Advanced methods for prototype-based classification

Schneider, Petra

Material based on:

Petra Schneider, Michael Biehl and Barbara Hammer - *"Hyperparameter Learning in Probabilistic Prototype-based models,"* Neurocomputing, vol. 73 , no. 7-9, 2010.

Petra Schneider, Tina Geweniger, Frank-Michael Schleif, Michael Biehl and Thomas Villmann - *"Generalizing Robust Soft LVQ to handle data with uncertain class labels,"* submitted, 2010.

Chapter 6

Hyperparameter learning in probabilistic prototype-based models

Abstract

We present 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 classifier's 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. Moreover, we generalize RSLVQ with respect to the treatment of vectorial class labels in the training data. This allows to integrate uncertain class information of training data into the learning process of an RSLVQ classifier.

6.1 Introduction

The learning rules of several LVQ procedures involve a hyperparameter, such as the window size in LVQ2.1 (Kohonen, 1997) or the softness parameter σ^2 in Soft LVQ (Seo and Obermayer, 2003) and Robust Soft LVQ (Seo and Obermayer, 2003). 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 Seo and Obermayer (2006), 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 work focuses on RSLVQ (see Sec. 2.3.3) 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. In this study, 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.

Finally, we generalize the RSLVQ cost function with respect to vectorial class labels for training data. In particular, we suppose that each element of the label vectors is in the range $[0, 1]$ describing the fuzzy assignment of the data to the respective class. This approach allows to consider insecure label information for the construction of a nearest prototype classifier.

In our experiments, we illustrate 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. Based on artificial data, we demonstrate that the extended cost function yields a generalization of the original RSLVQ algorithm.

6.2 Modifications of Robust Soft LVQ

The derivation of the update rules of the algorithms in Sec.s 6.2.1 and 6.2.3 are based on the assumption of a Gaussian mixture model. Hence, the densities $p(\boldsymbol{\xi}|j)$ in Eq. (2.8) have the normalized exponential form (see Eq. (2.9)) with

$$K(j) = \frac{1}{(2\pi\sigma_j^2)^{n/2}}, \qquad f(\boldsymbol{\xi}, \boldsymbol{w}_j, \sigma_j^2) = -\frac{(\boldsymbol{\xi} - \boldsymbol{w}_j)^T(\boldsymbol{\xi} - \boldsymbol{w}_j)}{2\,\sigma_j^2}. \tag{6.1}$$

The derivatives constitute

$$\frac{\partial f(\boldsymbol{\xi}, \boldsymbol{w}_j, \sigma_j^2)}{\partial \boldsymbol{w}_j} = \frac{1}{\sigma^2}(\boldsymbol{\xi} - \boldsymbol{w}_j), \tag{6.2}$$

$$\frac{\partial f(\boldsymbol{\xi}, \boldsymbol{w}_j, \sigma_j^2)}{\partial \sigma_j^2} = \frac{(\boldsymbol{\xi} - \boldsymbol{w}_j)^T (\boldsymbol{\xi} - \boldsymbol{w}_j)}{2\,\sigma_j^4}, \tag{6.3}$$

$$\frac{\partial K(j)}{\partial \sigma_j^2} = -\frac{n}{2}\,\frac{1}{(2\pi\sigma_j^2)^{n/2}\,\sigma_j^2}. \tag{6.4}$$

6.2.1 Hyperparameter adaptation in RSLVQ

In Seo and Obermayer (2006), 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. (2.12). We adapt the hyperparameter according to the gradient of E_{RSLVQ} with respect to σ^2. In case of a Gaussian mixture model, the derivatives in Eq. (3.27) and Eq. (6.3) lead us to the update rule

$$\Delta\sigma^2 = \alpha_2 \sum_j \Bigg(\Big(\delta_{y,c(\boldsymbol{w}_j)}\,(P_y(j|\boldsymbol{\xi}) - P(j|\boldsymbol{\xi})) - (1 - \delta_{y,c(\boldsymbol{w}_j)})\,P(j|\boldsymbol{\xi}) \Big) \cdot \frac{d(\boldsymbol{\xi}, \boldsymbol{w}_j)}{\sigma^4} \Bigg). \tag{6.5}$$

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 Sec. 3.A.2 and Eq.s (6.3), (6.4) we obtain the learning rule

$$\Delta\sigma_j^2 = \frac{\alpha_2}{\sigma_j^2} \cdot \begin{cases} (P_y(j|\boldsymbol{\xi}) - P(j|\boldsymbol{\xi}))\,(-n + d(\boldsymbol{\xi}, \boldsymbol{w}_j)/\sigma_j^2), & c(\boldsymbol{w}_j) = y, \\ -P(j|\boldsymbol{\xi})\,(-n + d(\boldsymbol{\xi}, \boldsymbol{w}_j)/\sigma_j^2), & c(\boldsymbol{w}_j) \neq y. \end{cases} \tag{6.6}$$

Using this approach, the update rules for the prototypes (see Seo and Obermayer, 2003) 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 from 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 (Kohonen, 1997), automatic adaptation provides a principled way to optimize these parameters.

6.2.2 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. (2.11). The closest prototype scheme as described in Sec. 2.2 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. 6.2.1.

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

6.2.3 Generalized cost function

In the following, we provide a generalization of the RSLVQ cost function with respect to vectorial input data. The objective of this approach is take into account insecure label information of training data lying close to the decision boundary. We derive the learning rules for the prototypes and the hyperparameter σ^2 as a stochastic gradient of the extended cost function. We term the novel algorithm Fuzzy Robust Soft LVQ. In the limit of crisp class memberships, the cost function and the updates are equivalent to the original RSLVQ algorithm. Note that the resulting classifier performs a crisp classification although fuzzy labeled data is used for the training process.

Every training pattern $\boldsymbol{\xi}$ carries a probabilistic label vector $\boldsymbol{y} \in [0, 1]^C$, with $\sum_i y^i = 1$. Component y^i constitutes the assignment probability of sample $\boldsymbol{\xi}$ to class i. Since the classifier performs a crisp classification, the prototypes still represent exactly one class. Under this assumption, the definition of $\mathbf{W}$ (see Eq. (2.1))) remains unchanged compared to the crisp version of the algorithm. The same holds for the

definition of the probability density $p(\boldsymbol{\xi}|\mathbf{W})$ in Eq. (2.8). Since $p(\boldsymbol{\xi}|\mathbf{W})$ only takes the prototypes' class memberships into account, the density remains unchanged due to the fuzzy labeling of the data. However, the class specific density in Eq. (2.10) needs to be generalized in order to handle data with vectorial class labels. This generalization constitutes the weighted sum of the class-specific densities; the weight values are given by the assignment probabilities to the different classes

$$p(\boldsymbol{\xi}, \boldsymbol{y}|\mathbf{W}) = \sum_{i=1}^{C} y^i \sum_{j:c(\boldsymbol{w}_j)=i}^{m} p(\boldsymbol{\xi}|j)P(j). \tag{6.7}$$

The cost function of Fuzzy RSLVQ arises out of E_{RSLVQ} by defining the likelihood ratio in Eq. (2.11) in terms of $p(\boldsymbol{\xi}, \boldsymbol{y}|\mathbf{W})$.

In accordance with original RSLVQ (Seo and Obermayer, 2003), we implement the optimization of E_{FRSLVQ} in terms of a stochastic gradient ascent. The derivative of the novel cost function with respect to the model parameters is provided in Sec. 6.A. Combining Eq.s (6.2) - (6.4) and Eq. (6.11) yields the update rules for the prototypes and the hyperparameters

$$\Delta \boldsymbol{w}_j = \frac{\alpha_1}{\sigma_j^2} \left(P_{\boldsymbol{y}}(j|\boldsymbol{\xi}) - P(j|\boldsymbol{\xi})\right) (\boldsymbol{\xi} - \boldsymbol{w}_j), \tag{6.8}$$

$$\Delta \sigma_j^2 = \alpha_2 \left(P_{\boldsymbol{y}}(j|\boldsymbol{\xi}) - P(j|\boldsymbol{\xi})\right) \left(\frac{d(\boldsymbol{\xi}, \boldsymbol{w}_j)}{\sigma_j^4} - \frac{n}{\sigma_j^2}\right), \tag{6.9}$$

where $\alpha_{1,2}$ are the learning rates. For the more general case of a global parameter $\sigma_j^2 = \sigma^2, \forall j$, the hyperparameter can be updated by the summation of the probability assignments

$$\Delta \sigma^2 = \alpha_2 \sum_{j=1}^{m} \left(P_{\boldsymbol{y}}(j|\boldsymbol{\xi}) - P(j|\boldsymbol{\xi})\right) \cdot \frac{d(\boldsymbol{\xi}, \boldsymbol{w}_j)}{\sigma^4}. \tag{6.10}$$

In the experimental section, we focus on the adaptation of a global hyperparameter σ^2 only. The Fuzzy RSLVQ algorithm is defined in terms of Eq.s (6.8) - (6.10).

Note that $(P_{\boldsymbol{y}}(j|\boldsymbol{\xi}) - P(j|\boldsymbol{\xi}))$ turns into the weight factors in the update rules of algorithms previously derived from RSLVQ, if $\boldsymbol{y}$ specifies a crisp class labeling of sample $\boldsymbol{\xi}$ (e.g. Eq.s (3.18), (3.19), (6.6)). The factor is positive for model parameters related to class j with $j = \arg\max_i (y^i)$ and negative otherwise. This is also illustrated in Fig. 6.1 by means of a one-dimensional example setting consisting of two prototypes. The plots depict the update strength $(P_{\boldsymbol{y}}(j|\boldsymbol{\xi}) - P(j|\boldsymbol{\xi}))$ caused by data samples of different fuzziness. Furthermore, the influence of the hyperparameter σ^2 is illustrated. The figures depict that $\int (P_{\boldsymbol{y}}(j|\boldsymbol{\xi}) - P(j|\boldsymbol{\xi}))\, d\boldsymbol{\xi}$ is small for highly

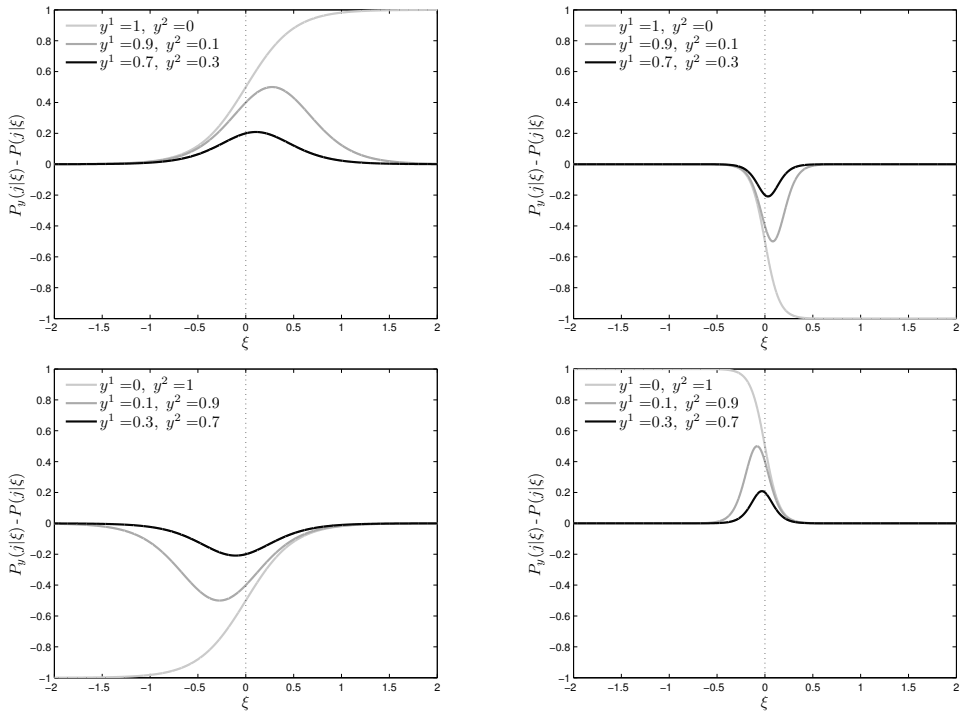

Figure 6.1: *Illustration of the weight factors* $(P_{\boldsymbol{y}}(j|\boldsymbol{\xi}) - P(j|\boldsymbol{\xi}))$ *affecting the prototypes* $\boldsymbol{w}_{1,2}$ *in a one-dimensional setting with two prototypes* $\boldsymbol{w}_1 = -1$ *and* $\boldsymbol{w}_2 = 1$*. In each figure, the curves correspond to training samples* $\boldsymbol{\xi}$ *with different fuzzy label* $\boldsymbol{y}$*.* **Left column***: Forces affecting* $\boldsymbol{w}_1$ *with* $\sigma^2 = 0.5$*. Top:* $y^1 > 0.5$ *(attractive forces). Bottom:* $y^1 < 0.5$ *(repulsive forces).* **Right column***: Forces affecting* $\boldsymbol{w}_2$ *with* $\sigma^2 = 0.15$*. Top:* $y^1 > 0.5$ *(repulsive forces). Bottom:* $y^1 < 0.5$ *(attractive forces).*

fuzzy data, but increases with decreasing fuzziness. In the limit of crisp class labeling, the integral tends to infinity. The hyperparameter determines the width of the active region. For small σ^2, the active region is very sharp and focused on the region around the decision boundary. Training samples from a larger area in feature space contribute to the training, if σ^2 is set to larger values.

6.3 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

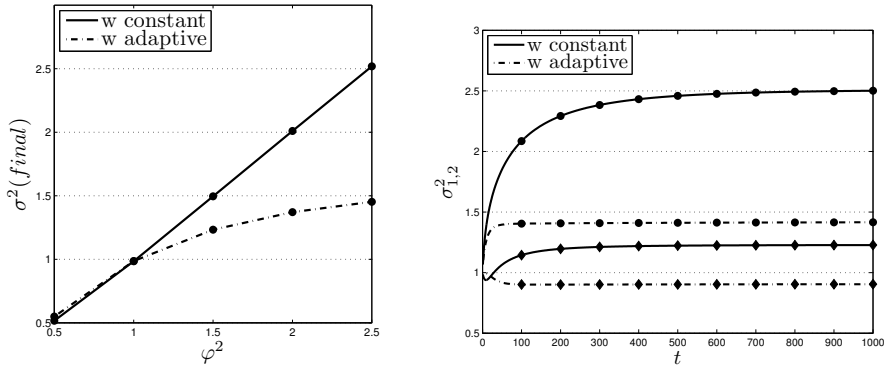

Figure 6.2: *Artificial data.* **Left:** *Variance φ^2 of the Gaussians vs. mean final value of the global hyperparameters σ^2 obtained on data sets with two clusters of equal variance and constant and adaptive prototypes.* **Right:** *Evolution of the local hyperparameters $\sigma^2_{1,2}$ as a function 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 (♦).*

space and correspond to binary classification problems. We investigate the relation between the hyperparameter σ^2 and the variance of the Gaussians for crisp and fuzzy labeled data. Furthermore, we highlight differences between the alternative decision rules based on the Euclidean distance and the likelihood ratio.

In order to evaluate the performance of the new techniques in real life situations, the methods are applied to two benchmark data sets from the UCI repository of machine learning (Newman et al., 1998).

In all experiments, the learning rates are continuously reduced with training time according to the schedule in Eq. (3.21). To initialize the prototypes, we choose the mean values of random subsets of data points selected from each class.

6.3.1 Artificial data

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 independent of other model parameters. In the next experiments, the hyperparameter and the prototypes are optimized simultaneously. The learning

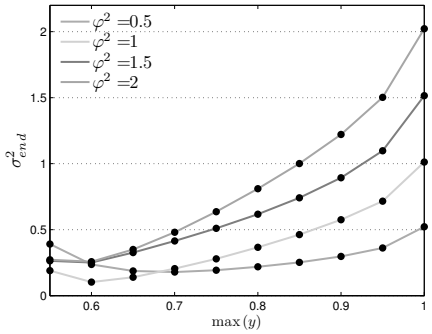

Figure 6.3: *Artificial data. Mean final value of the hyperparameter as a function of the fuzziness y_1^1 of the training data obtained on data sets with different variance φ^2.*

parameters are set to $\alpha_1 = 0.01$, $\alpha_2 = 0.001$, $c = 0.001$ and the system is trained for 1 000 epochs. The softness is initialized by $\sigma^2(0) = 1$. The results presented in the following are averaged over experiments on ten statistically independent data sets.

Fig. 6.2 (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. 6.2, right, for an example.

Hence, maximizing the likelihood ratio in Eq. (2.11) 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.

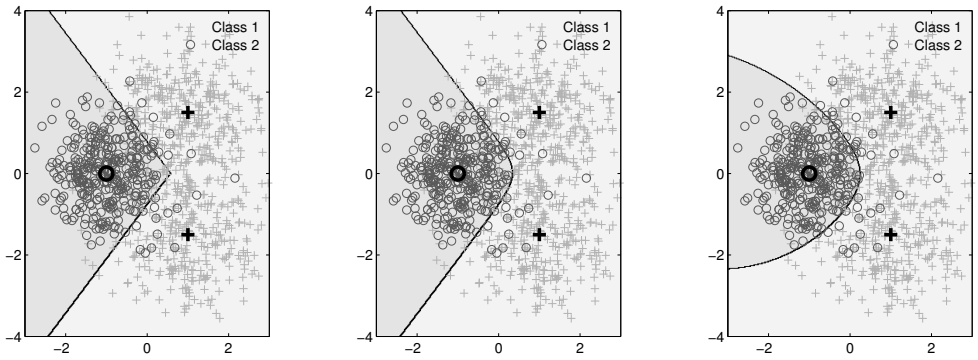

Figure 6.4: *Receptive fields induced by distance based classification and likelihood ratio based classification for a two-class problem consisting of three clusters. The cluster means serve as prototypes.* **Left:** *Distance based classification using the squared Euclidean distance.* **Middle:** *Likelihood ratio based classification after training of a global hyperparameter* σ^2. **Right:** *Likelihood ratio based classification after training of local hyperparameters* $\sigma^2_{1,2,3}$.

Furthermore, we perform equivalent experiments in a generalized framework to demonstrate that the previous observations are results of Fuzzy RSLVQ under special conditions. To generalize the previous settings, all data generated by one distribution need to carry identical label vectors $\boldsymbol{y}_1$ or $\boldsymbol{y}_2$. Moreover, the conditions $y_1^1 = y_2^2$ and $y_1^2 = y_2^1$ need to be fulfilled. In the following, we always refer to y_1^1 to specify the fuzziness of the data.

Following the same procedure, we first set the prototypes fixed in the cluster centers to analyse the adaptation of σ^2 exclusively. Afterwards, we train the prototypes with constant $\sigma^2 = 1$. We analyse, how the fuzziness of the input data influences the resulting model. We choose the learning parameter settings $\alpha_1 = 1 \cdot 10^{-4}$, $\alpha_2 = 1 \cdot 10^{-5}$ and $c = 0.001$. The hyperparameter initialized by $\sigma^2(0) = 1$.

Our observations are depicted in Fig. 6.3. According to the previous experiments, σ^2 approaches φ^2 in case of the crisp labeling $y_1^1 = 1$. Remarkably, with increasing fuzziness, σ^2 initially approaches smaller values. Note however that the curves pass a minimum. In the experiments with highly fuzzy data (y_1^1 close to 0.5), the final value $\sigma^2(t)$ increases again. The value y_1^1 which results in the minimal value σ^2 depends on φ^2. With $y_1^1 < 0.5$, σ^2 converges to very large values > 10.

If we adapt the prototypes with constant hyperparameter $\sigma^2 = 1$, $\boldsymbol{w}_{1,2}$ do not saturate in the cluster centers. The fuzziness determines the prototypes' distance to the decision boundary. In case of $y_1^1 = 1$, incorrectly classified data contribute much stronger to the value of the cost function compared to correctly classified samples.

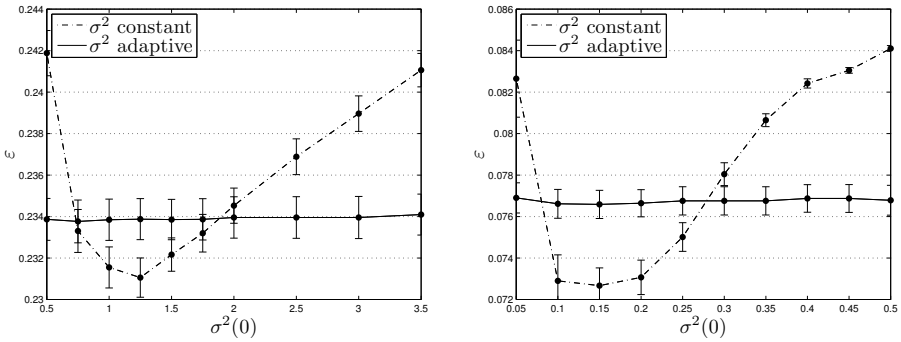

Figure 6.5: *Mean test performance at the end of RSLVQ-training with constant and adaptive hyperparameter as a function of the initial value $\sigma^2(0)$. The error bars indicate the standard error.* **Left:** Letter *data set* **Right:** Pendigits *data set.*

For this reason, the attractive forces caused by misclassified patterns on the opposite side of the decision boundary dominate the update. For this reason, the prototypes move in the direction of the decision boundary. Increasing fuzziness weakens this effect and the distance between the prototypes increases.

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. 6.4. 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 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 $\alpha_2 = 1 \cdot 10^{-5}$, $c = 1 \cdot 10^{-4}$, $\sigma^2(0) = \sigma_j^2(0) = 0.1, \forall j$ 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 $\sigma_{1,2,3}^2$ approach $\varphi_{1,2,3}^2$. Fig. 6.4 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, if local softness parameters are assigned to the prototypes as displayed in the right-most panel of Fig. 6.4.

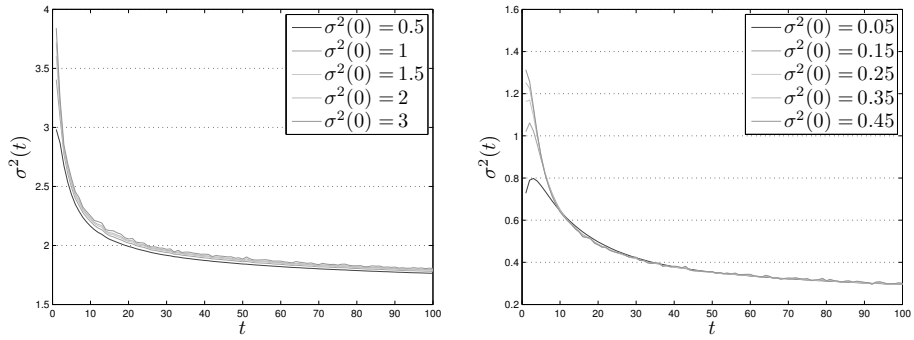

Figure 6.6: *Evolution of the global hyperparameter σ^2 as a function of training time for different initial settings $\sigma^2(0)$.* **Left:** Letter *data set.* **Right:** Pendigits *data set.*

6.3.2 Real life data

In the second set of experiments, we apply the algorithms to the Letter Recognition- and Pendigits data set provided by the UCI-Repository of Machine Learning (Newman et al., 1998).

Letter Recognition

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 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. 6.5, 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. 6.5 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.

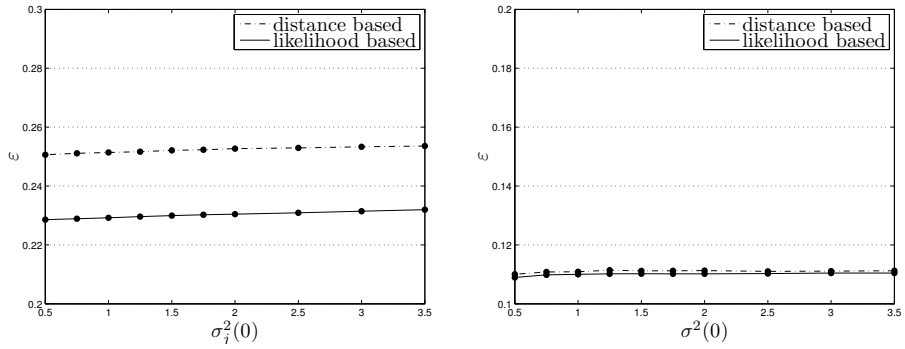

Figure 6.7: Letter *data set. Mean final test errors after training of global and local hyperparameters as a function of the initial values $\sigma^2(0)$ and $\sigma_j^2(0)$. Standard error bars would be smaller than the symbol size. The plots compare the performance yielded by the alternative decision 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.*

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. 6.5, left). Furthermore, the initialization $\sigma^2(0)$ does not influence the final value of the hyperparameter. As depicted in Fig. 6.6, 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 $\sigma^2 = \sigma^2_{opt}$ 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. 6.7, left, highest likelihood ratio classification achieves $\approx 2\%$ lower mean

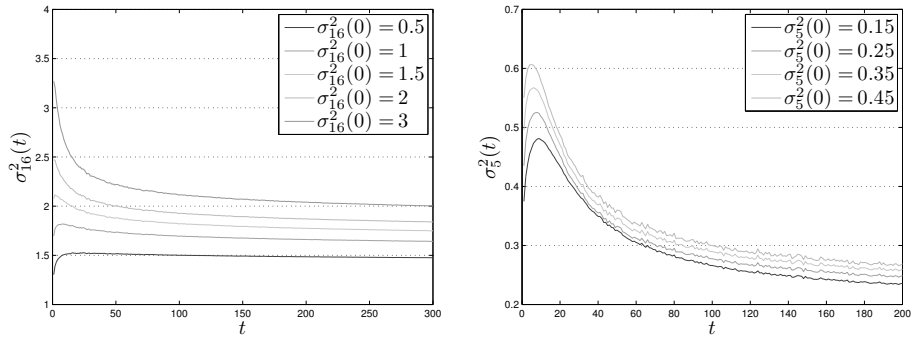

Figure 6.8: *Evolution of local hyperparameters σ_j^2 as function of training time for different initial settings $\sigma_j^2(0)$.* **Left:** Letter *data set, $j = 16$.* **Right:** Pendigits *data set, $j = 5$.*

error rate on the test samples. However, a slight dependence of the error rate on the initialization $\sigma_j^2(0)$ can be observed. In contrast to our experiments with global σ^2, the final local hyperparameters spread more for different settings $\sigma_j^2(0)$. This is visible in Fig. 6.8, left, which compares the evolution of $\sigma_{16}^2(t)$ for different initializations; the curves are representative for all σ_j^2. 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 $\sigma_{final}^2 \approx 0.8$, independent of $\sigma^2(0)$. Although the classification improves significantly due to the larger number 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. 6.7, 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 vary between 15.5%

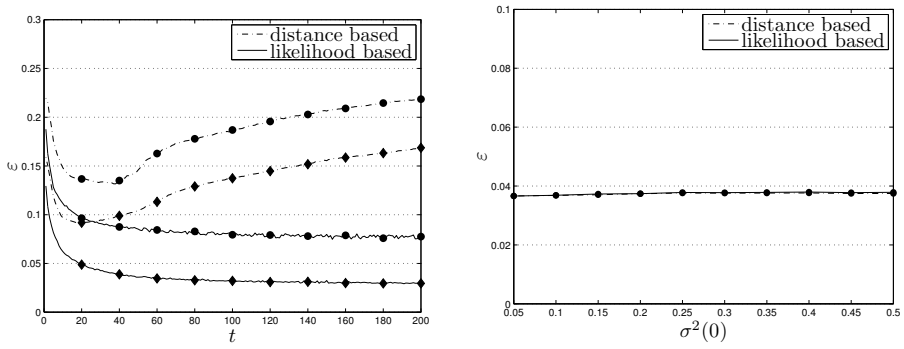

Figure 6.9: Pendigits *data set.* **Left:** *Mean error curves during training of local hyperparameters with one prototype per class and* $\sigma_j^2(0) = 0.3, \forall j$*; the curves are representative for all* $\sigma_j^2(0)$*. The symbols correspond to training error* (⧫)*, test error* (•)*.* **Right:** *Mean final test error after training of five prototypes per class and a global hyperparameter* σ^2 *as a function of the initial value* $\sigma^2(0)$*. Standard error bars would be smaller than the symbol size.*

and 8.5% mean error on the test sets depending on the kernel-function [1]. 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 are not examined in this study.

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 $[0.05, 0.5]$.

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

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. 6.6, 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. 6.5, right).

Training of local hyperparameters is continued for 200 epochs. As depicted in Fig. 6.9, left, the error curves for likelihood ratio based classification converge after ca. 100 epochs. However, the classification performance based 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^2_j do not exactly approach the same value for different $\sigma^2_j(0)$ (see Fig. 6.8, right, for an example); in the examined interval of $\sigma^2_j(0)$, 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. 6.9, right). The hyperparameter converges towards $\sigma^2_{final} \approx 0.18$ for all $\sigma^2(0)$.

For comparison purposes, we refer to Tsang et al. (2006); Perfetti and Ricci (2006). 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. 6.4, we discuss possible approaches for future work to further improve the presented RSLVQ modifications.

6.4 Conclusion

We presented 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. Furthermore, we extended the RSLVQ cost function with respect to vectorial input data.

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 Biehl, Ghosh and Hammer (2007).

The proposed generalization of the RSLVQ cost function allows to account for insecure label information of training data in overlapping regions of different classes. Hence, uncertainty concerning class memberships can be integrated in the learning process of a nearest prototype classifier. The update scheme for the new algorithm is derived as a stochastic gradient of the generalized cost function and includes the RSLVQ updates as a special case. It was demonstrated based on artificial data that previous RSLVQ results arise out of Fuzzy RSLVQ under certain conditions.

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, Fuzzy RSLVQ needs to be applied to more complex data sets. Real life applications will have to show to what extent the additional consideration of insecure label information influences the classification performance of a crisp classifier. Moreover, the question arises whether further generalization with respect to vectorial, adaptive prototype labels can be determined in a similar way. Vectorial system output allows to realize fuzzy class assignments. 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 Sec. 3.3.3, the algorithm is extended with respect to adaptive

distance measures. We are currently combining metric adaptation in RSLVQ with the different approaches presented in this chapter, showing first promising results.

6.A Appendix: Derivatives of E_{FRSLVQ}

We compute the derivative of the cost function with respect to a general parameter $\Theta_i \neq \boldsymbol{\xi}$. We assume the densities $p(\boldsymbol{\xi}|j)$ to have the normalized exponential form $p(\boldsymbol{\xi}|j) = K(j) \cdot \exp f(\boldsymbol{\xi}, \boldsymbol{w}_j, \sigma_j^2, \boldsymbol{\lambda}_j)$, where $\boldsymbol{\lambda}_j$ is the vector of metric parameters.

$$
\begin{aligned}
&\frac{\partial \log \frac{p(\boldsymbol{\xi},\boldsymbol{y}|\mathbf{W})}{p(\boldsymbol{\xi}|\mathbf{W})}}{\partial \Theta_i} \\
&\quad = \frac{\partial \log p(\boldsymbol{\xi},\boldsymbol{y}|\mathbf{W})}{\partial \Theta_i} - \frac{\partial \log p(\boldsymbol{\xi}|\mathbf{W})}{\partial \Theta_i} \\
&\quad = \frac{1}{p(\boldsymbol{\xi},\boldsymbol{y}|\mathbf{W})} \underbrace{\frac{\partial p(\boldsymbol{\xi},\boldsymbol{y}|\mathbf{W})}{\partial \Theta_i}}_{(a)} - \frac{1}{p(\boldsymbol{\xi}|\mathbf{W})} \underbrace{\frac{\partial p(\boldsymbol{\xi}|\mathbf{W})}{\partial \Theta_i}}_{(b)} \\
&\quad = \frac{y^{c(\boldsymbol{w}_i)} P(i) \exp f(\boldsymbol{\xi}, \boldsymbol{w}_i, \sigma_i^2, \boldsymbol{\lambda}_i)}{p(\boldsymbol{\xi},\boldsymbol{y}|\mathbf{W})} \frac{\partial K(i)}{\partial \Theta_i} \\
&\qquad + \frac{y^{c(\boldsymbol{w}_i)} P(i) K(i) \exp f(\boldsymbol{\xi}, \boldsymbol{w}_i, \sigma_i^2, \boldsymbol{\lambda}_i)}{p(\boldsymbol{\xi},\boldsymbol{y}|\mathbf{W})} \frac{\partial f(\boldsymbol{\xi}, \boldsymbol{w}_i, \sigma_i^2, \boldsymbol{\lambda}_i)}{\partial \Theta_i} \\
&\qquad - \frac{y^{c(\boldsymbol{w}_i)} P(i) \exp f(\boldsymbol{\xi}, \boldsymbol{w}_i, \sigma_i^2, \boldsymbol{\lambda}_i)}{p(\boldsymbol{\xi}|\mathbf{W})} \frac{\partial K(i)}{\partial \Theta_i} \\
&\qquad - \frac{y^{c(\boldsymbol{w}_i)} P(i) K(i) \exp f(\boldsymbol{\xi}, \boldsymbol{w}_i, \sigma_i^2, \boldsymbol{\lambda}_i)}{p(\boldsymbol{\xi}|\mathbf{W})} \frac{\partial f(\boldsymbol{\xi}, \boldsymbol{w}_i, \sigma_i^2, \boldsymbol{\lambda}_i)}{\partial \Theta_i} \\
&\quad = \left(P_{\boldsymbol{y}}(i|\boldsymbol{\xi}) - P(i|\boldsymbol{\xi})\right) \left(\frac{1}{K(i)} \frac{\partial K(i)}{\partial \Theta_i} + \frac{\partial f(\boldsymbol{\xi}, \boldsymbol{w}_i, \sigma_i^2, \boldsymbol{\lambda}_i)}{\partial \Theta_i} \right)
\end{aligned}
\tag{6.11}
$$

with (a)

$$\begin{aligned}
&\frac{\partial p(\boldsymbol{\xi}, \boldsymbol{y}|\mathbf{W})}{\partial \Theta_i} \\
&\quad = \frac{\partial}{\partial \Theta_i} \left(\sum_{k=1}^{C} y^k \sum_{j=1}^{m} \delta_{k,c(\boldsymbol{w}_j)} P(j) p(\boldsymbol{\xi}|j) \right) \\
&\quad = \sum_{k=1}^{C} y^k \sum_{j=1}^{m} \delta_{k,c(\boldsymbol{w}_j)} P(j) \frac{\partial p(\boldsymbol{\xi}|j)}{\partial \Theta_i} \\
&\quad = \sum_{k=1}^{C} y^k \sum_{j=1}^{m} \delta_{k,c(\boldsymbol{w}_j)} P(j) \exp f(\boldsymbol{\xi}, \boldsymbol{w}_j, \sigma_j^2, \boldsymbol{\lambda}_j) \\
&\qquad\qquad \times \left(\frac{\partial K(j)}{\partial \Theta_i} + K(j) \frac{\partial f(\boldsymbol{\xi}, \boldsymbol{w}_j, \sigma_j^2, \boldsymbol{\lambda}_j)}{\partial \Theta_i} \right) \\
&\quad = y^{c(\boldsymbol{w}_i)} P(i) \exp f(\boldsymbol{\xi}, \boldsymbol{w}_i, \sigma_i^2, \boldsymbol{\lambda}_i) \left(\frac{\partial K(i)}{\partial \Theta_i} + K(i) \frac{\partial f(\boldsymbol{\xi}, \boldsymbol{w}_i, \sigma_i^2, \boldsymbol{\lambda}_i)}{\partial \Theta_i} \right)
\end{aligned}$$

and (b) given in Sec. 3.A.2.

$P_{\boldsymbol{y}}(i|\boldsymbol{\xi})$ is the assignment probability to component i within class $c(\boldsymbol{w}_i)$, taking the probability into account that $\boldsymbol{\xi}$ was generated by one of the components of class $c(\boldsymbol{w}_i)$

$$\begin{aligned}
P_{\boldsymbol{y}}(i|\boldsymbol{\xi}) &= \frac{\sum_k y^k \delta_{k,c(\boldsymbol{w}_i)} P(i) K(i) \exp f(\boldsymbol{\xi}, \boldsymbol{w}_i, \sigma_i^2, \boldsymbol{\lambda}_i)}{\sum_k y^k \sum_j \delta_{k,c(\boldsymbol{w}_j)} P(j) K(j) \exp f(\boldsymbol{\xi}, \boldsymbol{w}_j, \sigma_j^2, \boldsymbol{\lambda}_j)} \\
&= \frac{y^{c(\boldsymbol{w}_i)} P(i) K(i) \exp f(\boldsymbol{\xi}, \boldsymbol{w}_i, \sigma_i^2, \boldsymbol{\lambda}_i)}{\sum_k y^k \sum_j \delta_{k,c(\boldsymbol{w}_j)} P(j) K(j) \exp f(\boldsymbol{\xi}, \boldsymbol{w}_j, \sigma_j^2, \boldsymbol{\lambda}_j)}.
\end{aligned}$$

$P(i|\boldsymbol{\xi})$ depicts the probability the $\boldsymbol{\xi}$ is assigned to any component i of the mixture (see Sec. 3.A.2).

The derivative with respect to a global parameter, e.g. a global hyperparameter $\sigma^2 = \sigma_j^2$ for all j can be derived thereof by summation.

Chapter 7

Conclusion

7.1 Summary

The thesis presents different concepts to improve the performance of LVQ algorithms. One issue refers to metric learning in LVQ, i.e., the optimization of the employed distance measure for a specific application. A novel parameterized distance measure is proposed which can be adapted to the training data by means of any LVQ algorithm. Furthermore, three approaches to modify Robust Soft LVQ are introduced.

After providing the required background to Learning Vector Quantization in chapter 2, the novel distance measure is introduced in chapter 3. The Euclidean distance is extended by a full matrix of adaptive weight values. This approach generalizes the popular concept of relevance learning. The diagonal elements of the adaptive matrix correspond to an explicit feature weighting. The off-diagonal elements additionally weight combinations of features. Hence, correlations can be taken into account to evaluate the similarity between prototypes and feature vectors. Equivalently, this metric corresponds to the squared Euclidean distance after performing a linear transformation of data and prototypes to a new feature space. Full rank matrices can be learned as well as distance measures which are restricted to a smaller number of features. Moreover, global, as well as class-wise or prototype-specific metrics can be implemented. We derive the learning rules for matrix learning in Generalized LVQ and Robust Soft LVQ. Experiments on artificial and benchmark real life data sets illustrated the new technique in practice. The applications demonstrate that matrix learning is advantageous in two respects: The approach is clearly beneficial with respect to classification accuracy compared to the use of the Euclidean distance or the weighted Euclidean distance. This holds especially for local matrix adaptation in case of multi-class problems. Furthermore, the additional parameters improve the interpretability of the final model. Matrix parameters reveal the importance of the input dimensions for the diagonal elements and the importance of correlations for the off-diagonal elements. Additionally, the experiments

show that the two training algorithms display different learning dynamics and different characteristics of the resulting model.

In chapter 4, we present a regularization technique to influence the convergence behavior of matrix learning. The proposed method prevents the training algorithm from yielding low rank distance measures. As it is frequently observed, the update rules tend to select a single or very few directions in feature space. We propose a regularization term which favors balanced eigenvalue profiles of the relevance matrix. The technique can be applied in combination with any matrix learning scheme. We demonstrate the usefulness by means of matrix learning in GLVQ. Beside the desired effect on the convergence behavior, the technique turns out to be beneficial to prevent over-fitting effects and numerical instabilities during training.

Theoretical aspects of matrix learning in LVQ are investigated in chapter 5. In particular, we research the convergence behaviour in terms local matrix adaptation in LVQ1. Under simplifying model assumptions, it is shown that the updates lead to the selection of single directions in feature space. This direction is determined by the statistical properties of the data assigned to the respective prototype. Furthermore, we investigate the effect of the proposed regularization approach in this framework. We show that the method eases the tendency of the learning algorithm to strongly reduce the number of dimensions. The derivations are verified by practical experiments.

Chapter 6 presents three approaches to extend Robust Soft LVQ. The study refers to the original version of the algorithm based on the squared Euclidean distance. First, we propose to adapt the algorithm's hyperparameter in the course of training based on the gradient information of the RSLVQ cost function. Our experiments depict that hyperparameter learning makes the algorithm very robust with respect to the initial choice of the hyperparameter. Moreover, local hyperparameters can be learned for every prototype individually. Furthermore, we present an alternative decision rule for RSLVQ classifiers: Training the model parameters maximizes the underlying cost function which is defined in terms of a likelihood ratio. Against this background, we propose to assign a sample in the working phase to the class of highest likelihood ratio. Contrary to the distance based classification rule, this approach naturally follows the objective of RSLVQ training. The following practical examples show that the novel decision rule outperforms the distance based approach, if the method is combined with local, adaptive hyperparameters. Finally, we derive the generalization of the RSLVQ cost function with respect to vectorial class labels of the input data. We demonstrate on artificial data set that RSLVQ is a special case of the novel fuzzy algorithm.

7.2 Future work

This work could be extended in several directions. In particular, we consider the following topics for future research:

- Throughout the thesis, we apply the novel algorithms to benchmark data sets. Future work will address the application to more sophisticated data sets. Due to the interpretability and the easy handling of missing values, LVQ is especially attractive for interdisciplinary applications in the medical or biological domain. Relevance learning has already proven to be valuable for the analysis of such data, e.g. for biomarker detection. Matrix learning has the potential to further advance this work. Currently, we are working on a diagnosis system for adrenal cancer using the algorithms presented in chapter 3. The experiments show promising results which will be presented in a forthcoming publication.

- Future work will also combine the proposed extensions of Robust Soft LVQ (Sec. 6.2) with matrix learning in RSLVQ (Sec. 3.3.3)

- Several learning problems introduced in this thesis are formulated in terms of the optimization of a cost function, and the optimization is implemented in terms a stochastic gradient descent or ascent. Alternatively, advanced optimization algorithms could be used to train the model parameters, e.g. line search methods or the conjugate gradient algorithm (Bishop, 1995). Although these methods are computationally more expensive, the approach could eliminate the sensitivity of the learning process with respect to initialization and learning parameter. Advanced optimization techniques could make to learning dynamics more robust and reduce the required number of learning steps.

- Finally, continuing the work presented in Sec. 6.2.3, we plan to further generalize the RSLVQ cost function with respect to vectorial, adaptive class labels for the prototypes. This natural extension of Fuzzy RSLVQ allows to realize insecure classification results, which is highly desirable, e.g. in medical applications.

Samenvatting

Dit proefschrift beschrijft verschillende concepten voor het verbeteren van LVQ algoritmen. Een van de behandelde punten is het aanleren van een metriek binnen LVQ, dat wil zeggen, het optimaliseren van de gebruikte afstandsmaat voor een specifieke toepassing. Een nieuwe geparametriseerde afstandsmaat wordt voorgesteld, welke door ieder LVQ-algoritme aangepast kan worden aan de traindata. Ook zullen drie verschillende benaderingen om Robust Soft LVQ aan te passen geïntroduceerd worden.

Na het geven van de benodigde achtergrond van LVQ in hoofdstuk 2 zal een nieuwe afstandsmaat geïntroduceerd worden in hoofdstuk 3. De Euclidische afstand wordt uitgebreid met een volledige matrix van adaptieve gewichten. Deze aanpak generaliseert het populaire concept van het leren van relevantie. De diagonale elementen van de adaptieve matrix corresponderen met het expliciet wegen van features, terwijl de buitendiagonale elementen gewicht toevoegen van combinaties van features. Zodoende kunnen correlaties in acht worden genomen bij het evalueren van de gelijkheidswaarde tussen prototypen en featurevectoren. Deze metriek is equivalent aan de gekwadrateerde Euclidische afstand na een lineaire transformatie van data en prototypen naar een nieuwe featureruimte. Zowel matrices met een volledige rang als afstandsmaten welke gelimiteerd zijn tot een klein aantal features kunnen aangeleerd worden. Daar bovenop kan zowel een globaal als een klassegewijs of prototype-specifiek metriek geïmplementeerd worden. We leiden de regels af voor het aanleren van matrices in Generalized LVQ en Robust Soft LVQ. Experimenten met kunstmatige datasets en reële referentiedatasets illustreren de nieuwe techniek in de praktijk. De toepassingen demonstreren dat het aanleren van de matrix is op twee punten profijtelijk: de aanpak heeft een duidelijke meerwaarde wanneer de accuraatheid van de classificatie in beschouwing wordt

genomen en wordt vergeleken met het gebruik van de Euclidische afstand of de gewogen Euclidische afstand. Dit geldt in het speciaal voor lokale-matrixadaptatie bij multi-klasseproblemen. Daarbovenop verbeteren de toegevoegde parameters de interpreteerbaarheid van het uiteindelijke model. Matrixparameters onthullen de belangrijke rol van de invoerdimensies op de diagonaalelementen en de belangrijke rol van correlaties op de buitendiagonale elementen. De experimenten leren ons alsmede dat de twee leeralgoritmen verschillen in leerdynamica en de karakteristieken van het resulterende model.

In hoofdstuk 4 presenteren we een regularisatietechniek om het convergentiegedrag van matrixleren te beïnvloeden. De voorgestelde methode voorkomt dat het leeralgoritme laagrangse afstandsmaten produceert. Zoals vaak wordt geobserveerd, hebben de aanpassingsregels de neiging om een enkele of een erg klein aantal richtingen in featureruimte te kiezen. We introduceren een regularisatieterm welke gebalanceerde eigenwaardeprofielen verkiest van de relevantiematrix. De techniek kan toegepast worden in combinatie met ieder matrixleerschema. We demonstreren haar bruikbaarheid door matrixleren met GLVQ. Naast het gewenste effect op het convergentiegedrag, blijkt de techniek profitabel om overspecialisatie-effecten en numerieke instabiliteiten te voorkomen tijdens het aanleren.

Theoretische aspecten van het matrixleren in LVQ worden onderzocht in hoofdstuk 5. In het bijzonder onderzoeken we het convergentiegedrag in termen van lokale-matrixadaptatie in LVQ1. Bij gesimplificeerde modelaannames wordt aangetoond dat de aanpassingen er toe leiden dat een enkele richting in featureruimte gekozen wordt. Deze richting wordt bepaald door de statistische eigenschappen van de data welke toegekend is aan het respectievelijke prototype.

Hoofdstuk 6 presenteert drie verschillende werkwijzes om Robust Soft LVQ uit te breiden. De studie refereert naar de oorspronkelijke versie van het algoritme dat is gebaseerd op de gekwadrateerde Euclidische afstand. Allereerst stellen we voor om tijdens het trainen van het algoritme, gebaseerd op de gradiëntinformatie van de RSLVQ-kostenfunctie, zijn hyperparameter te wijzigen. Onze experimenten bevestigen dat het leren van de hyperparameter zorgt voor een zeer robuust algoritme ten aanzien van de initiële keus van de hyperparameter. Ook kan voor ieder prototype individueel een lokale hyperparameter aangeleerd worden. Aansluitend presenteren we een alternatieve besluitregel voor RSLVQ-klassificatie: het trainen van de modelparameters maximaliseert de onderliggen kostenfunctie, welke gedefinieerd is door een waarschijnlijkheidsratio. Met deze achtergrond stellen we voor om een nieuw monster tijdens de werkfase toe te kennen aan de klasse met de hoogste waarschijnlijkheidsratio. In tegenstelling tot de afstandgebaseerde klassificatieregel volgt deze aanpak van nature het doel van RSLVQ-training. De hieropvolgende praktische voorbeelden laten zien dat de nieuwe beslissingsregel beter presteert dan

de afstandgebaseerde aanpak, wanneer de methode gecombineerd wordt met lokale adaptieve hyperparameters. Tot slot leiden we de generalisatie van de RSLVQ-kostenfunctie af ten aanzien van de vectoriale klassebenoeming van de invoerdata. Op kunstmatige data demonstreren we dat RSLVQ een special situatie is van het nieuwe fuzzy algoritme.

Bibliography

Arnonkijpanich, B., Hammer, B., Hasenfuss, A. and Lursinap, A.: 2008, Matrix learning for topographic neural maps, *International Conference on Artificial Neural Networks*, Prague, Czech Republic, pp. 572–582.

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

Biehl, M., Breitling, R. and Li, Y.: 2007, Analysis of tiling microarray data by learning vector quantization and relevance learning, *International Conference on Intelligent Data Engineering and Automated Learning*, Springer LNCS, Birmingham, UK, pp. 880–889.

Biehl, M., Ghosh, A. and Hammer, B.: 2007, Dynamics and generalization ability of LVQ algorithms, *Journal of Machine Learning Research* **8**, 323–360.

Biehl, M., Hammer, B., Schleif, F.-M., Schneider, P. and Villmann, T.: 2009, Stationarity of relevance matrix learning vector quantization, *Technical Report MLR-01-2009*, University of Leipzig.

Biehl, M., Hammer, B., Verleysen, M. and Villmann, T. (eds): 2009, *Similarity based clustering - recent developments and biomedical applications*, Vol. 5400 of *Lecture Notes in Artificial Intelligence*, Springer.

Biehl, M., Pasma, P., Pijl, M. and Petkov, N.: 2006, Classification of boar sperm head images using learning vector quantization, *in* M. Verleysen (ed.), *European Symposium on Artificial Neural Networks*, Bruges, Belgium, pp. 545–550.

Biehl, M., Schneider, P., Hammer, B., Schleif, F.-M. and Villmann, T.: submitted, 2010, Stationarity of matrix updates in relevance learning vector quantization.

Bishop, C. M.: 1995, *Neural Networks for Pattern Recognition*, 1 edn, Oxford University Press.

Bishop, C. M.: 2007, *Pattern Recognition and Machine Learning*, 1 edn, Springer.

Boehm, W. and Prautzsch, H.: 1993, *Numerical Methods*, Vieweg.

Bojer, T., Hammer, B., Schunk, D. and von Toschanowitz, K. T.: 2001, Relevance determination in learning vector quantization, *in* M. Verleysen (ed.), *European Symposium on Artificial Neural Networks*, Bruges, Belgium, pp. 271–276.

Cover, T. and Hart, P.: 1967, Nearest neighbor pattern classification, *IEEE Transactions on Information Theory* **13**(1), 21–27.

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

Darken, C., Chang, J., Z, J. C. and Moody, J.: 1992, Learning rate schedules for faster stochastic gradient search, *Neural Networks for Signal Processing 2 - Proceedings of the 1992 IEEE Workshop*, IEEE Press.

Duda, R., Hart, P. and Stork, D.: 2000, *Pattern Classification*, second edn, Wiley-Interscience.

Gath, I. and Geva, A. B.: 1989, Unsupervised optimal fuzzy clustering, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **11**(7), 773–780.

Ghosh, A., Biehl, M. and Hammer, B.: 2006, Performance analysis of lvq algorithms: a statistical physics approach, *Neural Networks* **19**(6), 817–829.

Gustafson, E. and Kessel, W.: 1979, Fuzzy clustering with a fuzzy covariance matrix, *IEEE Conference on Decision and Control*, San Diego, CA, USA, pp. 761–766.

Hammer, B., Schleif, F.-M. and Villmann, T.: 2005, On the generalization ability of prototype-based classifiers with local relevance determination, *Technical Report Ifi-05-14*, Clausthal University of Technology.

Hammer, B., Strickert, M. and Villmann, T.: 2004, Prototype based recognition of splice sites, *Bioinformatic using Computational Intelligence Paradigms*, Springer-Verlag, pp. 25–56.

Hammer, B., Strickert, M. and Villmann, T.: 2005a, On the generalization ability of GRLVQ networks, *Neural Processing Letters* **21**(2), 109–120.

Hammer, B., Strickert, M. and Villmann, T.: 2005b, Supervised neural gas with general similarity measure, *Neural Processing Letters* **21**(1), 21–44.

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

Kaski, S.: 2001, Principle of learning metrics for exploratory data analysis, *Neural Networks for Signal Processing XI, Proceedings of the 2001 IEEE Signal Processing Society Workshop*, IEEE, pp. 53–62.

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

Kohonen, T.: 1986, Learning vector quantization for pattern recognition, *Technical Report TKK-F-A601*, Helsinki Univeristy of Technology, Espoo, Finland.

Kohonen, T.: 1990, Improved versions of learning vector quantization, *International Joint Conference on Neural Networks*, Vol. 1, pp. 545–550.

Kohonen, T.: 1997, *Self-Organizing Maps*, second edn, Springer, Berlin, Heidelberg.

Kohonen, T.: 1998, Learning vector quantization, *The handbook of brain theory and neural networks*, MIT Press, Cambridge, MA, USA, pp. 537–540.

Kusumoputro, B. and Budiarto, H.: 1999, Improvement of artificial odor discrimination system using fuzzy-lvq neural network, *International Conference on Computational Intelligence and Multimedia Applications*, IEEE Computer Society, Los Alamitos, CA, USA, pp. 474–478.

Martinetz, T. and Schulten, K.: 1991, A "neural-gas" network learns topologies, *Artificial Neural Networks* **I**, 397–402.

Mendenhall, M. and Merényi, E.: 2006, Generalized relevance learning vector quantization for classification driven feature extraction from hyperspectral data, *Proceedings of ASPRS 2006 Annual Conference and Technology Exhibition*, p. 8.

Mwebaze, E., Schneider, P., Schleif, F.-M., Haase, S., Villmann, T. and Biehl, M.: 2010, Divergence based learning vector quantization, *in* M. Verleysen (ed.), *European Symposium on Artificial Neural Networks*, Bruges, Belgium, pp. 247–252.

Newman, D. J., Hettich, S., Blake, C. L. and Merz, C. J.: 1998, Uci repository of machine learning databases, `http://archive.ics.uci.edu/ml/`.

Ong, C., A. Smola, A. and Williamson, R.: 2005, Learning the kernel with hyperkernels, *Journal of Machine Learning Research* **6**, 1043–1071.

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

Petersen, K. B. and Pedersen, M. S.: 2008, The matrix cookbook, `http://matrixcookbook.com`.

Prudent, Y. and Ennaji, A.: 2005, A k nearest classifier design, *Electronic Letters on Computer Vision and Image Analysis* **5**(2), 58–71.

Sato, A. and Yamada, K.: 1996, Generalized learning vector quantization, *in* M. C. M. D. S. Touretzky and M. E. Hasselmo (eds), *Advances in Neural Information Processing Systems*, Vol. 8, MIT Press, Cambridge, MA, USA, pp. 423–429.

Sato, A. and Yamada, K.: 1998, An analysis of convergence in generalized lvq, *in* L. Niklasson, M. Bodén and T. Ziemke (eds), *Proceedings of the International Conference on Artificial Neural Networks*, Springer, pp. 170–176.

Schneider, P., Biehl, M. and Hammer, B.: 2007, Relevance matrices in lvq, *in* M. Verleysen (ed.), *European Symposium on Artificial Neural Networks*, Bruges, Belgium, pp. 37–42.

Schneider, P., Biehl, M. and Hammer, B.: 2009a, Adaptive relevance matrices in learning vector quantization, *Neural Computation* **21**(12), 3532–3561.

Schneider, P., Biehl, M. and Hammer, B.: 2009b, Distance learning in discriminative vector quantization, *Neural Computation* **21**(10), 2942–2969.

Schneider, P., Bunte, K., Stiekema, H., Hammer, B., Villmann, T. and Biehl, M.: 2010, Regularization in matrix relevance learning, *IEEE Transactions on Neural Networks* **21**(5), 831–840.

Schneider, P., Schleif, F.-M., Villmann, T. and Biehl, M.: 2008, Generalized matrix learning vector quantizer for the analysis of spectral data, *in* M. Verleysen (ed.), *European Symposium on Artificial Neural Networks*, Bruges, Belgium, pp. 451–456.

Seo, S., Bode, M. and Obermayer, K.: 2003, Soft nearest prototype classification, *IEEE Transactions on Neural Networks* **14**(2), 390–398.

Seo, S. and Obermayer, K.: 2003, Soft learning vector quantization, *Neural Computation* **15**(7), 1589–1604.

Seo, S. and Obermayer, K.: 2006, Dynamic hyper parameter scaling method for lvq algorithms, *International Joint Conference on Neural Networks*, Vancouver, CA.

Shalev-Schwartz, S., Singer, Y. and Ng, A.: 2004, Online and batch learning of pseudo-metrics, *Proceedings of the 21st International Conference on Machine Learning*, ACM, New York, USA, p. 94.

Strickert, M., Witzel, K., Mock, H.-P., Schleif, F.-M. and Villmann, T.: 2007, Supervised attribute relevance determination for protein identification in stress experiments, *Machine Learning in Systems Biology*, pp. 81–86.

Tamura, H. and Tanno, K.: 2008, Midpoint-validation method for support vector machine classification, *IEICE - Transactions on Information Systems* **E91-D**(7), 2095–2098.

Thiel, C., Sonntag, B. and Schwenker, F.: 2008, Experiments with supervised fuzzy lvq, *in* L. Prevost, S. Marinai and F. Schwenker (eds), *Artificial Neural Networks in Pattern Recognition*, Vol. 5064 of *Lecture Notes in Computer Science*, Springer, pp. 125–132.

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

Weinberger, K., Blitzer, J. and Saul, L.: 2006, Distance metric learning for large margin nearest neighbor classification, *in* Y. Weiss, B. Schölkopf and J. Platt (eds), *Advances in Neural Information Processing Systems*, MIT Press, Cambridge, MA, USA, pp. 1473–1480.

Wu, K.-L. and Yang, M.-S.: 2003, A fuzzy-soft learning vector quantization, *Neurocomputing* **55**(3-4), 681–697.