Hyperparameter Learning in Probabilistic Prototype-based models

Petra Schneider[a], Michael Biehl[a], Barbara Hammer[b]

[a] *University of Groningen - Institute of Mathematics and Computing Science*
P.O. Box 407, 9700 AK Groningen, The Netherlands
[b] *Clausthal University of Technology - Institute of Computer Science*
Julius Albert Strasse 4, 38678 Clausthal-Zellerfeld, Germany

Abstract

We present two approaches to extend Robust Soft Learning Vector Quantization (RSLVQ). This algorithm for nearest prototype classification is derived from an explicit cost function and follows the dynamics of a stochastic gradient ascent. The RSLVQ cost function is defined in terms of a likelihood ratio and involves a hyperparameter which is kept constant during training. We propose to adapt the hyperparameter in the training phase based on the gradient information. Besides, we propose to base the classifiers' decision on the value of the likelihood ratio instead of using the distance based classification approach. Experiments on artificial and real life data show that the hyperparameter crucially influences the performance of RSLVQ. However, it is not possible to estimate the best value from the data prior to learning. We show that the proposed variant of RSLVQ is very robust with respect to the initial value of the hyperparameter. The classification approach based on the likelihood ratio turns out to be superior to distance based classification, if local hyperparameters are adapted for each prototype.

Key words: Learning Vector Quantization, Robust Soft LVQ, distance based classification, likelihood, cost function, hyperparameter

1. Introduction

Learning Vector Quantization (LVQ) constitutes a family of supervised learning algorithms which are widely used for the classification of potentially high-dimensional data [1]. The basic approach is very intuitive: classification is based on a set of so-called prototype vectors representing the classes, and a

new feature vector is assigned to the class represented by the closest prototype. LVQ can naturally deal with multi-class problems and the prototype vectors allow for an immediate interpretation of the resulting classifier. Kohonen introduced the original LVQ scheme in 1986 [2] which uses Hebbian learning to adapt the prototypes to the data. Meanwhile, researchers proposed numerous modifications of the basic learning scheme. Recent variations of LVQ can be derived from an explicit cost function [6, 7, 8] or allow for the incorporation of adaptive distance measures [3, 4, 5]. In [9], the authors propose a likelihood-based decision metric for LVQ classifiers which are defined in terms of a set of probabilisitc models. A similar classification criterion will be considered in this article. However, our method still obeys the fundamental LVQ concept of prototypes defined in the space of the input data. LVQ variants are very attractive due to their computational simplicity and flexibility. For this reasons, the methods have been used in a huge variety of applications such as image analysis, satellite remote sensing or bioinformatics [10, 11, 12]. Furthermore, theoretical understanding of LVQ schemes has been advanced in years [13, 14, 15].

The learning rules of several LVQ procedures involve a hyperparameter, such as the window size in LVQ2.1 [2] or the softness parameter σ^2 in Soft LVQ [7] and Robust Soft LVQ (RSLVQ) [7]. The hyperparameter can have high impact on the performance of the resulting classifier. Usually, it is kept constant in the learning process, and it is chosen from a set of candidates by means of a validation procedure. In [8], an annealing schedule is proposed to reduce the respective hyperparameter of an LVQ algorithm in the course of training. However, this schedule is purely heuristically motivated and does not follow any learning objective.

This paper focuses on RSLVQ which is based on a well defined stochastic model of LVQ classification schemes. Training is based on the objective of likelihood optimization. The learning rules are derived from an explicit cost function which is optimized with respect to the model parameters. The aim of our study is twofold: we introduce a well-founded strategy to deal with the hyperparameter, and we propose a new decision rule to classify the data. Since the cost function also depends on σ^2, we propose to introduce the hyperparameter as a further degree of freedom and to maximize the objective function with respect to σ^2 as well. Further, we compare the performances of simple nearest prototype classification and schemes which are explicitly based on the likelihood. The latter corresponds naturally to the objective of training.

In our experiments, we demonstrate the influence of the hyperparameter on the classification accuracy of RSLVQ and study the effect of the proposed optimization method using artificial data and real-life data sets. Further, we show the superiority of likelihood based classification in particular for local adaptation schemes.

2. Robust Soft LVQ

Assume training data $\{\boldsymbol{\xi}_k, y_k\}_{k=1}^{l} \in \mathbb{R}^N \times \{1, \ldots, C\}$ are given, N denoting the data dimensionality and C the number of different classes. An LVQ network $W = \{(\boldsymbol{w}_j, c(\boldsymbol{w}_j)) : \mathbb{R}^N \times \{1, \ldots, C\}\}_{j=1}^{m}$ consists of a number of prototypes $\boldsymbol{w}_j \in \mathbb{R}^N$ which are characterized by their location in feature space and their class label $c(\boldsymbol{w}_j) \in \{1 \ldots, C\}$. Given a distance measure $d(\boldsymbol{\xi}, \boldsymbol{w})$ in $\mathbb{R}^N$, classification is based on a winner-takes-all scheme: a data point $\boldsymbol{\xi} \in \mathbb{R}^N$ is assigned to the label $c(\boldsymbol{\xi}) = c(\boldsymbol{w}_i)$ of the prototype i with $d(\boldsymbol{\xi}, \boldsymbol{w}_i) \leq d(\boldsymbol{\xi}, \boldsymbol{w}_j), \forall j \neq i$. Many LVQ variants use the squared Euclidean distance $d(\boldsymbol{\xi}, \boldsymbol{w}) = (\boldsymbol{\xi} - \boldsymbol{w})^T(\boldsymbol{\xi} - \boldsymbol{w})$ to quantify the similarity between feature vectors and the prototypes.

The Robust Soft LVQ - algorithm [7] to train the prototype locations is based on a statistical modeling of the given data distribution, i.e. the probability density is described by a mixture model. It is assumed that every component j of the mixture generates data which belongs to only one of the C classes. The probability density of the data is approximated by

$$p(\boldsymbol{\xi}|W) = \sum_{i=1}^{C} \sum_{j:c(\boldsymbol{w}_j)=i} P(j)\, p(\boldsymbol{\xi}|j), \tag{1}$$

where $\sum_j P(j) = 1$, and the conditional density $p(\boldsymbol{\xi}|j)$ is a function of the prototype $\boldsymbol{w}_j$. A possible choice is the normalized exponential form $p(\boldsymbol{\xi}|j) = K(j) \cdot \exp f(\boldsymbol{\xi}, \boldsymbol{w}_j, \sigma_j^2)$. In [7] a Gaussian mixture is assumed with $K(j) = (2\pi\sigma_j^2)^{-N/2}$ and $f(\boldsymbol{\xi}, \boldsymbol{w}_j, \sigma_j^2) = -d(\boldsymbol{\xi}, \boldsymbol{w}_j)/2\sigma_j^2$; where d is the squared Euclidean distance, and every component is assumed to have equal variance $\sigma_j^2 = \sigma^2$ and equal prior probability $P(j) = 1/m$, $\forall j$. RSLVQ maximizes the likelihood ratio

$$L = \prod_{k=1}^{l} L(\boldsymbol{\xi}_k, y_k), \text{where} \quad L(\boldsymbol{\xi}_k, y_k) = \frac{p(\boldsymbol{\xi}_k, y_k|W)}{p(\boldsymbol{\xi}_k|W)} \tag{2}$$

with respect to the prototype locations by means of stochastic gradient ascent. $p(\boldsymbol{\xi}, y|W)$ is the probability density that sample $\boldsymbol{\xi}$ is generated by a component of the correct class y. This local density corresponds to the inner sum in Eq. (1). The learning rule is obtained by taking the derivatives of the RSLVQ cost function $E = \log(L)$ with respect to $\boldsymbol{w}_j$ (see [7])

$$\Delta \boldsymbol{w}_j = \frac{\alpha_1}{\sigma^2} \begin{cases} (P_y(j|\boldsymbol{\xi}) - P(j|\boldsymbol{\xi}))(\boldsymbol{\xi} - \boldsymbol{w}_j), & c(\boldsymbol{w}_j) = y, \\ -P(j|\boldsymbol{\xi})(\boldsymbol{\xi} - \boldsymbol{w}_j), & c(\boldsymbol{w}_j) \neq y, \end{cases} \tag{3}$$

where $\alpha_1 > 0$ is the learning rate, and $P_y(j|\boldsymbol{\xi})$ and $P(j|\boldsymbol{\xi})$ are assignment probabilities

$$P_y(j|\boldsymbol{\xi}) = \frac{\exp f(\boldsymbol{\xi}, \boldsymbol{w}_j, \sigma^2)}{\sum_{i:c(\boldsymbol{w}_i)=y} \exp f(\boldsymbol{\xi}, \boldsymbol{w}_i, \sigma^2)}, \qquad P(j|\boldsymbol{\xi}) = \frac{\exp f(\boldsymbol{\xi}, \boldsymbol{w}_j, \sigma^2)}{\sum_i \exp f(\boldsymbol{\xi}, \boldsymbol{w}_i, \sigma^2)} \tag{4}$$

with respect to one example $(\boldsymbol{\xi}, y)$. The update rules reflect the fact that prototypes with $c(\boldsymbol{w}) = y$ are attracted by the training sample, while prototypes carrying any other class label are repelled.

The training dynamics of RSLVQ highly depends on the hyperparameter σ^2: since it determines the value of the assignment probabilities (see Eq. (4)), it controls the strength of the attractive and repulsive forces in Eq. (3). In the limit $\sigma^2 \to 0$, RSLVQ reduces to a learning-from-mistakes scheme, i.e. only in case of erroneous classification, the closest correct and incorrect prototype are updated. In the soft version of the algorithm, all training samples lying in an active region around the decision boundary cause an update of the prototype constellation; at the same time, a larger number of prototypes is adapted in each learning step (see [7] for details).

As already stated, classification in RSLVQ networks is commonly based on a winner assignment, i.e. a point is mapped to the class of its closest prototype. Hence, although the training procedure optimizes the likelihood ratio $L(\boldsymbol{\xi}, y)$, its value is irrelevant for the classifiers' decision in the working phase. Instead, the algorithm uses the standard LVQ-approach of nearest prototype classification based on the squared Euclidean distance. This may be inappropriate, since learning and classification aim at different objectives.

3. Modifications of Robust Soft LVQ

3.1. Hyperparameter adaptation in RSLVQ

In [8], a heuristic approach is introduced to anneal the value of the hyperparameter in the course of training. The authors propose a schedule which reduces σ^2 continuously in each learning step. This may lead to non-monotonic

learning curves, as the performance deteriorates when σ^2 becomes lower than the potential optimum. Hence, the method has to be used in combination with an early stopping procedure.

In this work, we propose a more systematic approach to treat the hyperparameter according to the optimization of the likelihood ratio in Eq. (2). We adapt the hyperparameter according to the gradient of E with respect to σ^2. The derivative $\frac{\partial E}{\partial \theta}$ of the RSLVQ cost function with respect to any global or local parameter θ of the system can be found in [5]. The general form $\frac{\partial E}{\partial \theta}$ in combination with

$$\frac{\partial f(\boldsymbol{\xi}, \boldsymbol{w}, \sigma^2)}{\partial \sigma^2} = \frac{(\boldsymbol{\xi} - \boldsymbol{w})^T (\boldsymbol{\xi} - \boldsymbol{w})}{2\,\sigma^4}$$

leads us to the update rule

$$\Delta\sigma^2 = \alpha_2 \sum_j \left(\left(\delta_{y,c(\boldsymbol{w}_j)} \left(P_y(j|\boldsymbol{\xi}) - P(j|\boldsymbol{\xi})\right) - (1 - \delta_{y,c(\boldsymbol{w}_j)})\, P(j|\boldsymbol{\xi}) \right) \frac{d(\boldsymbol{\xi}, \boldsymbol{w}_j)}{\sigma^4} \right).$$

The Kronecker symbol $\delta_{y,c(\boldsymbol{w}_j)}$ tests whether the labels $c(\boldsymbol{w}_j)$ and y coincide, and $\alpha_2 > 0$ is the learning rate. The method becomes even more flexible by training an individual hyperparameter σ_j^2 for every prototype $\boldsymbol{w}_j$. Due to the derivative in [5], in combination with

$$\frac{\partial f(\boldsymbol{\xi}, \boldsymbol{w}, \sigma_j^2)}{\partial \sigma_j^2} = \frac{(\boldsymbol{\xi} - \boldsymbol{w})^T (\boldsymbol{\xi} - \boldsymbol{w})}{2\,\sigma_j^4}, \qquad \frac{\partial K(j)}{\partial \sigma_j^2} = -\frac{N}{2} \frac{1}{(2\pi\sigma_j^2)^{N/2}\,\sigma_j^2}$$

we obtain the learning rule

$$\Delta\sigma_j^2 = \frac{\alpha_2}{\sigma_j^2} \cdot \begin{cases} \left(P_y(j|\boldsymbol{\xi}) - P(j|\boldsymbol{\xi})\right) \left(-N + d(\boldsymbol{\xi}, \boldsymbol{w}_j)/\sigma_j^2\right), & c(\boldsymbol{w}_j) = y, \\ -P(j|\boldsymbol{\xi}) \left(-N + d(\boldsymbol{\xi}, \boldsymbol{w}_j)/\sigma_j^2\right), & c(\boldsymbol{w}_j) \neq y. \end{cases}$$

Using this approach, the update rules for the prototypes in Eq. (3) also include the local hyperparameters σ_j^2.

Note that σ^2 or σ_j^2, respectively, play the role of the variance of local Gaussians when modelling the distribution of the probability density of data in this likelihood framework. As such, the question occurs whether these parameters converge to the variance underlying the data distribution, which would allow a prior estimation of these parameters from the data. We will show in experiments, that this is not the case in general. The optimum choice of σ^2 is subtle, and an automatic adaptation scheme as proposed in

this work constitutes the method of choice. These issues arise two reasons: on the one hand, the variance is optimized within a discriminative framework such that values which do not coincide with the underlying data distribution might be favorable. Further, the variance controls the influence of training points on the adaptation scheme and it determines the size of the region of interest during training. Unlike heuristics, such as window schemes in LVQ2.1 [2], automatic adaptation provides a principled way to optimize these parameters.

3.2. Decision rule based on likelihood ratio

Beyond the new strategy for dealing with the parameter σ^2, we propose to base the classification on the likelihood ratio defined in Eq. (2). The closest prototype scheme as described in Sec. 2 is replaced by a highest likelihood ratio classification, i.e. a feature vector $\boldsymbol{\xi}$ is assigned to class i with $L(\boldsymbol{\xi}, i) > L(\boldsymbol{\xi}, j), \forall j \neq i$. Note that the two different approaches lead to the same decision in case of LVQ-systems with one prototype per class and a global hyperparameter σ^2. However, different classification results can be obtained in case of a larger number of prototypes per class and/or training of individual hyperparameters for each prototype as proposed in Sec. 3.1.

This approach has no influence on learning rules of RSLVQ. It affects, however, the classification performance of a given system after training.

4. Experiments

In a first set of experiments, the proposed extensions of RSLVQ are applied to artificial toy data. The data sets consist of spherical Gaussian clusters in a two-dimensional space and correspond to binary classification problems. We investigate the relation between the hyperparameter σ^2 and the variance of the Gaussians. Furthermore, we highlight differences between the alternative decision rules based on distance measure and likelihood ratio.

In order to evaluate the performance of the new techniques in real life situations, the methods are applied to the *Letter* data set and the *Pendigits* data set from the UCI repository of machine learning [16].

In all experiments, the learning rates are continuously reduced in the course of learning. We use the schedule $\alpha_{1,2}(t) = \alpha_{1,2}/(1 + c\,(t-1))$, where t denotes the number of sweeps through the training set. The parameter c determines the speed of annealing; c is chosen for every application individually. To initialize the prototypes, we choose the mean values of random subsets of data points selected from each class.

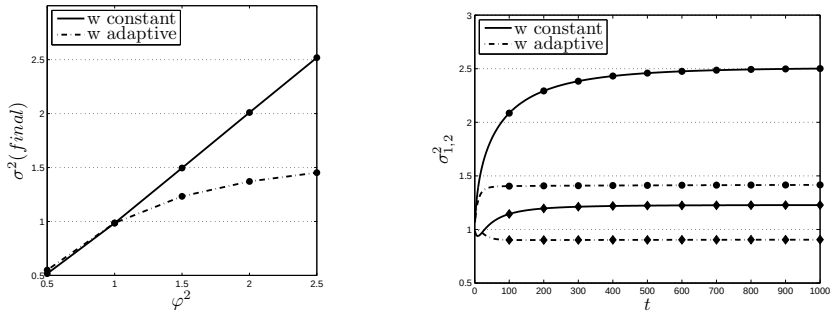

Figure 1: **Left:** Variance φ^2 of the Gaussians vs. mean final value of the global hyperparameters σ^2 obtained on data sets with two clusters of equal variance and constant and adaptive prototypes. **Right:** Evolution of the local hyperparameters $\sigma^2_{1,2}$ as a function of training time observed on a data set of two Gaussians with unequal variance and constant and adaptive prototypes. The variances are $\varphi^2_1 = 2.5$ and $\varphi^2_2 = 1.25$. The symbols correspond to σ^2_1 (•), σ^2_2 (⧫).

4.1. Artificial data sets

The adaptation of a global hyperparameter is analysed by means of data sets consisting of two Gaussian clusters of equal variance. The clusters are centered at $\mu_1 = [-2, 0]$, $\mu_2 = [2, 0]$ and consist of 1 000 data points each. The data sets differ with respect to the cluster variances φ^2 which vary between 0.5 and 2.5. At first, we fix the prototypes to the mean values of the distributions in order to analyse the adaptation of σ^2 independently of the other parameters of the system. In the next experiments, the hyperparameter and the prototypes are optimized simultaneously. The learning parameters are set to $\alpha_1 = 0.01$, $\alpha_2 = 0.001$ and $c = 0.001$. The softness is initialized by $\sigma^2(0) = 1$. We choose the mean values of random subsets of training samples from each class to initialize the prototypes and train the system for 1 000 epochs. The results presented in the following are averaged over experiments on ten statistically independent data sets.

Fig. 1 (left) visualizes the mean final values of the hyperparameter obtained on the different data sets as a function of φ^2. If the prototypes are constant and are placed in the cluster centers, σ^2 converges towards the variance of the Gaussians. However, σ^2 approaches smaller values in the experiments with adaptive prototypes. Concurrently, we observe that the prototypes saturate closer to the decision boundary as φ^2 increases.

These results are also confirmed by further experiments with two spherical clusters of different variance $\varphi^2_{1,2}$ and local adaptive hyperparameter; see Fig. 1, right, for an example.

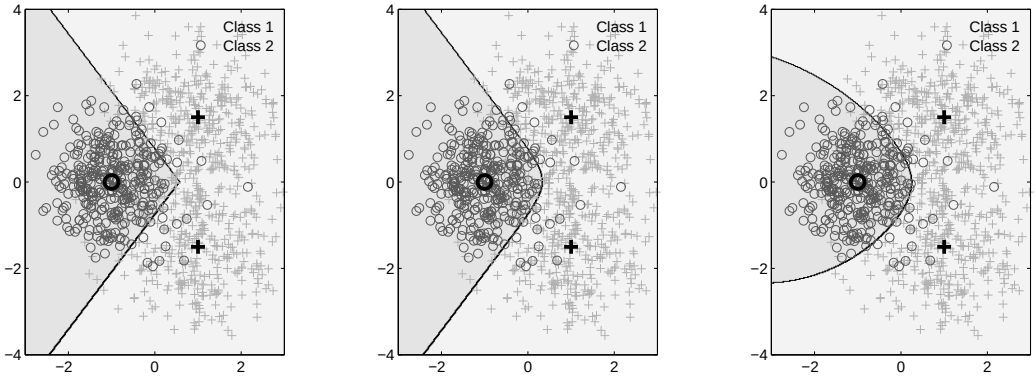

Figure 2: Receptive fields induced by distance based classification and likelihood ratio based classification for a two-class problem consisting of three clusters. The cluster means serve as prototypes. **Left:** Distance based classification using the squared Euclidean distance. **Middle:** Likelihood ratio based classification after training of a global hyperparameter σ^2. **Right:** Likelihood ratio based classification after training of local hyperparameters $\sigma^2_{1,2,3}$.

Hence, maximizing the likelihood ratio in Eq. (2) corresponds to a density estimation only if the prototypes correspond to the cluster means. However, the optimal hyperparameter cannot be estimated from the data directly, if the classifier is also optimized with respect to the prototype positions. This holds because of three reasons: for multi-modal data sets with several prototypes per class, the assignment of data to prototypes is not clear a priori, such that no statistical estimations can be made. Even for data sets which are represented using only one prototype per class, an estimation of σ^2 from the data is not obvious since, besides the bandwidth, σ^2 determines the influence of training points on the adaptation and hence the overall dynamics. Further, prototypes do not necessarily coincide with the class centers, rather, prototype locations and bandwidth are adapted by the learning rule to give an optimum decision boundary.

Finally, we compare the decision boundaries induced by nearest prototype classification and the decision rule based on the likelihood ratio. For this purpose, a data set consisting of three clusters is used; it is visualized in Fig. 2. The mean value of the class 1 data is $\mu_1 = [-1, 0]$. The class two data is split into two clusters centered at $\mu_2 = [1, 1.5]$ and $\mu_3 = [1, -1.5]$. The variances constitute $\varphi_1^2 = 0.5$, $\varphi_2^2 = 0.8$ and $\varphi_3^2 = 1.0$. Each cluster consists of 1 000 samples. According to the priorly known distribution, the

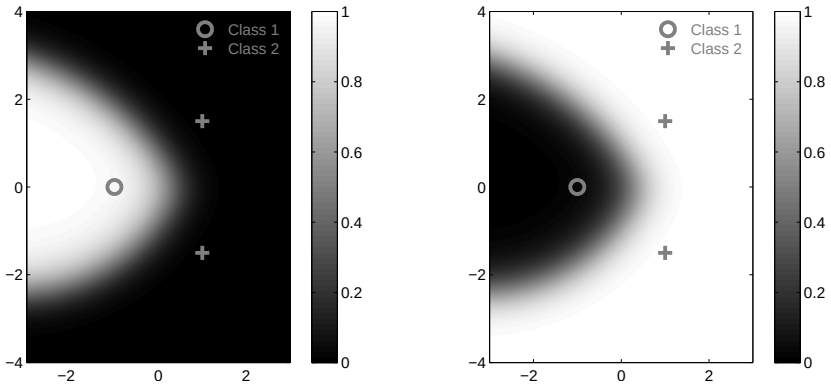

Figure 3: Visualization of the likelihood ratio as a function of location in feature space; see Fig. 2 for the data set. The cluster means serve as prototypes and individual hyperparameters $\sigma^2_{1,2,3}$ are optimized for each prototype. **Left:** Likelihood ratio $L(\boldsymbol{\xi}, 1)$ for class 1 membership. **Right:** Likelihood ratio $L(\boldsymbol{\xi}, 2)$ for class 2 membership.

data is approximated by three prototypes. We set the prototypes fixed to the mean values $\mu_{1,2,3}$ and adapt global and local hyperparameters. We use the learning parameter settings $\alpha_2 = 1 \cdot 10^{-4}$, $c = 1 \cdot 10^{-4}$, $\sigma^2(0) = \sigma_j^2(0) = 0.1, \forall j$ and train for 1 000 epochs. On average, the global hyperparameter saturates at $\sigma^2 \approx 0.7$. Similar to the previous experiments, the values $\sigma^2_{1,2,3}$ approach $\varphi^2_{1,2,3}$. Fig. 2 visualizes the receptive fields for the alternative decision rules resulting from these prototype and hyperparameter settings. Distance based classification with the squared Euclidean distance leads to piecewise linear decision boundaries. Remarkably, the receptive fields are no longer separated by straight lines, if a sample is assigned to the class of highest likelihood ratio. The effect is even more pronounced when local softness parameters can be assigned to the prototypes as displayed in th right-most panel of Fig. 2. A visualization of the likelihood ratios as a function of location in feature space is given in Fig. 3.

4.2. Letter data set

The data set consists of 20 000 feature vectors which encode 16 numerical attributes of black-and-white rectangular pixel displays of the capital letters of the English alphabet; hence, a 26-class problem is dealt with. We split the data randomly into a training and a test set of equal size. The following results are averaged over ten different compositions of training and test data.

At first, we focus on RSLVQ-training with global σ^2 and adapt one prototype per class. Note that the two alternative approaches for classification

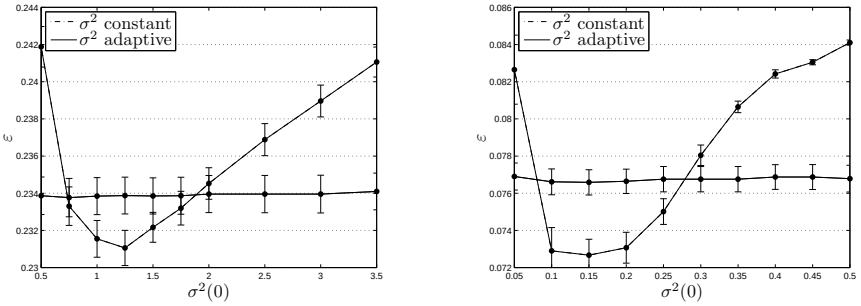

Figure 4: Mean test performance at the end of RSLVQ-training with constant and adaptive hyperparameter as a function of the initial value $\sigma^2(0)$. The error bars specify the standard error of the mean. **Left:** *Letter* data set **Right:** *Pendigits* data set.

always lead to the same decision in this case. We train prototypes and hyperparameter with different initial settings $\sigma^2(0)$ and compare the results to equivalent RSLVQ experiments without σ^2-adaptation. We choose various values $\sigma^2(0)$ from the interval [0.5, 3.5]. The remaining learning parameters are set to $\alpha_1 = 0.01$, $\alpha_2 = 0.001 \cdot \sigma^2(0)$ and $c = 0.1$. Training is continued for 100 epochs.

The experiments with constant σ^2 show that the performance of RSLVQ is highly sensitive with respect to the value of the hyperparameter (see Fig. 4, left). The lowest mean rate of misclassification on the test sets is achieved with $\sigma^2_{opt} = 1.25$; the performance constitutes $\varepsilon_{test} \approx 23.1\%$. However, the curve in Fig. 4 shows a very sharp minimum, indicating a strong dependence of the classification performance on the value of the hyperparameter. For small $\sigma^2 < 1$, we observe instabilities and highly fluctuating learning curves.

Remarkably, by including the proposed optimization scheme for global hyperparameter into the training, the sensitivity of the algorithm with respect to σ^2 can be eliminated. In all experiments with adaptive hyperparameter, the mean test error saturates at $\varepsilon_{test} \approx 23.4\%$, independent of the initial setting $\sigma^2(0)$ (see Fig. 4, left). Furthermore, the initialization $\sigma^2(0)$ does not influence the final value of the hyperparameter. As depicted in Fig. 5, left, the parameter converges towards $\sigma^2_{final} \approx 1.8$ in all experiments. Hence, the proposed variant of RSLVQ is much more robust related to the initial choice of the hyperparameter. Especially for large values $\sigma^2(0)$, the proposed optimization method achieves a clear improvement in classification performance and speed of convergence, compared to RSLVQ training with constant σ^2. However, despite the extended flexibility, learning with constant $\sigma^2 = \sigma^2_{opt}$

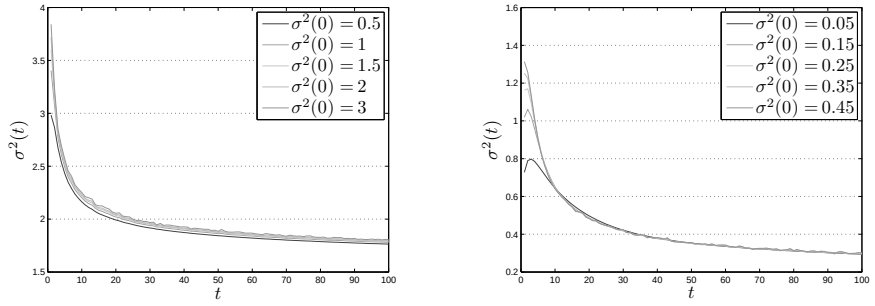

Figure 5: Evolution of the global hyperparameter σ^2 as a function of training time for different initial settings $\sigma^2(0)$. **Left:** *Letter* data set. **Right:** *Pendigits* data set.

still achieves a slightly better performance than our method. This observation can be explained by the fact that the relation between the RSLVQ cost function and the classification performance is not obvious. The optimum of the likelihood ratio does not necessarily coincide with minimal rate of misclassification. Nevertheless, the learning strategy for σ^2 may simplify the identification of σ^2_{opt} to achieve the optimal classification performance.

The adaptation of local hyperparameters is continued for 300 epochs. The error curves converge on constant level after ca. 150 epochs. Interestingly, the classification performance differs significantly for the alternative decision rules. As depicted in Fig. 6, left, highest likelihood ratio classification achieves $\approx 2\%$ lower mean error rate on the test samples. However, a slight dependence of the error rate on the initialization $\sigma^2_j(0)$ can be observed. In contrast to our experiments with global σ^2, the final local hyperparameters spread more for different settings $\sigma^2_j(0)$. This is visible in Fig. 7, left, which compares the evolution of $\sigma^2_{16}(t)$ for different initializations; the curves are representative for all σ^2_j. We expect the differences to vanish for smaller learning rates and training over a larger number of epochs. Note that the performance of distance based classification even degrades due to the adaptation of local hyperparameters, if compared to the experiments with global σ^2.

Furthermore, we analyse how the number of prototypes affects the performance of the alternative classification strategies. We train five prototypes per class and adapt a global hyperparameter. Due to the larger number of prototypes, σ^2 approaches to smaller values compared to the previous experiments; on average, it saturates at $\sigma^2_{final} \approx 0.8$, independent of $\sigma^2(0)$. Although the classification improves significantly due to the larger number

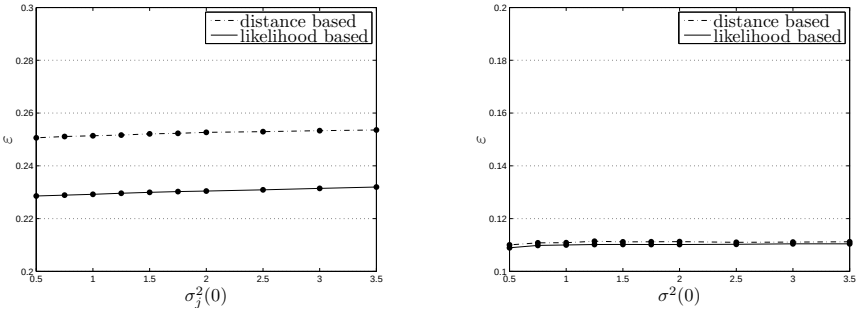

Figure 6: *Letter* data set. Mean final test errors after training of global and local hyperparameters as a function of the initial values $\sigma^2(0)$ and $\sigma_j^2(0)$. The plots compare the performance yielded by the alternative decision rules for classification. **Left:** Experiments with local adaptive hyperparameters and one prototype per class. **Right:** Experiments with global adaptive hyperparameter and five prototypes per class.

of prototypes, distance based classification and likelihood ratio based classification achieve nearly the same error rate of $\varepsilon_{test} \approx 11\%$ in all experiments (see Fig. 6, right).

Apparently, only training of local hyperparameters causes a significant difference between the two alternative decision rules. Compared to RSLVQ with a global hyperparameter, the adaptation of local σ_j^2 improves the performance of likelihood ratio based classification, but decreases the performance of distance based classification. In contrast, the performance of both approaches is nearly equal, if a larger number of prototypes is used in combination with a global parameter σ^2. The first observation concerning the effect of local hyperparameter adaptation is also confirmed in further experiments with more than one prototype per class.

Known classification results of the support vector machine (SVM) [17] vary between 15.5% and 8.5% mean error on the test sets depending on the kernel-function [1]. The k-nearest neighbour classifier [17] achieves $\varepsilon_{test} \approx 5.6\%$ with $k = 4$. However, we would like to stress that our main interest in the experiments is related to the analysis of our methods in comparison to original RSLVQ. For this reason, further validation procedures to optimize the classifiers with respect to the number of prototypes or the incorporation of adaptive distance measures were not examined in this study.

[1] We thank U. Bodenhofer, Johannes Kepler University Linz, Austria, for providing the results.

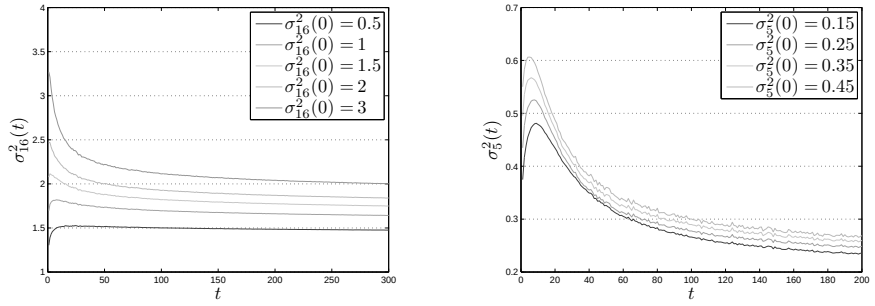

Figure 7: Evolution of local hyperparameters σ_j^2 as function of training time for different initial settings $\sigma_j^2(0)$. **Left:** *Letter* data set, $j = 16$. **Right:** *Pendigits* data set, $j = 5$.

4.3. Pendigits data set

This data set realizes a 10 class classification problem in a 16 dimensional feature space. The classification task consists in the recognition of the handwritten digits 0 to 9. The data was collected from 44 writers; each writer provided 250 samples. The samples of 33 writers form the training set, the data of the remaining 11 writers form the test set. As a preprocessing step, we normalize the data to zero mean and unit variance features. We perform all experiments with 10 different prototype initializations and present the averaged results in the following.

The first set of experiments deals with the adaptation of a global hyperparameter. We use one prototype per class and train the system with constant and adaptive σ^2 for 100 epochs. We apply the learning parameter settings $\alpha_1 = 0.001$, $\alpha_2 = 5 \cdot 10^{-4} \cdot \sigma^2(0)$ and $c = 0.01$. The initial values of the hyperparameter are selected from the interval $[0.05, 0.5]$.

The outcome confirms the observations made on the Letter data set: The initialization $\sigma^2(0)$ has no influence on the final value of the hyperparameter; it converges towards $\sigma^2_{final} \approx 0.3$ in all experiments (see Fig. 5, right). In consequence, the classification performance after training does not depend on $\sigma^2(0)$; the mean final test error is $\varepsilon_{test} = 7.7\%$ for all $\sigma^2(0)$. Learning with constant hyperparameter turns out to be very sensitive with respect to the value of the hyperparameter. However, training with constant $\sigma^2 = \sigma^2_{opt} = 0.15$ slightly outperforms training with adaptive σ^2; on average, the resulting classifiers achieve $\varepsilon_{test} = 7.3\%$ (see Fig. 4,right).

Training of local hyperparameters is continued for 200 epochs. As depicted in Fig. 8, left, the error curves for likelihood ratio based classification converge after ca. 100 epochs. However, the classification performance based

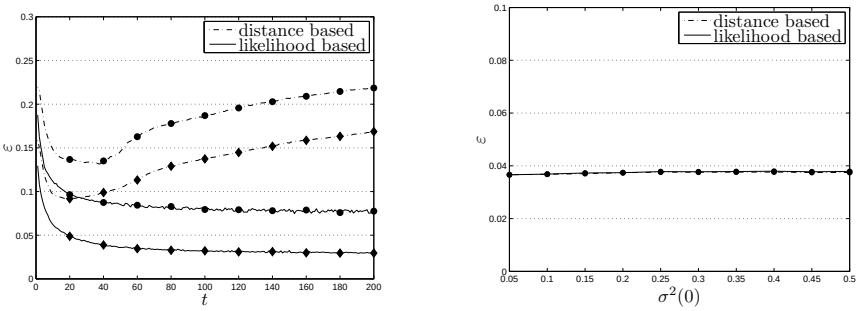

Figure 8: *Pendigits* data set. **Left:** Mean error curves during training of local hyperparameters with one prototype per class and $\sigma_j^2(0) = 0.3$, $\forall j$; the curves are representative for all $\sigma_j^2(0)$. The symbols correspond to training error (⧫), test error (•). **Right:** Mean final test error after training of five prototypes per class and a global hyperparameter σ^2 as a function of the initial value $\sigma^2(0)$.

on the Euclidean distance passes an optimum after ca. 40 epochs and degrades in the further course of training. Note that the curves do not reflect overfitting, since the mean error on the training sets increases simultaneously. After 40 sweeps through the training set, the likelihood ratio based classification performs $\approx 5\%$ better than the distance based approach. The final classification performance based on the likelihood ratio also slightly depends on the initialization, since the σ_j^2 do not exactly approach the same value for different $\sigma_j^2(0)$ (see Fig. 7, right, for an example); in the examined interval of $\sigma_j^2(0)$, we observe final error rates between $\varepsilon_{test} = 7.6\%$ and $\varepsilon_{test} = 7.8\%$.

The decision rule does not influence the classification performance significantly, if we use a global hyperparameter and vary the number of prototypes. RSLVQ training with adaptive global σ^2 and five prototypes per class shows similar performance for both classification strategies (see Fig. 8, right). The hyperparameter converges towards $\sigma_{final}^2 \approx 0.18$ for all $\sigma^2(0)$.

For comparison purposes, we refer to [18, 19]. Here, researches achieved between 1.8% and 2.2% mean test error using different variants of the SVM. Hence, our results are not yet competitive which was not the main objective of this study at the current state. In Sec. 5 we discuss possible approaches for future work to further improve the presented RSLVQ modifications.

5. Conclusion

We presented two modifications of Robust Soft Learning Vector Quantization. We introduced a novel technique to treat the hyperparameter σ^2

and proposed an alternative decision rule different from the standard LVQ-approach of closest prototype classification.

As demonstrated in experiments, the classification accuracy of RSLVQ is highly sensitive with respect to the correct choice of σ^2. However, an optimum choice of the hyperparameter is not possible based on statistical quantities directly from the data since its influence on the underlying discriminative objective function is not clear a priori. We proposed to adapt σ^2 according to the optimization of the likelihood ratio which takes the influence of the hyperparameter on the RSLVQ cost function into account. This approach made the algorithm very robust with respect to the hyperparameter and renders any trial and error search for an appropriate value unnecessary. Hence, the computational effort of a cross validation procedure can be avoided. In general, the model parameters do not exactly converge to the corresponding values of an underlying distribution given by mixtures of Gaussians. This fact can be explained by the influence of the parameters on the region of interest which influences the adaptation, on the one hand, and the discriminative power of the resulting model, on the other hand. As shown in the experiments, a simple automatic optimization provides a very robust scheme which converges to appropriate values almost independently on the initialization. Note that the parameter σ^2 is crucial for the success of the resulting classifier as demonstrated, e.g., in the mathematical investigation of the limit case for $\sigma^2 \to 0$ as provided in [14]. Based on these findings, the question arises whether further metaparameters of the learning scheme can be determined in a similar way. This includes, for example, the labeling of prototypes which has been made adaptive in [20].

Furthermore, we suggested the likelihood ratio of the RSLVQ cost function as a novel criterion for classification of the data. This concept follows naturally from the learning objective of the training procedure. In our experiments, the method turned out to be superior to distance based classification, in particular, if local hyperparameters are optimized for each prototype.

In future work, we will make explicit use of the stochastic formulation of RSLVQ. A generalization of the approach are fuzzy class assignments based on the vector of likelihood ratios $\boldsymbol{L}(\boldsymbol{\xi}) = (L(\boldsymbol{\xi}, 1), \ldots, L(\boldsymbol{\xi}, C))$. Using $\boldsymbol{L}(\boldsymbol{\xi})$ as system output allows to realize fuzzy class assignments for prototype-based classification. Thus, unsafe classification decisions can be realized, which is highly desirable, e.g. in medical applications.

A serious restriction of standard RSLVQ consists in the use of the Euclidean distance measure. In [5], the algorithm was extended with respect to

adaptive distance measures. We are currently combining metric adaptation and hyperparameter adaptation in RSLVQ, showing first promising results.

References

[1] *Bibliography on the Self-Organizing Map (SOM) and Learning Vector Quantization (LVQ)*, Neural Networks Research Centre, Helskinki University of Technology (2002).

[2] T. Kohonen, Self-Organizing Maps, 2nd Edition, Springer, Berlin, Heidelberg, 1997.

[3] T. Bojer, B. Hammer, D. Schunk, K. T. von Toschanowitz, Relevance determination in learning vector quantization, in: M. Verleysen (Ed.), European Symposium on Artificial Neural Networks, 2001, pp. 271–276.

[4] B. Hammer, T. Villmann, Generalized relevance learning vector quantization, Neural Networks 15 (8-9) (2002) 1059–1068.

[5] P. Schneider, M. Biehl, B. Hammer, Distance learning in discriminative vector quantization, Neural Computation 21 (10) (2009) 2942–2969.

[6] A. Sato, K. Yamada, Generalized Learning Vector Quantization, in: M. C. M. D. S. Touretzky, M. E. Hasselmo (Eds.), Advances in Neural Information Processing Systems 8. Proceedings of the 1995 Conference, MIT Press, Cambridge, MA, USA, 1996, pp. 423–9.

[7] S. Seo, K. Obermayer, Soft Learning Vector Quantization, Neural Computation 15 (7) (2003) 1589–1604.

[8] S. Seo, K. Obermayer, Dynamic hyper parameter scaling method for LVQ algorithms, in: International Joint Conference on Neural Networks, Vancouver, Canada, 2006.

[9] J. Hollmén, V. Tresp, O. Simula, Learning vector quantization algorithm for probabilistic models, in: EUSIPCO, X European Signal Processing Conference, Vol. II, 2000, pp. 721–724.

[10] T. C. Kietzmann, S. Lange, M. Riedmiller, Incremental GRLVQ: Learning relevant features for 3d object recognition, Neurocomputing 71 (13-15) (2008) 2868–2879.

[11] M. J. Mendenhall, E. Merényi, Relevance-based feature extraction for hyperspectral images., IEEE Transactions on Neural Networks 19 (4) (2008) 658–672.

[12] T. Villmann, F.-M. Schleif, M. Kostrzewa, A. Walch, B. Hammer, Classification of mass-spectrometric data in clinical proteomics using learning vector quantization methods, Briefings in Bioinformatics 9 (2) (2008) 129–143.

[13] K. Crammer, R. Gilad-Bachrach, A. Navot, A. Tishby, Margin analysis of the lvq algorithm, in: Advances in Neural Information Processing Systems, Vol. 15, MIT Press, Cambridge, MA, 2003, pp. 462–469.

[14] M. Biehl, A. Ghosh, B. Hammer, Dynamics and generalization ability of LVQ algorithms, Journal of Machine Learning Research 8 (2007) 323–360.

[15] B. Hammer, M. Strickert, T. Villmann, On the generalization ability of GRLVQ networks, Neural Processing Letters 21 (2) (2005) 109–120.

[16] D. J. Newman, S. Hettich, C. L. Blake, C. J. Merz, UCI repository of machine learning databases, `http://archive.ics.uci.edu/ml/` (1998).

[17] R. Duda, P. Hart, D. Stork, Pattern Classification, 2nd Edition, Wiley-Interscience, 2000.

[18] I. W. Tsang, A. Kocsor, J. T. Kwok, Diversified svm ensembles for large data sets, in: ECML, Vol. 4212 of Lecture Notes in Computer Science, Springer, 2006, pp. 792–800.

[19] R. Perfetti, E. Ricci, Reduced complexity rbf classifiers with support vector centres and dynamic decay adjustment, Neurocomputing 69 (16-18) (2006) 2446–2450.

[20] T. Villmann, B. Hammer, F.-M. Schleif, T. Geweniger, W. Herrmann, Fuzzy classification by fuzzy labeled neural gas, Neural Networks 19 (2006) 772–779.