 9%
WBC	93± 3%	87± 7%	**94± 3%**	**94± 3%**	62± 20%	96± 3%	96± 3%	**96± 2%**
Afford	**99± 2%**	95± 7%	**99± 2%**	99± 3%	67± 24%	**99± 2%**	**99± 2%**	**99± 2%**
Class Cluster Mean Initialization, 1 Epoch of Training, 20 Trials								
Sim	82± 12%	**98± 5%**	78± 17%	93± 8%	90± 9%	91± 9%	85± 13%	**93± 8%**
Iris	**96± 5%**	**96± 5%**	**96± 5%**	**96± 5%**	95± 5%	95± 5%	95± 5%	**96± 5%**
Iono	87± 6%	80± 10%	**88± 6%**	**88± 6%**	**90± 5%**	88± 6%	89± 5%	**90± 5%**
Wine	95± 5%	86± 11%	**96± 5%**	**96± 5%**	**97± 4%**	97± 5%	97± 5%	97± 5%
Soy	**100± 2%**	95± 10%	100± 3%	99± 5%	**100± 2%**	99± 4%	**100± 2%**	99± 5%
WBC	**95± 3%**	88± 7%	94± 3%	94± 3%	96± 3%	96± 3%	**97± 3%**	95± 3%
Afford	**99± 2%**	98± 4%	**99± 2%**	**99± 2%**	**99± 2%**	**99± 2%**	**99± 2%**	**99± 2%**
Class Cluster Mean Initialization, 300 Epochs of Training, 5 Trials								
Sim	84± 11%	86± 16%	87± 12%	**91± 10%**	90± 9%	**97± 6%**	90± 10%	96± 8%
Iris	96± 5%	95± 6%	**96± 4%**	96± 5%	96± 6%	95± 5%	**97± 4%**	96± 4%
Iono	88± 5%	82± 9%	**89± 5%**	88± 5%	89± 5%	90± 5%	90± 5%	**91± 5%**
Wine	96± 5%	82± 12%	**97± 4%**	96± 5%	97± 4%	**98± 3%**	**98± 3%**	**98± 3%**
Soy	**100± 0%**	94± 11%	99± 6%	98± 7%	**100± 0%**	98± 8%	99± 5%	99± 6%
WBC	**96± 2%**	89± 5%	95± 3%	95± 3%	96± 3%	96± 3%	**97± 2%**	**97± 2%**
Afford	**99± 2%**	98± 3%	**99± 2%**	**99± 2%**	99± 3%	**99± 2%**	**99± 2%**	**99± 2%**

Table 2. 10-Fold cross validation, 10 prototypes. Highest scores for LVQ1 & GLVQ based algorithms are shown in bold.

classifiers in short-term training, with both FC1 and FC2 relevance determination doing well over both short-term and long-term training periods, with FC1 out-performing FC2 in some cases and vice versa. Over all the evaluations, FC1GLVQ and FC2GLVQ trained over 300 epochs with class cluster mean initialization tended to score well when compared with the other methods. It should also be noted that, when the class distribution in the data is multi-modal, as is the case with the simulated dataset, FC2-based methods tend to be a better choice than FC1-based methods, as predicted.

A third consideration was to compare the new methods to a state-of-the-art batch method such as the *support vector machine (SVM)*. Batch methods, as opposed to online methods that are trained sample-by-sample, have access to the entire training set during training, and therefore usually provide superior results. Table 3 shows the results of a comparison between FC1GLVQ, FC2GLVQ and a multi-class SVM trained with a radial basis function (RBF) kernel [2]. For this comparison, the results for FC1GLVQ and FC2GLVQ from the 300 epoch, class cluster mean-initialized evaluation described previously were used, while ten-fold cross validation over five trials was also used for the SVM, where the test data results were averaged over the five trials and SVM parameters were optimized using cross validation over the training data prior to training. The results show both FC1GLVQ and FC2GLVQ performing well when compared with SVM over the various datasets, particularly in the case of the simulated multi-modal dataset.

It is difficult to evaluate the performance of the algorithms with respect to the estimation of the λ_l weighting factors themselves, but examples of the

Dataset	FC1GLVQ	FC2GLVQ	SVM
Simulated	90±10%	**96±8%**	78±14%
Iris	**97±4%**	96±4%	96±6%
Ionosphere	90±5%	91±5%	**94±4%**
Wine	**98±3%**	**98±3%**	**98±3%**
Soybean	99±5%	99±6%	**100±0%**
WBC	97±2%	97±2%	**98±2%**
Affordance	**99±2%**	**99±2%**	99±3%

Table 3. FC1GLVQ & FC2GLVQ versus SVM. Highest mean scores are shown in bold.

mean values for certain datasets are provided here. For the simulated dataset, $\lambda_{\text{FC1GLVQ}} = \{0.10, 0.42, 0.07, 0.06, 0.10, 0.06, 0.04, 0.03, 0.07, 0.04\}$ and $\lambda_{\text{FC2GLVQ}} = \{0.40, 0.43, 0.06, 0.01, 0.01, 0, 0, 0, 0, 0\}$, thus demonstrating that FC2GLVQ does indeed do a better job of handling the multi-modal distribution. For the Iris dataset, $\lambda_{\text{FC1GLVQ}} = \{0.02, 0.02, 0.55, 0.40\}$ and $\lambda_{\text{FC2GLVQ}} = \{0.03, 0.07, 0.37, 0.53\}$. For the object affordance dataset, $\lambda_{\text{FC1GLVQ}} = \{0.04, 0.56, 0.05, 0.05, 0.03, 0.05, 0.04, 0.04, 0.01, 0.09, 0.05\}$ and $\lambda_{\text{FC2GLVQ}} = \{0.05, 0.34, 0.07, 0.07, 0.07, 0.08, 0.01, 0.08, 0.06, 0.12, 0.06\}$, where one of the 3D curvature features is favoured in each case.

5. Conclusion

In conclusion, two new relevance determination algorithms have been proposed for LVQ that exploit the positioning of prototypes in the input feature space to calculate Fisher criterion score estimates in the input dimensions for an adaptive metric. An advantage provided by these methods over other metric-adaptive LVQ methods based on gradient descent, is that they do not require a learning rate or other parameters to be specified. Moreover, they provide incremental update rules that operate alongside regular LVQ update rules and can therefore be applied to any algorithms based on the general LVQ paradigm. Experimental evaluations were provided under various stress conditions and over various datasets and the proposed methods were shown to perform competitively against various other LVQ-based methods, and against SVM. With regard to future work, it would be interesting to apply the proposed techniques to prototype-based methods other than LVQ, such as supervised neural gases.

References

[1] T. Bojer, B. Hammer, D. Schunk, and K. T. von Toschanowitz. Relevance determination in learning vector quantization. In *European Symposium on Artificial Neural Networks*, pages 271–276, 2001. 1, 2, 3, 5

[2] C. Chang and C. Lin. LIBSVM: a library for support vector machines. 2001. 7

[3] K. Crammer, R. Gilad-Bachrach, A. Navot, and N. Tishby. Margin analysis of the LVQ algorithm. *Advances in Neural Information Processing Systems*, page 479–486, 2003. 1

[4] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936. 3

[5] A. Frank and A. Asuncion. *UCI Machine Learning Repository*. University of California, Irvine, School of Information and Computer Sciences, 2010. 6

[6] B. Hammer and T. Villmann. Generalized relevance learning vector quantization. *Neural Networks*, 15(8-9):1059–1068, 2002. 2, 3, 5

[7] S. Hotta. Learning vector quantization with local subspace classifier. In *Proceedings of the 19th International Conference on Pattern Recognition*, page 1–4, 2008. 1

[8] D. E. Knuth. *The Art of Computer Programming (Volume 2)*. Addison–Wesley, 1981. 5

[9] T. Kohonen. *Self-organizing maps*. Springer, 1997. 1, 2

[10] C. E. Pedreira. Learning vector quantization with training data selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1):157–162, 2006. 2

[11] M. Pregenzer, G. Pfurtscheller, and D. Flotzinger. Automated feature selection with a distinction sensitive learning vector quantizer. *Neurocomputing*, 11(1):19–29, 1996. 1

[12] A. Qin and P. Suganthan. Initialization insensitive LVQ algorithm based on cost-function adaptation. *Pattern Recognition*, 38(5):773–776, 2005. 6

[13] B. Ridge, D. Skočaj, and A. Leonardis. Self-Supervised Cross-Modal online learning of basic object affordances for developmental robotic systems. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Anchorage, AK, 2010. 6

[14] A. Sato and K. Yamada. Generalized learning vector quantization. In *Advances in Neural Information Processing Systems 8: Proceedings of the 1995 Conference*, page 423–429. MIT Press, 1996. 1, 2, 3

[15] M. K. Tsay, K. H. Shyu, and P. C. Chang. Feature transformation with generalized learning vector quantization for hand-written chinese character recognition. *IEICE Transactions on Information and Systems*, E82-D(3):687–692, 1999. 2