Generalized derivative based kernelized Learning Vector Quantization

Frank-Michael Schleif[1], Thomas Villmann[2],
Barbara Hammer[1], Petra Schneider[3], and Michael Biehl[3]

[1] Dept. of Techn., Univ. of Bielefeld,
Universitätsstrasse 21-23, 33615 Bielefeld, Germany,
schleif@informatik.uni-leipzig.de,hammer@in.tu-clausthal.de
[2] Faculty of Math./Natural and CS, Univ. of Appl. Sc. Mittweida,
Technikumplatz 17, 09648 Mittweida, Germany,
villmann@hsmw.de
[3] Johann Bernoulli Inst. for Math. and CS, Univ. of Groningen,
P.O. Box 407, 9700 AK Groningen, The Netherlands,
{m.biehl,p.schneider}@rug.nl

Abstract. We derive a novel derivative based version of kernelized Generalized Learning Vector Quantization (KGLVQ) as an effective, easy to interpret, prototype based and kernelized classifier. It is called D-KGLVQ and we provide generalization error bounds, experimental results on real world data, showing that D-KGLVQ is competitive with KGLVQ and the SVM on UCI data and additionally show that automatic parameter adaptation for the used kernels simplifies the learning.

1 INTRODUCTION

Kernelized learning vector quantization (KGLVQ) was proposed in [9] as an extended approach of Generalized Learning Vector Quantization (GLVQ) [10] with the goal to provide non-linear modeling capabilities for learning vector quantizers and to improve the performance in classification tasks. While the approach was quite promising it has been used only rarely due to its calculation complexity. One drawback is the storage of a large kernel matrix and additionally the storage and update of a combinatorial coefficient matrix Ψ. To address this problem multiple authors have proposed alternative strategies to deal with non-linear separable data focusing either on local linear models, the direct integration of alternative diagonal metrics in the cost functions or by full matrix learning strategies [6, 2]. Data analysis using kernel methods is a very active field of research (see e.g. [3]) and was very successful in analyzing non-linear problems. The underlying parameters for the kernel are thereby determined in general using cross validation approaches. The obtained models show good generalization behavior but are in general hard to interpret due to the non-linear mapping of the data in a kernel space and the fact that the model parameters are identified based on decision boundary points. Prototype based algorithms provide models which are calculated on typical points of the data space and are hence easily interpretable by experts of the field [11].

In this paper we propose an approach combining the positive aspects of both domains. We extend the GLVQ by a kernelized, differentiable metric called D-KGLVQ which allows the non-linear representation of the data, on the other hand we keep the prototype concept and obtain very compact representation models. Additionally the presented approach allows for an adaptation of kernel parameters during the learning procedure.

In Sec. 2 we present a short introduction into kernels and give the notations used throughout the paper. Subsequently we present the D-KGLVQ algorithm and evaluate it in comparison to standard GLVQ, the kernelized KGLVQ and a state of the art SVM. Finally, we conclude with a discussion in Section 4.

2 PRELIMINARIES

We consider a mapping of a data space X to a potentially high dimensional space $\mathcal{F} : \phi : X \longrightarrow \mathcal{F}$. Then, a *kernel function* $\kappa : X \times X \longrightarrow \mathbf{R}$. can be characterized by the following necessary and sufficient properties, see [14],

1. κ is either continuous or has a finite domain
2. κ can be computed by decomposition using a certain mapping ϕ

$$\kappa_\phi (\mathbf{x}_1, \mathbf{x}_2) = \langle \phi (\mathbf{x}_1) , \phi (\mathbf{x}_2) \rangle_{\mathcal{F}} \tag{1}$$

From the last equation we have that κ has to be positive semi-definite because of the properties of the inner product. We now assume that the kernel is differentiable with respect to the arguments. Using the linearity in the Hilbert-space $\mathcal{F}$, dot products of data with the elements of $\mathcal{F}$ generated by X and ϕ can be described as $\mathcal{F}_X = \left\{ \sum_{i=1}^{l} \alpha_i \kappa (\mathbf{x}_i, \mathbf{y}) : l \in \mathbf{N}, \mathbf{x}_i \in X, \alpha_i \in \mathbf{R} \right\}$. This property is used in [9], adapting the α_i to derive a kernelization of GLVQ.

3 ALGORITHM

Learning vector quantization was introduced as a generic concept for intuitive prototype-based classification algorithms [8]. Several variants were developed to improve the standard algorithms [7, 10, 13]. GLVQ is an extension of the standard LVQ providing a cost function [10]. It has the benefit that it can be interpreted as a margin optimization method [5].

All LVQ-algorithms typically constitute distance based approaches. However, as pointed out in [6] more general *similarity measures* can be considered with the remaining restriction of differentiability. Now the idea is to replace such a general similarity measure by inner products which implies the utilization of *kernels*. In this way we obtain a kernel variant of the underlying LVQ algorithms. Focusing on GLVQ extended by a differentiable kernel we obtain the *D-KGLVQ*.

3.1 Standard GLVQ

Let $c_{\mathbf{v}} \in \mathcal{L}$ be the label of input $\mathbf{v}$, $\mathcal{L}$ a set of labels (classes) with $\#\mathcal{L} = N_{\mathcal{L}}$ and $V \subseteq \mathbf{R}^{D_V}$ be a finite set of inputs $\mathbf{v}$ with $|V| = N$ the number of data

points. LVQ uses a fixed number of prototypes (weight vectors) for each class. Let $\mathbf{W} = \{\mathbf{w_r}\}$ be the set of all prototypes and $c_\mathbf{r}$ be the class label of $\mathbf{w_r}$. Furthermore, let $\mathbf{W}_c = \{\mathbf{w_r} | c_\mathbf{r} = c\}$ be the subset of prototypes assigned to class $c \in \mathcal{L}$. Further, let d be an arbitrary (differentiable) distance measure in V. We start with the cost function for GLVQ

$$Cost_{GLVQ} = \sum_{\mathbf{v}} \mu(\mathbf{v}) \quad \mu(\mathbf{v}) = \frac{d_{\mathbf{r}_+} - d_{\mathbf{r}_-}}{d_{\mathbf{r}_+} + d_{\mathbf{r}_-}} \tag{2}$$

which has to be minimized by gradient descent. Thereby, $d_{\mathbf{r}_+}$ is determined by

$$\mathbf{v} \mapsto \mathbf{s}(\mathbf{v}) = argmin_{\mathbf{r} \in A} d(\mathbf{v}, \mathbf{w_r}) \tag{3}$$

and $d_{\mathbf{r}_+} = d(\mathbf{v}, \mathbf{w_s})$ with the additional constraint that $c_\mathbf{v} = c_\mathbf{r}$, i.e. $d_{\mathbf{r}_+}$ is the squared distance of the input vector $\mathbf{v}$ to the nearest prototype labeled with $c_{\mathbf{r}_+} = c_\mathbf{v}$. Analogously, $d_{\mathbf{r}_-}$ is defined. Note that $\mu(\mathbf{v})$ is positive if the vector $\mathbf{v}$ is misclassified and negative otherwise.

The learning rule of GLVQ is obtained taking the derivatives of the above cost function. Using $\frac{\partial \mu(\mathbf{v})}{\partial \mathbf{w_{r+}}} = \xi^+ \frac{\partial d_{\mathbf{r}_+}}{\partial \mathbf{w_{r+}}}$ and $\frac{\partial \mu(\mathbf{v})}{\partial \mathbf{w_{r-}}} = \xi^- \frac{\partial d_{\mathbf{r}_-}}{\partial \mathbf{w_{r-}}}$ with

$$\xi^+ = \frac{2 \cdot d_{\mathbf{r}_-}}{(d_{\mathbf{r}_+} + d_{\mathbf{r}_-})^2} \quad \xi^- = \frac{-2 \cdot d_{\mathbf{r}+}}{(d_{\mathbf{r}_+} + d_{\mathbf{r}_-})^2} \tag{4}$$

one obtains for the weight updates [6]:

$$\triangle \mathbf{w}_{\mathbf{r}_+} = \epsilon^+ \cdot \xi^+ \cdot \frac{\partial d_{\mathbf{r}_+}}{\partial \mathbf{w_{r+}}} \quad \triangle \mathbf{w}_{\mathbf{r}_-} = \epsilon^- \cdot \xi^- \cdot \frac{\partial d_{\mathbf{r}-}}{\partial \mathbf{w_{r-}}} \tag{5}$$

3.2 Kernelized GLVQ

We now briefly review the main concepts used in Kernelized GLVQ (KGLVQ) as given in [9]. The KGLVQ makes use of the same cost function as GLVQ but with the distance calculations done in the kernel space. Under this setting the prototypes cannot explicitly be expressed as vectors in the feature space due to lack of knowledge about the feature space. Instead in [9] the feature space is modeled as a linear combination of all images $\phi(\mathbf{v})$ of the datapoints $\mathbf{v} \in V$. Thus a prototype vector may be described by some linear combination of the feature space data sample, i.e. $\mathbf{w}_j^F = \sum_{i=1}^N \psi_{j,i} \phi(\mathbf{v_i})$, where $\psi_k \in \mathbf{R}^{|W| \times N}$ is the combinatorial coefficient vector. The distance in feature space between a sample $\phi(\mathbf{v_i})$ and the feature space prototype vector $\mathbf{w}_k^F$ can be formulated as:

$$\begin{aligned} d_{i,j}^F &= \|\phi(\mathbf{v_i}) - \mathbf{w}_j^F\|^2 = \|\phi(\mathbf{v_i}) - \sum_{i=1}^N \psi_{j,i} \phi(\mathbf{v_i})\|^2 \\ &= k(\mathbf{v_i}, \mathbf{v_j}) - 2 \sum_{m=1}^N k(\mathbf{v_j}, \mathbf{v_m}) \cdot \psi_{j,m} + \sum_{s,t}^N k(\mathbf{v_s}, \mathbf{v_t}) \cdot (\psi_{j,s} \psi_{j,t}) \end{aligned}$$

The update rules of GLVQ are modified in [9] accordingly, using the kernelized representation of the distances and prototypes. Subsequently additional derivatives with respect to the ψ parameters are determined in a direct manner. The algorithm performs all calculations in the feature space using the kernel trick and updates the model parameters by means of ψ updates. The final model consists of the pre-calculated kernel matrix and the combinatorial coefficient matrix for the ψ coefficients. The detailed equations are available in [9].

3.3 Inner product based GLVQ and Kernel GLVQ

Now we replace the squared distance measure in (2) by a *differentiable* inner product σ defining a norm d_σ.Thus, identifying any subsets by utilization of σ can be done equivalently (in topological sense) by means of the norm d_σ and vice versa. In context of GLVQ this implies that all margin analysis is still valid also for inner product based variants of GLVQ. Further, among all inner products σ those are of particular interest, which are generated by kernels κ_ϕ defined in (1), i.e. $\sigma = \kappa_\phi$. The prototypes are subsequently preimages of its kernel space counterparts. Here, using the *differentiability assumption* for the used kernels provides an alternative easier solution than the one proposed in [9]. Consider the inner product σ based classifier function

$$\mu_\sigma(\mathbf{v}) = \frac{\sigma^2_{\mathbf{r}_+} - \sigma^2_{\mathbf{r}_-}}{\sigma^2_{\mathbf{r}_+} + \sigma^2_{\mathbf{r}_-}}$$

which has to be positive if $\mathbf{v}$ is correctly classified, i.e.

$$\mathbf{v} \mapsto \mathbf{s}(\mathbf{v}) = argmax_{\mathbf{r}\in A}\left[\left(\sigma(\mathbf{v}, \mathbf{w}_\mathbf{r})\right)^2\right] \tag{6}$$

and $\sigma_{\mathbf{r}_+}$ as well $\sigma_{\mathbf{r}_-}$ play the same role as $d_{\mathbf{r}_+}$ and $d_{\mathbf{r}_-}$. The cost changes to

$$Cost_{KGLVQ} = \sum_{\mathbf{v}} \mu_\sigma(\mathbf{v})\,. \tag{7}$$

we get prototype derivatives as:

$$\frac{\partial \mu_\sigma(\mathbf{v})}{\partial \mathbf{w}_{\mathbf{r}\pm}} = \xi^\pm_\sigma \frac{\partial \sigma_{\mathbf{r}_\pm}}{\partial \mathbf{w}_{\mathbf{r}\pm}} \quad \xi^+_\sigma = \frac{4 \cdot \sigma_{\mathbf{r}+} \cdot \sigma^2_{\mathbf{r}-}}{(\sigma^2_{\mathbf{r}_+} + \sigma^2_{\mathbf{r}_-})^2} \quad \xi^-_\sigma = -\frac{4 \cdot \sigma^2_{\mathbf{r}+} \cdot \sigma_{\mathbf{r}-}}{(\sigma^2_{\mathbf{r}_+} + \sigma^2_{\mathbf{r}_-})^2}$$

The final updates for the gradient ascent are obtained as

$$\triangle \mathbf{w}_{\mathbf{r}_+} = \epsilon^+ \cdot \xi^+_\sigma \cdot \frac{\partial \sigma_{\mathbf{r}_+}}{\partial \mathbf{w}_{\mathbf{r}+}} \quad \triangle \mathbf{w}_{\mathbf{r}_-} = \epsilon^- \cdot \xi^-_\sigma \cdot \frac{\partial \sigma_{\mathbf{r}-}}{\partial \mathbf{w}_{\mathbf{r}-}} \tag{8}$$

containing the derivatives of the kernel σ. In case of the usual Euclidean inner product $\sigma_\phi(\mathbf{v}, \mathbf{w}_\mathbf{r}) = \mathbf{v}^T \cdot \mathbf{w}_\mathbf{r}$ with ϕ as identity function, one simply gets $\frac{\partial \sigma_\phi}{\partial \mathbf{w}_\mathbf{r}} = \mathbf{v}$. Yet, in case of a kernel based inner product κ_ϕ, the derivative of the inner product can easily be carried out without any explicit knowledge of the underlying function ϕ taking into account the kernel trick property. For example,

if κ_ϕ is the polynomial kernel $\kappa_\phi = \langle \mathbf{v}, \mathbf{w} \rangle^d$ we have $\frac{\partial \kappa_\phi}{\partial \mathbf{w}} = d \cdot \langle \mathbf{v}, \mathbf{w} \rangle^{d-1} \cdot \mathbf{v}$. For the *rbf-kernel*

$$\kappa_\phi \left(\mathbf{v}, \mathbf{w}, \gamma \right) = \exp \left(- \frac{\left(\mathbf{v} - \mathbf{w} \right)^2}{2\gamma^2} \right) \tag{9}$$

one obtains $\frac{\partial \kappa_\phi}{\partial \mathbf{w}} = \frac{1}{\gamma^2} \exp \left(- \frac{(\mathbf{v}-\mathbf{w})^2}{2\gamma^2} \right) (\mathbf{v} - \mathbf{w})$ whereas for the exponential kernel $\kappa_\phi = \exp \left(\langle \mathbf{v}, \mathbf{w} \rangle \right)$ this procedure yields $\frac{\partial \kappa_\phi}{\partial \mathbf{w}} = \exp \left(\langle \mathbf{v}, \mathbf{w} \rangle \right) \cdot \mathbf{v}$. Further prominent problem specific differentiable kernels are e.g. the Sobolev-Kernel which is well suited for the analysis of functional data or the Tanimoto-kernel in the context of taxonomical data [16],[15]. For further kernel examples we refer to [14].

Generalization error analysis It has been shown in [5, 12] that generalization bounds for LVQ schemes can be derived based on the notion of the hypothesis margin of the classifier, independent of the input dimensionality of the classifier, rather the margin, i.e. the difference of the distance of points to its closest correct ($\mathbf{w}_{\mathbf{r}+}$) and wrong prototype ($\mathbf{w}_{\mathbf{r}-}$) determine the generalization ability. This fact makes the algorithm particularly suitable for kernelization where the generalization ability is measured in the feature space $\mathcal{F}$ since the nonlinear feature map as well as the kernel are fixed. However, the feature map Φ typically maps to a high (probably infinite) dimensional space such that the generalization ability of classifiers can severely decrease if the generalization ability would depend on the input dimensionality of the classifier. For GLVQ as a large margin approach, a straightforward transfer of the bounds as provided in [5] based on techniques as given in [1] is possible.

Assume a classification into two classes is considered: we refer to the corresponding prototypes by $\mathbf{w}_i^S$ with $S = \pm 1$. Classification takes place by a winner takes all rule, i.e.

$$f : \mathbf{v} \mapsto \operatorname{sgn} \left(\max_{\mathbf{w}_i^+} \{ \sigma(\mathbf{v}, \mathbf{w}_i^+)^2 \} - \max_{\mathbf{w}_i^-} \{ \sigma(\mathbf{v}, \mathbf{w}_i^-)^2 \} \right) \tag{10}$$

where sgn selects the sign of the term. A trainable D-KGLVQ network corresponds to a function f in this class with N prototypes. We can assume that data $\mathbf{v}$ are limited in size and, thus, also the images $\Phi(\mathbf{v})$ and the possible location of prototype vectors are restricted by a finite size B. We assume that data and their labeling stem from a (partially unknown) probability distribution P. Generalization bounds aim at bound the generalization error $E_P(f) = P(f(\mathbf{v}) \neq c_{\mathbf{v}})$ An important role will be taken by the margin of a classification. For this purpose, the sign is dropped in (10) leading to the related function M_f. We fix a positive value, the margin, ρ and the associated loss

$$L : \mathbf{R} \to \mathbf{R}, t \mapsto \begin{cases} 1 & \text{if } t \leq 0 \\ 1 - t/\rho & \text{if } 0 < t \leq \rho \\ 0 & \text{otherwise} \end{cases}$$

Then, a connection of the generalization error and the empirical error on m samples with respect to the loss L can be established with probability $\delta > 0$

$$\hat{E}_m^L(f) = \sum_{i=1}^{m} L(c_{\mathbf{v}} \cdot M_f(\mathbf{v}))/m \tag{11}$$

simultaneously for all functions f using techniques of [1]:

$$E_P(f) \leq \hat{E}_m^L(f) + \frac{2}{\rho} R_m(M_{\mathcal{F}}) + \sqrt{\frac{\ln(4/\delta)}{2m}}$$

$R_m(M_{\mathcal{F}})$ denotes the so-called Rademacher complexity of the class of functions implemented by D-KLVQ networks with function M_f. The quantity can be upper bounded, using techniques of [12] and structural properties given in [1], by a term

$$\mathcal{O}\left(\frac{N^{2/3}B^3 + \sqrt{\ln(1/\delta)}}{\sqrt{m}}\right)$$

The quantity B depends on the concrete kernel and can be estimated depending on the data distribution. Thus, generalization bounds for D-KGLVQ with arbitrary kernel result which are comparable to generalization bounds for GLVQ.

3.4 Parameter adaptation for Gaussian kernels

Kernel width The width γ of the Gaussian kernel (9) crucially influences the performance of the classifier. Yet, an alternative is to individualize the kernel width $\gamma_{\mathbf{r}}$ for each prototype $\mathbf{w_r}$ and, afterwards treat them as parameters to be learned. As the prototypes itself, this can be done by stochastic gradient ascent on $Cost_{KGLVQ}$ based on $\frac{\partial Cost_{KGLVQ}}{\partial \gamma_{\mathbf{r}}}$. For the localized Gaussian kernel (9):

$$\frac{\partial \mu_\sigma(\mathbf{v})}{\partial \gamma_{\mathbf{r}_\pm}} = \xi_\sigma^\pm \cdot \frac{\partial \kappa_\phi\left(\mathbf{v}, \mathbf{w}_{\mathbf{r}_\pm}, \gamma_{\mathbf{r}_\pm}\right)}{\partial \gamma_{\mathbf{r}_\pm}} = \xi_\sigma^\pm \cdot \frac{\kappa_\phi\left(\mathbf{v}, \mathbf{w}_{\mathbf{r}_\pm}, \gamma_{\mathbf{r}_\pm}\right)}{\gamma_{\mathbf{r}_\pm}^3} \cdot \left(\mathbf{v} - \mathbf{w}_{\mathbf{r}_\pm}\right)^2.$$

Relevance learning The Gaussian kernel usually takes as ingredients the Euclidean norm of the vector difference, but more special choices like Sobolev-norms for functional data are also possible. Here we look at the scaled Euclidean metric

$$d^\lambda(\mathbf{v}, \mathbf{w}) = \sum_i \lambda_i \cdot (v_i - w_i)^2 \qquad \sum_i \lambda_i = 1$$

As usual in relevance learning [6], the scaling parameters λ_i can be adapted with respect to the classification task at hand by gradient learning, leading to a gradient ascent but now as $\frac{\partial Cost_{KGLVQ}}{\partial \lambda_i}$. Considering $\frac{\partial \mu_\sigma(\mathbf{v})}{\partial \lambda_i}$ we obtain for $\mathbf{w_{r\pm}}$

$$\frac{\partial \mu_\sigma(\mathbf{v})}{\partial \lambda_i} = \xi_\sigma^+ \cdot \frac{\partial \kappa_\phi\left(\mathbf{v}, \mathbf{w}_{\mathbf{r}_\pm}, \gamma_{\mathbf{r}_\pm}\right)}{\partial \lambda_i} = -\xi_\sigma^+ \cdot \frac{\kappa_\phi\left(\mathbf{v}, \mathbf{w}_{\mathbf{r}_\pm}, \gamma_{\mathbf{r}_\pm}\right)}{2\gamma^2} \left(v_{\mathbf{r}_\pm, i} - w_{\mathbf{r}_\pm, i}\right)^2$$

We denote this approach as *Kernelized Relevance GLVQ (D-KGRLVQ).* [4].

[4] Extendable to matrix learning [2], giving *Kernelized Matrix GLVQ (D-KGMLVQ)*

4 EXPERIMENTS

We present a comparison for 5 benchmark datasets 3 derived from the UCI [4] and the other 2 from [11]. We analyze the performance of GLVQ, KGLVQ, D-KGLVQ and SVM using Radial Basis Function (RBF) kernels. The σ parameter of the RBF kernel has been optimized using a separate test set, evaluated in a 3-fold cross validation in a range of $\{1e^{-6}, 1e^{-5}, \ldots, 1e^{6}\}$ and fine tuned for D-KGLVQ using the proposed gamma-learning scheme. We observed an improvement of the D-KGLVQ in the generalization of around 1%. For KGLVQ the parameter settings have been found to be very sensitive while for D-KGLVQ and SVM the specific setting of the sigma parameter was quite robust.

GLVQ, D-KGLVQ and KGLVQ have been trained with 1 prototype per class. In comparison to GLVQ the other methods show improved results demonstrated by the mean errors over the datasets see Table 1. We find that the performance of D-KGLVQ and KGLVQ are quite similar, and both are competitive to SVM. The D-KGLVQ allows for metric adaptation like in [11] to e.g. identify rele-

	Dim	Dataset size	GLVQ Error/#PT	D-KGLVQ Error/#PT	KGLVQ Error/#PT	SVM Error/#SV
Breast Cancer	32	569	26.19/2	08.00/2	07.30/2	02.64/74
Diabetes	8	768	28.26/2	30.00/2	27.00/2	23.32/370
Heart	13	270	25.93/2	17.00/2	18.81/2	15.43/102
Colorectal Cancer	1408	95	23.16/2	16.25/2	17.87/2	11.58/57
Lung Cancer	1408	100	34.00/2	29.00/2	27.50/2	25.00/65
Mean			25.51/2	20.05/2	19.68/2	15.59/134

Table 1. Generalization error and model complexity (averaged) for the datasets.

vant individual features in the original data space. Individual prototypes can be analyzed with respect to their receptive fields. Sets of data points, represented by one prototype in a high-dimensional space can be related back to belong to each other and can be considered to be similar in its characteristics. This is not possible using SVM models because their model parameters are extreme points rather prototypes. Considering the model complexity we find that with only 2 prototypes the LVQ methods perform quite well. Especially for D-KGLVQ a very compact model is obtained whereas for KGLVQ the model contains additionally the kernel matrix and the combinatorial coefficient matrix. For SVM the performance is a bit better than for all other methods but with a large number of support vectors in the model.

5 CONCLUSIONS

In this paper we derived a novel kernelized learning vector quantizer employing differentiable kernel functions. We provided generalization error bounds analogous to that used in [5]. We presented experimental results on real world datasets which showed that D-KGLVQ is competitive with both KGLVQ and SVM. In

terms of computational complexity the demands of D-KGLVQ during training are significantly lower with respect to KGLVQ because no large combinatorial coefficient matrix has to be calculated and stored. As pointed out in Section 2 the D-KGLVQ can be easily extended to use problem specific parametrized kernel functions. The parameters of the kernel functions can also be subject of the optimization provided by D-KGLVQ. The solutions generated by D-KGLVQ are in the kernel space but the model parameters keep the prototype concept and are therefore compact and easier to interpret than typical kernel models.

References

1. Bartlett, P., Mendelson, S.: Rademacher and Gaussian complexities: risk bounds and structural results. Journal of Machine Learning Research 3, 463–482 (2003)
2. Biehl, M., Hammer, B., Schleif, F.M., Schneider, P., Villmann, T.: Stationarity of matrix relevance learning vector quantization. Machine Learning Reports 3, 1–17 (2009), ISSN:1865-3960 http://www.uni-leipzig.de/compint/mlr/mlr_01_2009.pdf
3. Bishop, C.: Pattern Recognition and Machine Learning. Springer, NY (2006)
4. Blake, C., Merz, C.: UCI repository of machine learning databases. Irvine, CA: University of California, Department of Information and Computer Science, available at: http://www.ics.uci.edu/ mlearn/MLRepository.html (1998)
5. Crammer, K., Gilad-Bachrach, R., A.Navot, A.Tishby: Margin analysis of the LVQ algorithm. In: Proc. NIPS 2002. pp. 462–469 (2002)
6. Hammer, B., Villmann, T.: Generalized relevance learning vector quantization. Neural Networks 15(8-9), 1059–1068 (2002)
7. Hammer, B., Villmann, T.: Mathematical aspects of neural networks. In: Verleysen, M. (ed.) Proc. Of European Symposium on Artificial Neural Networks (ESANN'2003). pp. 59–72. d-side, Brussels, Belgium (2003)
8. Kohonen, T.: Self-Organizing Maps, Springer Series in Information Sciences, vol. 30. Springer, Berlin, Heidelberg (1995), (Second Extended Edition 1997)
9. Qin, A.K., Suganthan, P.N.: A novel kernel prototype-based learning algorithm. In: Proc. of ICPR'04. pp. 2621–624 (2004)
10. Sato, A.S., Yamada, K.: Generalized learning vector quantization. In: Tesauro, G., Touretzky, D., Leen, T. (eds.) Advances in Neural Information Processing Systems, vol. 7, pp. 423–429. MIT Press (1995)
11. Schleif, F.M., Villmann, T., Kostrzewa, M., Hammer, B., Gammerman, A.: Cancer informatics by prototype-networks in mass spectrometry. Artificial Intelligence in Medicine 45, 215–228 (2009)
12. Schneider, P., Biehl, M., Hammer, B.: Adaptive relevance matrices in learning vector quantization. Neural Computation (to appear)
13. Seo, S., Obermayer, K.: Soft learning vector quantization. Neural Computation 15, 1589–1604 (2003)
14. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis and Discovery. Cambridge University Press (2004)
15. Simmuteit, S., Schleif, F.M., Villmann, T., Elssner, T.: Tanimoto metric in treesom for improved representation of mass spectrometry data with an underlying taxonomic structure. In: Proc. of ICMLA 2009. pp. 563–567. IEEE Press (2009)
16. Villmann, T., Schleif, F.M.: Functional vector quantization by neural maps. In: Proceedings of Whispers 2009. p. CD (2009)