# Integration of Auxiliary Data Knowledge in Prototype Based Vector Quantization and Classification Models

Der Fakultät für Mathematik und Informatik
University Leipzig
angenommene

DISSERTATION

zur Erlangung des akademischen Grades

DOCTOR RERUM NATURALIUM MARIKA KADEN
(Dr. rer. nat. Marika Kaden)
im Fachgebiet
Informatik

Vorgelegt von
M. Sc. Marika Kaden

geboren am 26.07.1985 in Räckelwitz
Geburtsname: Kästner

Die Annahme der Dissertation haben empfohlen:

1) Professor Dr. Martin Bogdan (Leipzig)
2) Professor Dr. John A. Lee (Louvain, Frankreich)
3) Professor Dr. Thomas Villmann (Mittweida)

Die Verleihung des akademischen Grades erfolgt mit Bestehen der Verteidigung am 23.05.2016 mit dem Gesamtprädikat *magna cum laude*.

## Bibliographic Descriptions

Kaden, Marika:
Integration of Auxiliary Data Knowledge in Prototype Based Vector Quantization and Classification Models. - 2016. - 195 pages, Leipzig, University Leipzig, Faculty of Mathematics and Computer Science, Dissertation, 2016

# Contents

## Acknowledgement

Thanks to all those which support me on the whole way of learning, studying and working.

First of all my special thank goes to my adviser Prof. Thomas Villmann (Villy). He support me very much and believe in me all my PHD-time. Moreover, I thank Martin Bogdan as my second adviser.

Furthermore, I would like to warmly thank the whole *Computational Intelligence Group* of Mittweida especially David Nebel for the lot of discussions, my *dissister* Tina Geweniger supporting and helping me for further enhancing of the thesis especially improving the English, and my *dissister* Mandy Lange, who shared the office with me.

I also thank the ESF/SAB for financial support.

Last but not least: Thanks to my family and friends: Ganz herzlich möchte ich mich bei meinem Mann Kai-Uwe, meinen Eltern, Schwiegereltern und Freunden bedanken. Alle haben die ganze Zeit an mich geglaubt und mich unterstützt, wo es nur ging. Danke!!!

Marika Kaden

# Symbols and Abbreviations

| | |
|---|---|
| $\mathcal{H}$ | Hilbert space |
| $\mathbb{R}_+$ | positive real number including zero |
| $N_V \in \mathbb{N}$ | number of data point |
| $N_W \in \mathbb{N}$ | number of prototypes |
| $N_C \in \mathbb{N}$ | number of classes |
| $D \in \mathbb{N}$ | number of dimensions/features |
| $\iota \in \mathbb{N}$ | iteration step |
| $V \in \mathbb{R}^D$ | set of data points |
| $\mathbf{v} \in V$ | data point |
| $v(t) \in V$ | functional data point |
| $W \in \mathbb{R}^D$ | set of prototypes |
| $\mathbf{w} \in W$ | prototype |
| $w(t) \in W$ | functional prototype |
| $s(\mathbf{v}) \in \mathbb{N}$ | index of the winning prototype for $\mathbf{v}$ |
| $R_j(V, W) \subseteq V$ | receptive field of prototype $\mathbf{w}_j$ |
| $A$ | grid or lattice (SOM) |
| $\mathbf{w}_\mathbf{r} \in W$ | prototype assigned to neuron $\mathbf{r}$ (SOM) |
| $\mathbf{r} \in A$ | neuron on the grid $A$ |
| $\hat{\mathbf{s}}(\mathbf{v}) \in A$ | neuron of the according winning prototype (SOM) |
| $C = \{1, \ldots, N_C\}$ | set of classes |
| $c(\mathbf{v}) \in \mathcal{C}$ | class assignment of a data point |
| $\mathbf{c}(\mathbf{v}) \in [0,1]^{N_C}$ | fuzzy class assignment vector of a data point |
| $\hat{c}(\mathbf{v}) \in \mathcal{C}$ | predicted class of a data point |
| $y(\mathbf{w}) \in \mathcal{C}$ | class assignment of a prototype |
| $\mathbf{y}(\mathbf{w}) \in [0,1]^{N_C}$ | fuzzy class assignment vector of a prototype |
| $d(\mathbf{v}, \mathbf{w}) \in \mathbb{R}_+$ | dissimilarity measure of $\mathbf{v}$ and $\mathbf{w}$ |
| $d_E(\mathbf{v}, \mathbf{w}) \in \mathbb{R}_+$ | Euclidean metric of $\mathbf{v}$ and $\mathbf{w}$ |
| $d_E^2(\mathbf{v}, \mathbf{w}) \in \mathbb{R}_+$ | squared Euclidean dissimilarity measure of $\mathbf{v}$ and $\mathbf{w}$ |

| | |
|---|---|
| $\|\|\cdot\|\|_{\mathcal{H}}, \|\|\cdot\|\|_E, \|\|\cdot\|\|_{l_p}$ | norm of $\mathcal{H}$, Euclidean norm, $l_p$-norm |
| $<\cdot,\cdot>_{\mathcal{H}}$ | inner product in $\mathcal{H}$ |
| $\kappa(\mathbf{v},\mathbf{w}) \in \mathbb{R}_+$ | positive semi-definite kernel |
| $lc(\mathbf{v}, W) \in \mathbb{R}_+$ | local cost function |
| $\sigma \in \mathbb{R}_+$ | neighborhood range |
| $\theta$ | transfer function parameter (GLVQ) |
| $\delta_{i,j} \in \{0,1\}$ | Kronecker delta |
| $\mathtt{H}(x)$ | Heaviside function |
| $\mathtt{h}(x)$ | Delta distribution |

| | |
|---|---|
| **SGD** | **S**tochastic **G**radient **D**escent |
| **VQ** | **v**ector **q**uantization |
| **uVQ** | **u**nsupervised **VQ** |
| **SOM** | **S**elf-**O**rganizing **M**aps (Heskes) |
| **NG** | **N**eural **G**as |
| **semi-sVQ** | **semi**-**S**upervised **VQ** |
| **LASSO** | **L**earning **A**ssociations by **S**elf-**O**rganization |
| **Tib-LASSO** | **Tib**shirani- **L**east **A**bsolut **S**election and **S**hrinkage **O**perator |
| **FLNG/FLSOM** | **F**uzzy **L**abeled **NG**/**SOM** |
| **FSNG/FSSOM** | **F**uzzy **S**upervised **NG**/**SOM** |
| **sVQ** | **s**upervised **VQ** |
| **LVQ** | **L**earning **V**ector **Q**uantization |
| **GLVQ** | **G**eneralized **LVQ** |
| **GRLVQ** | **G**eneralized **R**elevance LVQ |
| **GMLVQ** | **G**eneralized **M**atrix LVQ |
| **LiRaM LVQ** | **L**imited **R**ank **MLVQ** |
| **DK-GLVQ** | **D**ifferential **K**ernel GLVQ |
| **SVM** | **S**upport **V**ector **Machine** |
| **SV** | **S**upport **V**ectors |
| **RBF** | **R**adial **B**asis **F**unction |
| **GFR/MLVQ** | **G**eneralized **Functional R**elevance/**M**atrix **LVQ** |
| **eGR/MLVQ** | enhanced **G**eneralized **R**elevance/**M**atrix **LVQ** |
| **BS-GLVQ** | **b**order-**s**ensitive **GLVQ** |
| **P-GLVQ** | **GLVQ** with additive penalty function |
| **AEA** | **A**symmetric **E**rror **A**ssessment |
| $\beta$**-GlVQ**, **Γ-GLVQ** | **GLVQ** with (asymmetric) misclassification costs |
| **ROC**-curve | **R**eceiver **O**peration **C**haracteristic curve |
| **AUROC** | **A**rea **u**nder the **ROC**-curve |

# Chapter 1

# Introduction

## 1.1 Motivation and Problem Description

In the last decades, the possibilities of recoding data volumes have been developing rapidly. This development requires new techniques in data processing like storage, reading and analyzing abilities for the huge amounts of data. In this context, *Big Data* is one of the buzzwords becoming popular in the last years. Yet, an explicit definition of *Big Data* cannot be found. The term includes the handling of huge data sets in general. Thereby the comprehensions of the words *handling* and *huge* depend on the user, his field of research and point in time.

The fact of having such huge data volumes does not only enforce benefit but also challenges especially in the analysis of such data sets. Obviously, manual handling of the data is not longer possible in many applications. Thus, the development of methods which extract essential information from the data shifts more and more into focus. Often, the analyses of large data sets are summarized in the term *Data Mining*. Again, the keyword *Data Mining* does not have a clear definition. In general, methods belonging to *Data Mining* search systematically for hidden information about and in the data sets.

An essential part of the collective term *Data Mining* is *Machine Learning* (ML) which is an additional tool for classical statistic for data processing. The goal of ML methods is to generate a generalized model of the input data by *learning from examples*: In contrast to classical approaches, not an exact physical, chemical or biological model is developed, yet, a model is learned using a given data base. Due to the effectiveness and versatility of applications, the ML methods become important for the analysis of *Big Data*.

In general, I distinguish into the three groups of ML methods:

**unsupervised** Given is a data set. It is searched for a generalized but compressed model describing the data set as close as possible.

**supervised** Given are input data and output information. Learning takes place as a reduction between predicted and true output of the data. Thereby it is distinguished between discrete output, the *classification*, and real output, denoted as *regression*. Problems, which include both, data with and without output information, are called *semi-supervised*.

**reinforcement learning** A special kind of supervised learning is the reinforcment learning, where the output reward information, i. e. success or non-success, is usually very sparse. This feedback is given after a delay-time only and not immediately like in supervised learning.

In this thesis, the focus is put on supervised learning, especially on classification. For this purpose I will utilize basic tools of unsupervised learning. The required details will be described and related to the general (un-)supervised methods in this thesis. For an introduction to reinforcement learning methods I refer to [Sutton and Barto, 1988].

The application of unsupervised ML methods is diverse:

**basic analysis** Self-organizing Maps [Kohonen, 1982], Hidden Markov Model [Baum et al., 1979]), Generative Topographic Mapping [Bishop et al., 1998],

**visualization** Principle Component Analysis, Multidimensional Scaling, Stochastic Neighborhood Embedding[1]

**clustering/compressing** c-means [MacQueen, 1967], Fuzzy c-means [Bezdek, 1974], Affinity Propagation [Frey and Dueck, 2007], Hierarchical Clustering [Kaufman and Rousseeuw, 1990], Spectral Clustering [v. Luxburg, 2007].

Clustering methods assign *similar* data points to groups of so called clusters. The term *similar* is task dependent and cannot be defined clearly. Hence, the problem of clustering as well as visualization belongs to the ill-posed problems according to the definition of `Hadamard` [Hadamard, 1902], i. e. the evaluation and comparison of models is not unique due to the unclear definition of *similar*.

As for unsupervised tasks, for supervised problems exist a huge number of algorithms and methods based on different concepts, e. g.:

**Support Vector Machine** classification/regression based on convex optimization and learning theory [Schölkopf and Smola, 2002]

**Neuronal Network** biological inspired classification/regression [Haykin, 1994]

**Bayes Classifier** classification based on statistical decision theory[Bishop, 2006]

**Learning Vector Quantization** prototype based methods obtained by combination of Bayes classifier and vector quantization [Kohonen, 1986]

The number of new or extended models for classification is rapidly increasing. One reason thereof is the diversity in the problems. The data bases can be distinguished with regard to many attributes like the number of data samples available for learning, imbalanced class probabilities, class separability, the characteristic, e. g. vectorial data or non-vectorial data, etc. The following two examples illustrate those attributes.

[1] An overview of visualization methods is given in [v. d. Maaten et al., 2009]

**A - remote sensing data** A data set may consist of spectra recorded with a hyperspectral camera over an area. Thereby, reflecting or absorbing spectra are assigned to a pixel representing soil conditions in this area. Here, two specificities regarding classification problems have to be pointed out. First, within the spectra occur correlations depending on the wavelengths/frequencies. Thus, although hyperspectra are usually provided as high dimensional vectors, the problem complexity is much smaller due to these dependencies. The correlations can be incorporated during learning to built up more efficient classification models. Second, in general, a pixel has a resolution of several squared meters on the ground. Hence, a mixture of soil conditions is reflected and thus, a unique class assignment cannot be given. This aspect causes difficulties in classification and is related to this thesis.

**B - medical data** In medical classifications application a typical task is to assign a diagnosis to a patient record based on measurements, personal data, etc. Frequently, the data basis for model training suffers due to the fact that the number of patients is much lower than the number of healthy persons, i. e. I observe imbalanced class probabilities. For these cases classification models simply optimizing classification accuracy during training might be misleading. Therefore, particularly in medicine, *sensitivity* and *precision* are favored which should be reflected in the classifier. Further, misclassification costs can be non-symmetric, i. e. a misclassification of a<n healthy person to an unhealthy person might be more costly with respect to side effects or treatment expenses than the other way around. Again, these aspects should be incorporated into the model training.

In general, I believe that those aspects as well as other expert knowledge and a-priori information should contribute to classifier specification to obtain more appropriate and task dependent models. Further, the interpretability of a model gains importance especially in applications.

In this thesis I concentrate on the prototype based classification which originated from unsupervised Vector Quantization. Particularly, I focus on models optimizing well defined criteria comprised in so called cost-functions. This description allows a mathematical precise treatment on the one hand. On the other hand prototype based methods generally allow a good interpretation. Moreover, I will investigate how to integrate expert knowledge in those classifier models. Especially I will consider two types of integration:

**I** selection and task driven adaption of dissimilarity measures for data comparison

**II** direct integration of expert knowledge into the structure of cost functions.

The functionality of the new investigations is demonstrated on several artificial and real world data sets which are described in the next section.

## 1.2 Utilized Data Sets

The classification problems with underlying data sets are manifold. The kind and characteristics of such problems are described in Section. 3. In the following, the data sets used in this thesis are described closer. An overview of the important facts of the single data sets can be found in Table 1.4. Further, it is distinguished between artificial and real world data sets.

### Artificial Data Sets

Artificial data sets reflect typical properties of real data sets and the methods should be able to handle them adequately. The design of artificial data sets gives the possibility to demonstrate the specialty of the proposed methods. Further, if the dimensionality of the data is set to two or three, the resulting model can be visualized.

#### Flags of Palau, Ukraine and Czech

The flags of the nations are very multifaceted regarding colors, symbols, and geometric forms. Therefore, they are well suited for the creation of artificial 2D-data sets with special features and the visualization of the specific ideas of the classification methods. In our case, we use the special characteristics of the flags: Palau, Ukraine and Czech.

**Palau-Flag** The flag of Palau[2] with the derived data set are depicted in Fig. 1.1. The data set consists of two classes and 1099 data points with a class distribution around one to three. The special characteristic of this data set is the non-linear separability in the two dimensional Euclidean space.

**Ukraine-Flag** The design of the Ukraine flag (see Fig 1.2) is a simple linear separable problem. The data set consists of two class with 1000 data points per class.

**Czech-Flag** The Czech flag reflects a three class problem (see Fig 1.3), where the classes are pairwise linear separable and touch each other. The number of total data points is 1000.

#### Triangle

The *Triangle* data set consists of two-dimensional data points divided into three classes. Each class is made of 100 equally distributed data samples which are arranged in a rectangle. The special characteristic is the overlap of the pairwise classes (see Fig. 1.4).

[2]Palau is an island country next to the Philippines and belonging to the island group of Micronesia.

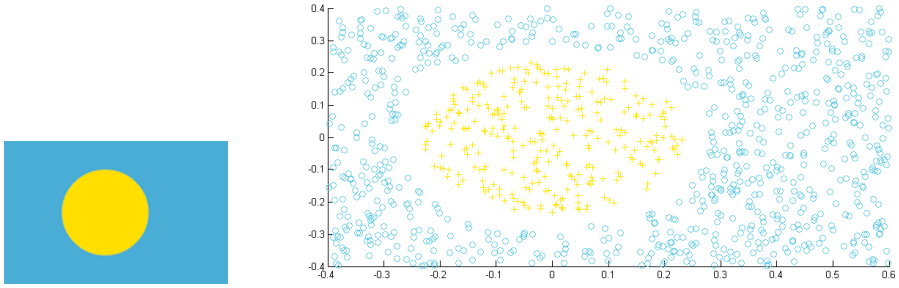

**Figure 1.1**: *Palau data set*: original Palau flag (left), derived 2D data set with the two classes $+$ and $o$ (right)

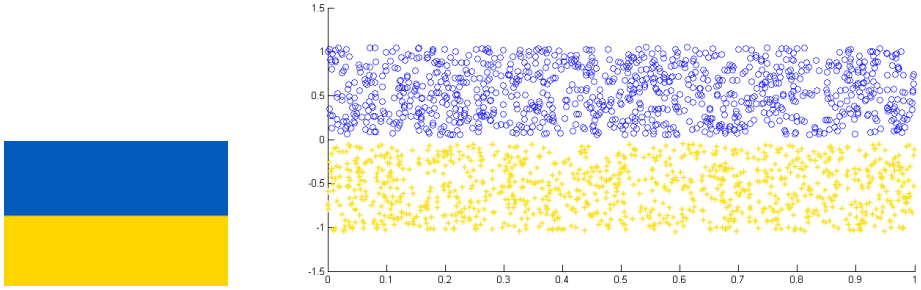

**Figure 1.2**: *Ukraine data set:* original Ukraine flag (left), derived 2D data set with the two classes $+$ and $o$ (right)

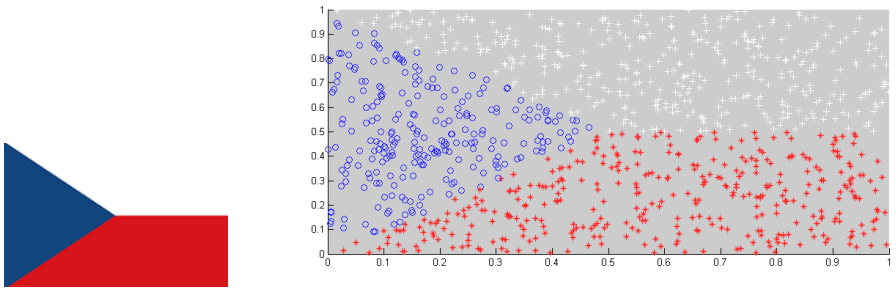

**Figure 1.3**: *Czech data set:* original Czech flag (left), derived 2D data set with the three classes $o$, $\star$ and white $+$ (right)

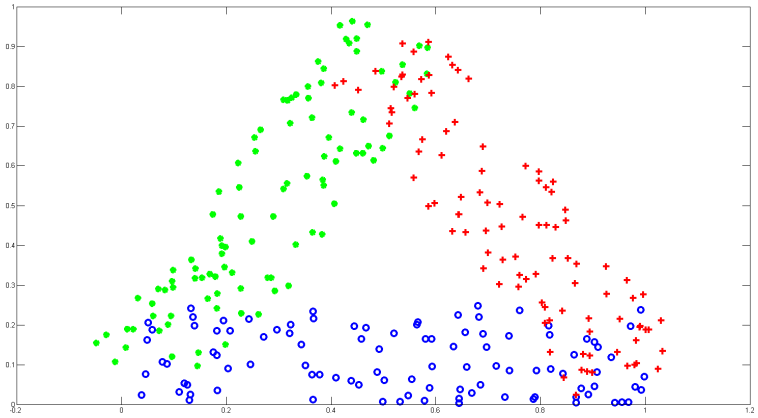

**Figure 1.4**: *Triangle data set:* two-dimensional data set with the three classes $o$, $\star$ and $+$. Each class consist of 100 data points.

### Real world data sets

The novel methods should also work on real world data sets. A special kind of real world data are functional data:

**Note 1.1.** *Functional* data are data points with lateral dependencies in the neighbored dimensions. In general, we consider vectors $v_t \in \mathbb{R}^D$, which are discrete representations of functions $v(t)$. Examples are time series or spectra.

The following real world data sets, used in this thesis, are described closer. The data sets belonging to different kind of scientific fields like food industry, remote sensing or medicine.

#### Coffee - hyperspectra from the food industry

The *Coffee data set* contains *short-wave infrared range* (SWIR) spectra of five different coffee types (see Fig. 1.5). The data set is provided by Prof. Udo Seiffert and his research group at Fraunhofer IFF Magdeburg/Germany. The spectra are obtained by utilizing a hyperspectral camera (HySpex SWIR-320m-e, Norsk Elektro Optikk A/S) with the SWIR between 970 nm and 2500 nm at 6 nm resolution yielding 256 bands per spectrum. Moreover, proper image calibration was done by using a standard reflection pad (polytetrafluoroethylene (PTFE)) [Backhaus et al., 2011]. A data sample is a vector containing 256 absorption values. Per coffee type we selected 5000 spectral vectors and normalized them by the $l_2$-norm.

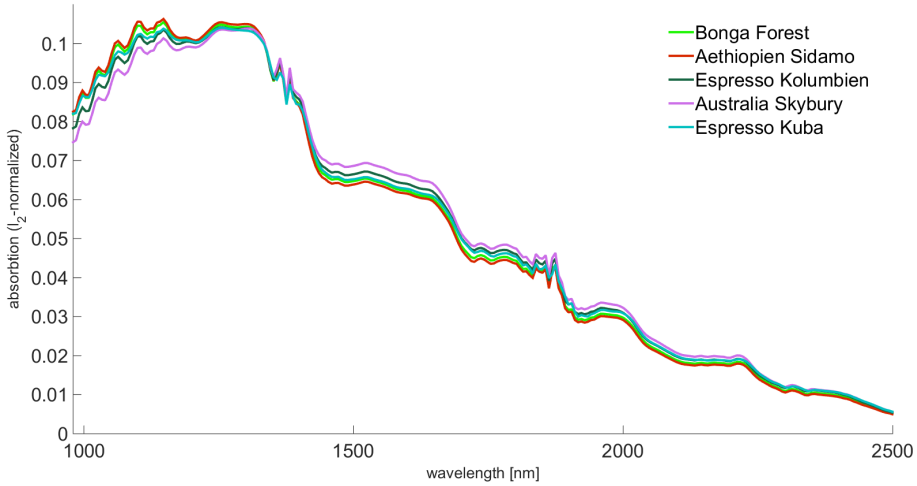


**Figure 1.5**: *Coffee data set:* mean spectra of the five coffee. types

### Colorado LANDSAT TM - spectra of remote sensing

The sensors of LANDSAT TM produce images of the ground with seven different spectral bands. The resolution of the band $1-5$ and $7$ is $30 \times 30\,m^2$ and of band $6$ it is $60 \times 60\,m^2$. Thereby, the first three bands are in the visible spectrum (corresponding roughly to the colors blue, green and red) and the bands $4, 5, 7$ are in the near-infrared spectrum. Thermal band $6$ has a lower spatial resolution and therefore, it was dropped it according to common practice [Augusteijn et al., 1995]. The bands are designed to detect and distinguish between different vegetations, rock formations, and other ground characters. The Colorado LANDSAT TM data set[3], abbreviated *Colorado*, is an image of a mountainous region in central Colorado with the pixel size $1907 \times 1784$. The basis of this data set are manually labeled pixels of the image, i. e. each label is ground truth. The ground cover is subdivided into $14$ classes with different occurrences (see Tab. 1.1).

The full Colorado data set consists of $3\,402\,088$ data samples with six bands. For training the data set is usually divided into $10\%$ training and $90\%$ test data samples keeping the class distribution. Moreover, the spectra are normalized by the $l_2$-norm.

Further, a second data set is derived. Therefore, I packed $10 \times 10$ pixels of the image to one data sample by averaging the single features and generated fuzzy label vectors. A fuzzy label vectors $\mathbf{c}(\mathbf{v}) \in [0,1]^{14}$ contains the percentage of each class in the packed $10 \times 10$ image. The resulting data set consists of $33\,820$ data points belonging to a $190 \times 178$ packed image.

[3]The data set was provided by M. Augustijn (University of Colorado).

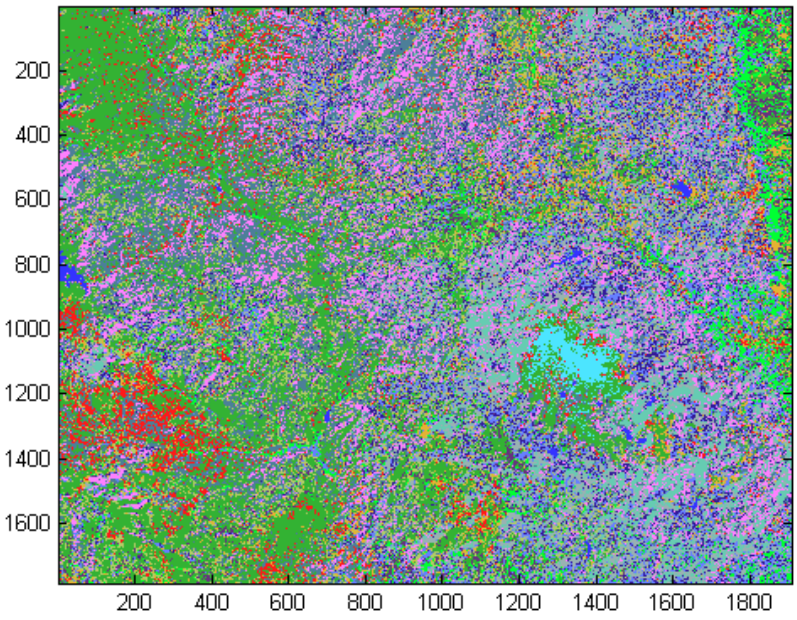


**Figure 1.6**: *Colorado data set*: False color image of a mountainous region in Colorado with 14 different kinds of vegetation or other ground characters (class assignments in Table 1.1).

### Indian Pine - spectra of remote sensing

The *Indian Pine* data set[4] was generated by an AVIRIS sensor capturing an area corresponding to $145 \times 145$ pixels in the Indian Pine test site in the northwest of Indiana (see Fig. 1.8, [Langrebe, 2003]). The spectrometer operates in the visible and mid-infrared wavelength range ($0.4 - 2.4\mu\text{m}$) with $D = 220$ equidistant bands. The area includes $16$ different kinds of forest or other natural perennial vegetation and non-agricultural sectors, which are also denoted as background. These background pixels are removed from the data set as usual. Additionally, we remove $20$ wavelengths, mainly affected by water content (around $1.33\mu\text{m}$ and $1.75\mu\text{m}$, see Fig. 1.9). Finally, all spectral vectors were normalized according to the $l_2$-norm. This overall preprocessing is usually applied to this data set [Langrebe, 2003]. The mean spectra of the $16$ classes are depicted in Figure 1.9.

[4]The data set can be found at www.ehu.es/ccwintco/uploads/2/22/Indian_pines.mat

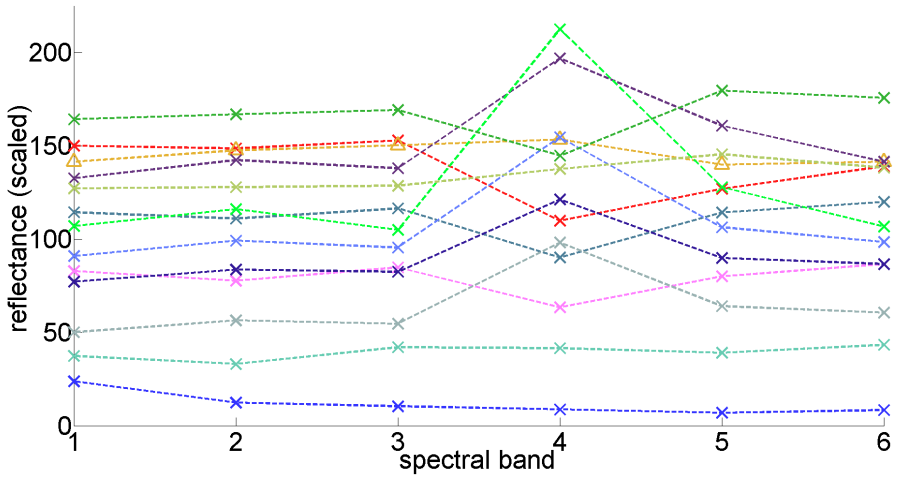


**Figure 1.7**: *Colorado data set*: mean bands of the 14 ground cover classes with the class assignment in Table 1.1.

| class | description | quantity |
|---|---|---|
| 1 | Scotch pine | 17.1 % |
| 2 | Douglas fir | 10.4 % |
| 3 | Pine/fir | 5.3 % |
| 4 | Mixed pine fir | 8.0 % |
| 5 | Supple/prickle pine | 4.3 % |
| 6 | Aspen/mixed pine forest | 6.1 % |
| 7 | Without vegetation | 5.0 % |
| 8 | Aspen | 8.2 % |
| 9 | Water | 0.5 % |
| 10 | Moisten meadow | 2.9 % |
| 11 | Bush land | 3.7 % |
| 12 | Grass/pastureland | 7.8 % |
| 13 | Dry meadow | 19.8 % |
| 14 | Alpine vegetation | 0.8 % |

**Table 1.1**: *Colorado data set:* name of the classes with the class distribution (color and classes are conformal to Figure 1.7)

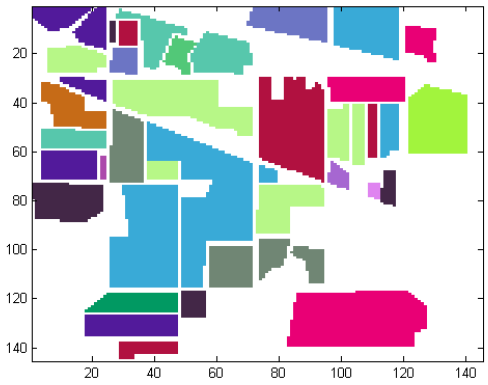


**Figure 1.8**: *Indian Pine data set:* $145 \times 145$ pixels in the *Indian Pine* test site with 16 different kinds of natural perennial vegetation or non-agricultural areas.

| class | description | quantity |
|---|---|---|
| 0 | background | 10659 |
| 1 | Alfalfa | 54 |
| 2 | Corn-no till | 1434 |
| 3 | Corn-mimimum till | 834 |
| 4 | Corn | 234 |
| 5 | Grass/pasture | 497 |
| 6 | Grass/trees | 747 |
| 7 | Gass/pasture-mowed | 26 |
| 8 | Hay-windrowe | 489d |
| 9 | Oats | 20 |
| 10 | Soyabeans - to till | 968 |
| 11 | Soyabeans - minimum till | 2486 |
| 12 | Soyabeans | 614 |
| 13 | Wheat | 212 |
| 14 | Woods | 1294 |
| 15 | Building-grass-trees-drives | 380 |
| 16 | Stone-stell towers | 95 |

**Table 1.2**: *Indian Pine data set:* name of the classes and class distributions (color and classes are conformal to Figure 1.8)

### Morbus Wilson - medical data set

Morbus Wilson is an autosomal-recessive disorder of copper metabolism. The copper accumulates in the central nerve system, liver and other organs, which leads to disturbances in liver function and basal ganglia showing hepatic and extrapyramidal mo-

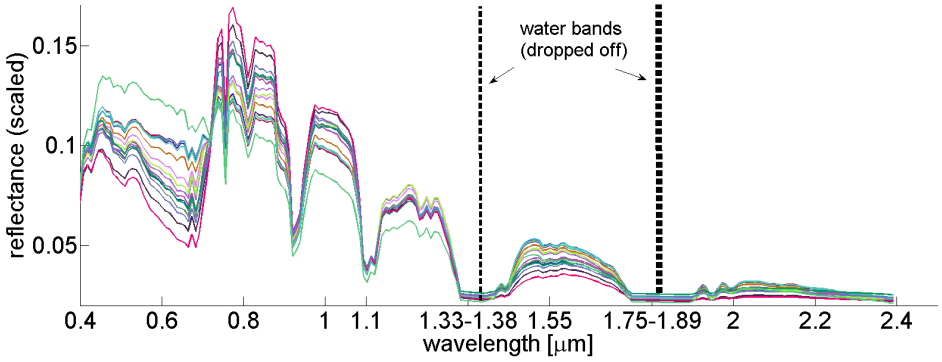


**Figure 1.9**: *Indian Pine data set:* mean spectra of the 16 ground cover classes (class assignment see Table 1.2)

tor symptoms. Patients suffering Morbus Wilson develop neurophysiological impairments. In the initial non-neurologic phase (NN), impairments are negligible or at least not defacing, whereas later during the neurologic phase (N) the disturbances become severe [Hermann et al., 2003]. During the course of the disease the non-neurological phases manifested in the neurological state. Medical treatment may slow this process down and reduce the symptoms. Depending on the impairment level and the respective pharmaceutical dose rate the treatment causes side effects and could also be expensive. Therefore, a precise classification is demanded.

According to a clinical scheme suggested by `Konovalov`, patients can be divided into two main groups: neurological (N) and non-neurological (NN) [Hermann et al., 2003]. Moreover, these two main groups can be subdivided. The neurological group includes the pseudo-sclerotic (PS), pseudo-parkinsonian (PP) and merged type (MT) subgroup. The non-neurological group can be separated into the hepatic type (HT) and the asymptomatic type (AT). A further group consisted of healthy volunteers (V). A clinical (expert) distinction between these subgroups is difficult and requires substantial medical expertise. Different physical examinations are usually applied including among others genetic analysis, fine-motoric, and neuro-physiological tests. The results are condensed in the expert diagnosis by the medical doctor.

There are two different data sets related to Morbus Wilson [5]:

**Morbus Wilson - EIP** The impairments of the nervous pathways can be detected by investigation of latencies of evoked potentials collected in a data vector denoted as electro physiological impairment profile (EIP). The Wilson data set *EIP* contains 122 five-dimensional EIPs described in [Hermann et al., 2003]. The task here is, to classify the patients according to the `Konovalov`-scheme only on the basis of their neurophys-

[5]The data sets were provided by Wieland Hermann (Paracelsus-Klinik Zwickau, Germany).

| class | PS | PP | MT | HT | AT | V |
|---|---|---|---|---|---|---|
| EIP2 | | N | | | NN | |
| EIP4 | PS | PP+MT | | HT+AT | | V |
| EIP6 | PS | PP | MT | HT | AT | V |
| number | 34 | 14 | 8 | 9 | 8 | 47 |

**Table 1.3**: *Morbus Wilson data set:* class subdivisions

iological data. However, it is not clear whether a precise classification based on these data is possible [Hermann et al., 2002].

Three different tasks can be derived from the EIP data set: The first task is to distinguish between the two main groups only, NN (including V) vs N. A more challenging task is the six-class problem with all the three subgroups. Due to the small number of data samples, we merge several classes (see Tab. 1.3) to obtain a third data set *EIP4*.

**Morbus Wilson - PET** The accumulation of copper in the central nervous system causes a disturbed glucose consumption. To evaluate the smooth transition between the NN and N phase a *[$^{18}F$]-Fluorodesoxyglucose-Positron-Emission-Tomography* ([$^{18}F$]FDG-PET) was applied delivering a neurological impairment profile for each patient/proband (see Fig. 1.10) [Hermann et al., 2002].

Thus, the second Wilson data set, denoted as *PET*, consists of an eleven-dimensional vector containing the normalized glucose consumption in different brain regions (frontal lobe, parietal lobe, temporal lobe, occipital lobe, ant. cingulum, post cingulum, putamen, caput nuclei caudati, cerebellum, midbrain, thalamic area). This data set can be used to learn a binary classification decision based on the neurophysiological impairment profile. The PET data set contains a non-neurological group (NN) consisting $15$ volunteers samples and $16$ non-neurologic samples, and a neurological group (N) of $34$ neurologic samples (N).

**Tecator - spectra of the food industry**

The *Tecator* data sets consists of 215 spectra measured for several meat probes. The spectral range is $850$ - $1050$ $nm$ with $D = 100$ spectral bands (see Fig. 1.11). The original data set is labeled according to the fat and protein level. We only use the labels for the fat content and divided the probes between high and low fat level. Further, the data set is provided as a training set ($N_{V_{train}} = 172$) and a test set ($N_{V_{test}} = 43$) [Krier et al., 2009]. The data set is available at library *StaLib*[6].

[6]Tecator data set is available at http://lib.stat.cmu.edu/datasets/tecator.

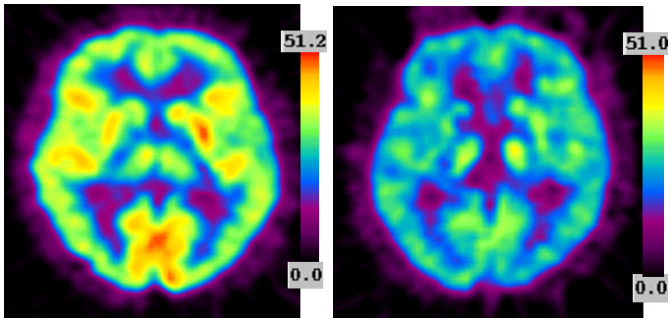


**Figure 1.10**: *Wilson data set:* [$^{18}F$]FDG-PET analysis: normal (left) and disturbed (right) [Hermann et al., 2002].

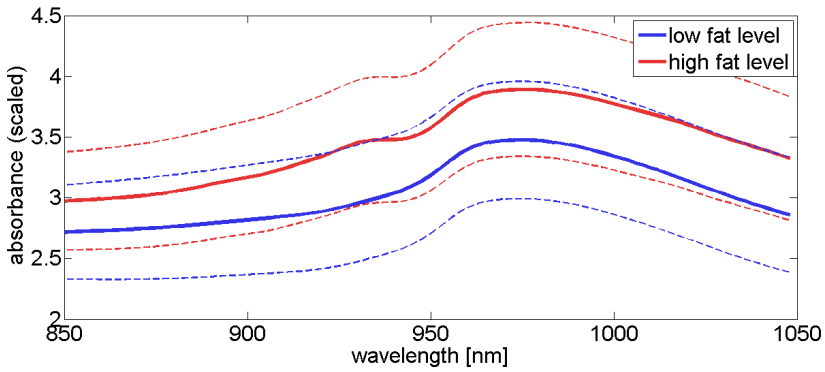


**Figure 1.11**: *Tecator data set:* mean absorbing spectra for the two classes *high* and *low fat* content with their standard deviation (dotted lines).

### Wine - spectra of the food industry

Another example from the food industry is the *Wine* data set which contains 121 absorbing infrared spectra of wine. The spectra ranges from 4000 - 400 $cm^{-1}$ (or equivalently from $2.5 - 25\mu m$) with $D = 256$ equidistant frequency bands (see Fig.1.12). The data are labeled according to the two alcohol levels (low/high) as given in [Krier et al., 2009]. Further, we use the same splitting between training ($N_{V_{train}} = 94$) and test data ($N_{V_{test}} = 30$) as in [Krier et al., 2009]. The spectra with the number 34, 35 and 84 are identified as outliers and dropped off the data set. The wine data set can be downloaded from the UCI repository[7].

[7]Wine data set is available at http://www.ucl.ac.be/mlg/index.php?page=database.

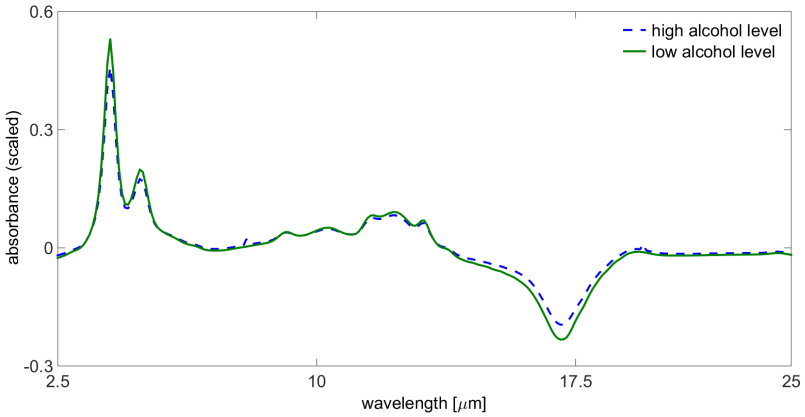


**Figure 1.12**: *Wine data set:* mean absorbing spectra for the two classes. The blue line corresponds to a high alcoholic level whereas the green one stands for a low level.

| data set | subsets | $N_V$ | $D$ | $N_C$ | normalization | characteristic | applied |
|---|---|---|---|---|---|---|---|
| Palau-Flag | - | 1099 | 2 | 2 | - | artificial | GLVQ, DK-GLVQ |
| Ukraine-Flag | - | 2000 | 2 | 2 | - | artificial | BS-GLVQ |
| Czech-Flag | - | 1000 | 2 | 3 | - | artificial | P-GLVQ, BS-GLVQ |
| Triangle | - | 300 | 2 | 3 | - | artificial | $\beta$-GLVQ, $\Gamma$-GLVQ |
| Coffee | - | 25.000 | 256 | 5 | $l_2$ | hyper-spectra | GRLVQ, sGRLVQ, GMLVQ, DK-GMLVQ |
| Colorado | all | 3 402 088 | 6 | 14 | - | spectral image | GLVQ, FSSOM |
| | packed | 33 820 | 6 | - | | spectral image with fuzzy labeling | FSNG, FSMNG |
| Indian Pine | - | 10 366 | 200 | 16 | $l_2$ | hyper-spectral image | eGMLVQ |
| Morbus Wilson | EIT2 | 122 | 11 | 2 | z-score | clinical trail | BS-GLVQ, SVM |
| | EIT4 | 122 | 11 | 4 | z-score | clinical trail | $\beta$-GLVQ, $\Gamma$-GLVQ |
| | EIT6 | 122 | 11 | 6 | z-score | clinical trail | $\beta$-FSNG, U-FSNG |
| | PET2 | 65 | 7 | 2 | z-score | clinical trail | $F_\beta^2$-GLVQ |
| Tecator | - | 215 | 100 | 2 | - | absorbent spectra | GFRLVQ |
| Wine | - | 121 | 256 | 2 | - | infrared spectrum | GRLVQ, GMLVQ, sGRLVQ GFRLVQ, sGFRLVQ, eGRLVQ |

**Table 1.4**: Overview of the important facts of the used data sets ($N_V$-number of data points, $D$ - number of features/dimension, $N_C$- number of classes)

# Chapter 2

# Prototype Based Methods for Vector Quantization

Generally speaking, the goal of vector quantization methods (VQ) is to partition of the data space. Vector quantization in the context of Machine Learning is a powerful and essential tool for exploring and investigating the underlying structure of a data set. This thesis concentrates on prototype based methods, i. e. the data are described by representatives also denoted as prototypes. In the following *prototype based VQ* we will abbreviated by VQ.

The VQ methods can essentially be divided into unsupervised (uVQ), supervised (sVQ), and semi-supervised (semi-sVQ) methods. One goal among others is to estimate the density distribution of the data set by a few prototypes. Thereby, the data samples together with their (dis-)similarities are given. For unsupervised VQ the cost function, also denoted as *expected risk function*, is the quantization error of the model. Well known uVQ methods are the c-means (CM, [MacQueen, 1967]), the Fuzzy-c-means (FCM, [Bezdek, 1974]), the Self-Organizing Maps (SOM, [Kohonen, 1998]), and Neural Gas (NG, [Martinetz et al., 1993]). These methods try to minimize the description error by adapting the prototypes. Thereby, in CM as well as in FCM the prototypes are learned independently [Buttoi and Bengio, 1994]. Yet, SOM and NG are inspired by natural processes and consider additionally the neighborhood cooperation between the prototypes during learning.

Compressing the data by finding data representative prototypes often is only a preprocessing step. Tasks following afterwards are: clustering, visualization of the data set, and further analysis in various areas of application like image processing, pattern recognition and data mining. Thereby, clustering methods should group given *similar* data points such that data with related semantical meaning are linked together. A popular prototype based clustering method is Affinity Propagation (AP, [Frey and Dueck, 2007]). Yet, a big issue in unsupervised VQ is the evaluation and interpretation of *similarity*. Obviously, the evaluation is data-dependent as well as task-dependent. Therefore, clustering and visualization are ill-posed problems. Hence, for cluster solutions exist a lot of validity measures like the Dunn-Index [Dunn, 1973], CONN-index ([Taşdemir and Merényi, 2011], [T. Geweniger and Villmann, 2012]), Cohen's Kappa [Cohen, 1960] and many more ([Hashimoto et al., 2009], [Zalik and Zalik, 2011]). A lot of them like Dunn- and CONN-index rely on separation and compactness, i. e. it is assumed, that a *good* cluster solution

has compact clusters and the clusters are well separated from each other. Other evaluation measures are based on further criteria like on information theoretical aspects, e.g. Information Entropy [Bezdek, 1974]. Yet as mentioned before, an optimal cluster solution in general or the one and only measure for model evaluation does not exist.

In the field of supervised VQ, additionally to the data samples the class assignment of each data point is known. It has to be distinguished strictly between training, test and application phase. During the training, a classifier model is adapted on the basis of the training data set using the label information. The evaluation of the model is done on a test set, i. e. labeled data which are not used for training. In the application phase, the classifier model assigns each data point to a class to predict the labels for those new data.

Frequently, the classification error ($cerr$) or the classification accuracy ($acc$) is calculated for validation during training and test. Thereby, $cerr$ is the number of data points which are misclassified by the model and $acc$ the number of correct classified data points. However, other statistical quality measures for classification problems can also be used. A few of them are listed in Section 3. In general, a desirable classification model ends up with a high generalization ability and thus a minimal expected risk, i. e. high performance on data with unknown labels (application phase). Good results on the training set do not imply a good *generalization*. Roughly speaking, *generalization* means that a model remains valid for a test data set, i. e. it has similar performance like on the training set and avoids learning by heart. Statistical theory about the generalization ability, the Vapnik-Chervonenkis-theory (VC-theory), and how to measure the expected risk, can be found in [Vapnik, 1995]. An alternative description of the generalization ability of a classifier is the Rademacher and Gaussian complexity [Bartlett and Mendelson, 2002]. Although the classification task seems to be well defined, the goal and condition in several application methods differ, which leads to a large number of classifiers. A more detailed description of these aspects is addressed in Section 3.

Cluster methods may also be adapted to be used also for classification. Thereby, after uVQ and clustering of the prototypes, the prototypes are post-labeled. Note, a *good* cluster solution does not imply a good classification error or the other way around. Thus, beside the data samples the label information should be incorporated during the learning like in Supervised Neural Gas [Villmann et al., 2003] or in semi-supervised classification methods e. g. the Fuzzy Labeled Neural Gas (FLNG, [Villmann et al., 2006b]) or the Fuzzy Supervised Neural Gas (FSNG, [Kästner and Villmann, 2012]). In semi-supervised leaning both labeled and unlabeled data are given. The unlabeled data also contribute to build a model for improving the classification outcome.

Generally, all mentioned methods are based on similarities or dissimilarities between

the data. We define *distance* or *dissimilarity* measures as follows:

**Note 2.1.** In this work, we use the same denotation as given in `Pekalska&Duin` [Pekalska and Duin, 2005]. A *metric* $d : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}_+$ fulfills the following requirements for $x, y, z \in \mathbb{R}^D$

**reflexivity** $d(x, x) = 0$

**symmetry** $d(x, y) = d(y, x)$

**definiteness** $d(x, y) = 0 \Rightarrow x = y$

**triangle inequality** $d(x, y) + d(y, z) \leq d(x, z)$ .

If $d(x, y)$ is not a metric, we distinguish between *hollow*, *pre-*, *quasi* and *semi* metrics (see Tab. 2.1). Unless stated otherwise, the notations *distance* or *dissimilarity* measure refer to the hollow metric, i. e. only the reflexivity has to be fulfilled.

| | reflexivity | symmetry | definiteness | triangle equation |
|---|---|---|---|---|
| hollow metric | x | - | - | - |
| pre metric | x | x | - | - |
| quasi metric | x | x | x | - |
| semi metric | x | x | - | x |

**Table 2.1**: Different kinds of dissimilarity measures $d : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}_+$

In the following a few prototype based unsupervised, supervised and semi-supervised vector quantization methods will be introduced in their basic variants. Later on some of these will be extended and improved.

**Parts of the section are based on**

*T. Geweniger, L. Fischer, M. Kaden, M. Lange and T. Villmann:* Clustering by Fuzzy Neural Gas and Evaluation for Fuzzy Clusters. *Computational Intelligence and Neuroscience, 2013.*

## 2.1 Unsupervised Vector Quantization

Prototype based uVQ uses a set of prototypes $W = \{\mathbf{w}_1, \ldots, \mathbf{w}_{N_W}\} \in \mathbb{R}^D$ to represent a vectorial data set $V = \{\mathbf{v}_1, .., \mathbf{v}_{N_V}\} \in \mathbb{R}^D$. In crisp VQ, an index mapping function $\Upsilon : I_V = \{1, \ldots, N_V\} \longmapsto I_W = \{1, \ldots, N_W\}$ maps the index $i$ of each data point $\mathbf{v}_i$ to an index $j$ of a prototype $\mathbf{w}_j$. In fuzzy VQ, this index mapping is not unique, i. e. a data point $\mathbf{v}_i$ can be assigned to several prototypes (cf. page 29). First, we consider crisp VQ. Each data point $\mathbf{v}_i$ is assigned to a prototype by the *winner-takes-all* (WTA) rule:

$$i \longmapsto j = s(\mathbf{v}_i) = \underset{k \in I_W}{\operatorname{argmin}} \left( d(\mathbf{v}_i, \mathbf{w}_k) \right) \ , \tag{2.1}$$

such that $s(\mathbf{v}_i)$ refers to the index of the best matching prototype in terms of the distance function $d(\mathbf{v}_i, \mathbf{w}_j)$. The distance $d(\mathbf{v}_i, \mathbf{w}_j)$ describes the dissimilarity between $\mathbf{v}_i$ and $\mathbf{w}_j$.

Often $d(\mathbf{v}, \mathbf{w})$ is based on the Euclidean norm

$$d_E(\mathbf{v}, \mathbf{w}) = ||\mathbf{v} - \mathbf{w}||_E = \sqrt{\sum_{k=1}^{D} (v_k - w_k)^2} \tag{2.2}$$

or the squared Euclidean distance

$$d_E^2(\mathbf{v}, \mathbf{w}) = \left(d_E(\mathbf{v}, \mathbf{w})\right)^2 = \sum_{k=1}^{D} (v_k - w_k)^2. \tag{2.3}$$

It has to be mentioned that the squared Euclidean distance not a metric, i. e. it is a quasi metric (see Note 2.1). Furthermore, any other dissimilarity measure such as the Sobolev distances, divergences, the $l_p$ distances, or kernel distances can be applied depending on the applications and the VQ model. We will discuss deeper details about distance or dissimilarity measures in Section 2.4.

The WTA rule (2.1) realizes a partition of the data space into Voronoi cells, denoted as receptive fields in prototype based VQ. Each receptive field

$$R_j(V, W) = \{\mathbf{v} \in V | d(\mathbf{v}, \mathbf{w}_j) \leq d(\mathbf{v}, \mathbf{w}_k) \ \forall j \neq k\} \tag{2.4}$$

includes all data points assigned to prototype $\mathbf{w}_j$ in accordance to (2.1). Properties of

the receptive fields[1] are:

$$\bigcup_{k=1}^{N_W} R_k(V,W) = \mathbb{R}^D \quad \text{and} \quad \mu\left(\bigcap_{k=1}^{N_W} R_k(V,W)\right) = 0$$

where math measure $\mu(M) = 0$ implies that the set $M$ is of the measure zero.

As already mentioned, the goal of uVQ is, among others, to minimize the description error. This can be modeled by the following general continuous cost function:

$$E_{uVQ} = \int P(\mathbf{v}) \cdot lc\,(\mathbf{v}, W)\ d\mathbf{v} \tag{2.5}$$

where $P(\mathbf{v})$ is the data density and $lc\,(\mathbf{v}, W)$ are the *local costs*. Thereby, each method has its own specific local cost function. If it is set to $lc\,(\mathbf{v}, W) = d_E^2(\mathbf{v}, \mathbf{w}_{s(\mathbf{v})})$, $E_{uVQ}$ describes the common quadratic Euclidean description error.

Unsupervised VQ is an unconstrained, in general non-linear and non-convex optimization problem. The *stochastic gradient descent* (SGD) strategy is a well established method solving this kind of cost function based optimization problems. The SGD is applied on several other cost functions mentioned in this work. The selection of this optimization strategy has an historical background. A more detailed description of SGD can be found in appendix A.1. In the following, in one *epoch* $\iota$ of SGD all data are considered once in a random sequence. The general update in uVQ of the prototypes $\mathbf{w}_j$ can be determined by:

$$\mathbf{w}_j \leftarrow \mathbf{w}_j - \alpha \cdot \Delta\mathbf{w}_j \tag{2.6}$$

$$\Delta\mathbf{w}_j \sim \frac{\partial\, lc\,(\mathbf{v}, W)}{\partial\, \mathbf{w}_j}, \tag{2.7}$$

where $0 < \alpha \lll 1$ is the learning rate[2]. The SGD is performed until convergence, i. e. the change of the outcome of the cost function is smaller than a predisposed value, or until a predefined number of epochs is reached.

For the data compression task, the number of prototypes has to be very small compared to the number of data points, i. e. $N_W \lll N_V$. The existing cost function based VQ methods differ in the definition of the local costs. Several models are motivated by neural data processing in the brains whereas other are inspired by physics.

[1] We assume data densities without Dirac impulses.

[2] The parameter $\mathbf{w}_j$ is a vector. The gradient $\frac{\partial d(\mathbf{v},\mathbf{w}_j)}{\partial \mathbf{w}_j}$ is a shorthand notation for a vector with entries of the single derivatives $\frac{\partial d\left([\mathbf{v}]_l,[\mathbf{w}_j]_l\right)}{\partial [\mathbf{w}_j]_l}$, $l = 1, \ldots, D$.

### 2.1.1 C-means

Many advanced methods are based on the *c-means* method introduced first by `MacQueen` [MacQueen, 1967]. The goal of *c-means* is to partition the data space into $c$ clusters such that the sum of the distances of the data samples to the cluster means is minimal. The local costs in (2.5) are defined by

$$lc^{CM}(\mathbf{v}, W) = d(\mathbf{v}, \mathbf{w}_{s(\mathbf{v})}), \tag{2.8}$$

where in the original work of `MacQueen` $d(\mathbf{v}, \mathbf{w})$ is the squared Euclidean distance. The learned prototypes $\mathbf{w}_j$ can interpreted as centroid in the contentious and as cluster means in the discrete case. The number of prototypes, i. e. the number of clusters $c$, has to be chosen in advance.

Finding the minimum of the cost function (2.5) with the local costs (2.8) is *NP*-hard. As already mentioned, the SGD can be applied to find a local minimum of the cost function. In each iteration step, where one randomly chosen data point $\mathbf{v}$ is considered, only the best matching prototype $\mathbf{w}_{s(\mathbf{v})}$ is adapted. The update of prototype $\mathbf{w}_{s(\mathbf{v})}$ is obtained by the partial derivatives of the cost function according to $\mathbf{w}_{s(\mathbf{v})}$:

$$\mathbf{w}_{s(\mathbf{v})} \leftarrow \mathbf{w}_{s(\mathbf{v})} - \alpha \cdot \frac{\partial\, d\left(\mathbf{v}, \mathbf{w}_{s(\mathbf{v})}\right)}{\partial\, \mathbf{w}_{s(\mathbf{v})}}\,,$$

where $0 < \alpha \lll 1$ is the learning rate and has to be decreased during the learning process.

Recent studies for c-means replace the squared Euclidean distances $d(\mathbf{v}, \mathbf{w})$ by further distance measures [Karayiannis and Randolph-Gips, 2003]. Yet, the distance function has to be differentiable according to the parameters, i. e. to the prototypes.

Although, CM is known to find only local optima [Bottou and Bengio, 1994], it is widely used. To overcome this locality behavior, the algorithm is usually performed several runs with different initializations and the best achieved model is taken. Another problem of CM is the occurrence of dead units. Dead units are never winning prototypes $\mathbf{w}_j$, i. e. $j \neq s(\mathbf{v})\ \forall \mathbf{v} \in V$. An alternative to assure a faster convergence, avoid local optima and dead units is to include local neighborhood cooperativeness between the prototypes as outlined in the following.

### 2.1.2 Self-Organizing Map

The Self-Organizing Map (SOM) is introduced for efficient VQ inspired by cortical maps as well as data visualization by `Kohonen` ([Kohonen, 1982], [Kohonen, 1998]). The neighborhood cooperativeness in `Kohonen`-SOM is inspired by cortical sensory maps in the human brain, i. e. similar stimulus mapped spatially to neighbored neuronal structures. This principle is modeled by topology preserving mapping from a D-

dimensional data space $V$ onto a low-dimensional simple geometric grid $A$ (typically two or three dimensions). Thereby, each prototype $\mathbf{w_r} \in \mathbb{R}^D$ is assigned to a node also denoted as *neuron* $\mathbf{r} \in A$. Generally, the geometric grid $A$ is fixed and the prototypes $\mathbf{w_r}$ are adapted according to the neighborhood cooperativeness in $A$.

A lot of modifications, extensions and mathematical analyses of the Kohonen-SOM are published ([Heskes, 1999], [Vesanto, 1999], [Kaski et al., 1998], [Kohonen, 2013]). The original SOM by Kohonen is not based on a cost function [Erwin et al., 1992]. One fundamental improvement was presented by Heskes [Heskes, 1999] who established a modified SOM based on a cost function keeping the main idea of Kohonen-SOM neighborhood cooperativeness. The cost function introduced by Heskes has the same structure as (2.5). However, the local costs are defined via:

$$lc^{SOM}\left(\mathbf{v}, W, \sigma\right) = \sum_{\mathbf{r}' \in A} h_\sigma^{SOM}\left(\hat{\mathbf{s}}\left(\mathbf{v}\right), \mathbf{r}'\right) d\left(\mathbf{v}, \mathbf{w}_{\mathbf{r}'}\right), \tag{2.9}$$

taking into account the topological structure of the external grid $A$. In opposite to the WTA-rule (2.1), the winner in (2.9) is determined by:

$$\hat{\mathbf{s}}\left(\mathbf{v}\right) = \mathrm{argmin}_{\mathbf{r} \in A}\left(\sum_{\mathbf{r}' \in A} h_\sigma^{SOM}\left(\mathbf{r}, \mathbf{r}'\right) d\left(\mathbf{v}, \mathbf{w}_{\mathbf{r}'}\right)\right) \tag{2.10}$$

depending on the neighborhood function

$$h_\sigma^{SOM}\left(\mathbf{r}, \mathbf{r}'\right) = \exp\left(-\frac{d_A\left(\mathbf{r}, \mathbf{r}'\right)}{2\sigma^2}\right) \tag{2.11}$$

evaluate in $A$. The value $\sigma > 0$ determines the influence range in the lattice. It has to be pointed out: the neighborhood function $h_\sigma^{SOM}$ and, therefore, the winner determination $\hat{\mathbf{s}}\left(\mathbf{v}\right)$ take into account the distance $d_A\left(\mathbf{r}, \mathbf{r}'\right)$ of the neurons in the external lattice $A$. The distance $d_A$ depends on the type of A, e.g. in general, if the grid is rectangular, $d_A$ is the Manhattan metric or if A is hexagonal, $d_A$ is chosen as Euclidean distance.

The optimization of the cost function (2.5) with the local costs (2.9) by SGD leads to the learning rule

$$\triangle\, \mathbf{w_r} \sim -h_\sigma^{SOM}\left(\mathbf{r}, \hat{\mathbf{s}}\left(\mathbf{v}\right)\right)\ \frac{\partial d\left(\mathbf{v}, \mathbf{w_r}\right)}{\partial \mathbf{w_r}}. \tag{2.12}$$

It should be emphasized that this derivation of the gradient descent learning is only valid iff the local costs $lc^{SOM}\left(\mathbf{v}, W, \sigma\right)$ in the cost function (2.5) are exactly the same as those used for the winner mapping in (2.10) [Villmann, 2006].

Besides compressing and clustering the information while preserving the basic topological and metric relations, the main virtue of the SOM is its visualization ability. The clustering solution becomes visible with the projection $\Upsilon^{SOM} : I_V \longmapsto A$. An overview

of visualizations methods are given in [Vesanto, 1999]. Two well-established visualization tools for SOM models with a two dimensional lattice $A$ are:

**Component planes** A component plane coded for each neuron in this lattice is the value of one dimension or also called component $[\mathbf{w_r}]_k$. This value is coded by a color scale or gray levels. Thus, the number of component planes is identical with the number of dimensions. The component planes may give a hint whether the SOM model is topology preserving [Vesanto, 1999, Villmann, 2006]. An example is depicted in 2.1.

**U-matrix** The U-matrix was developed by `Ultsch` [Ultsch and Semon, 1990] and is a distance matrix based on the SOM-lattice. For each neuron $\mathbf{r}$ the direct neighbors in the grid taken as graph are determined $NN(\mathbf{r})$, e. g. in the rectangular lattice these are four for the neurons inside the lattice and, accordingly fewer for neurons at the border. In the U-matrix the distance $d(w_\mathbf{r}, w_{\mathbf{r}'})$ is visualized by using a gray level scale or component planes, respectively. Thus, the size of the u-matrix for a rectangular lattice with the size $a \times b$ is $(2 \cdot a - 1)(2 \cdot b - 1)$. A generalization thereof is: each neuron is attached by a third dimension called the u-high. This u-high codes the mean distance of a neuron to its direct neighbors:

$$u(\mathbf{r}) = \frac{1}{|NN(\mathbf{r})|} \sum_{\mathbf{r}' \in NN(\mathbf{r})} d(\mathbf{w_r}, \mathbf{w_{r'}})$$

The u-matrix is a powerful tool for the interpretation of the resulting model. It also can be used to determine the number of clusters [Ultsch and Semon, 1990]. An exemplified U-matrix is shown in Fig. 2.2.

Crucial issues concerning the structure of the SOM-lattice are:

- number of nodes
- neighborhood structure and
- shape of the grid.

These questions are data depending and there exists no general answer. The resulting model should be checked, whether the SOM solution is topology preserving [Villmann, 2006]. If the model is topology preserving, the result should be used for further analyses and conclusions. Thus, the fixed prior chosen lattice causes topological restrictions and therefore, can induce difficulties in applications. A further method with natural visualization ability which also relies a data depended lattice (not fixed beforehand) is the *Topology Representing Network*. A description thereof can be found in [Martinetz and Schulten, 1994]. A further alternative is the *Neural Gas*, which is explicated in the next part.

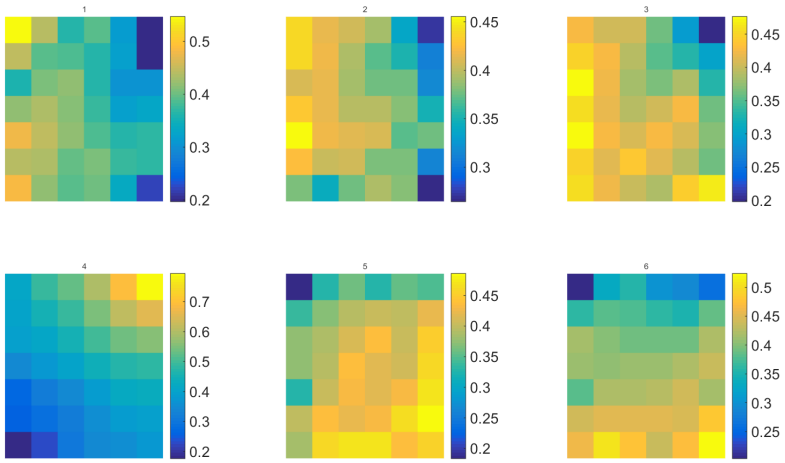


**Figure 2.1**: Component planes for the six features of the Colorado data set (see page 8). The rectangular grid size is $6 \times 7$.The component planes show a good behavior of the SOM-model. This is also reflected in the topographic product of $-8.7 \cdot 10^{-4} \approx 0$ [Villmann, 2006].

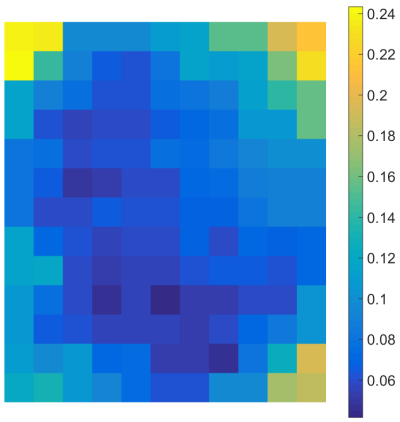


**Figure 2.2**: U-matrix of the Colorado data set is based on the lattice $6 \times 7$.

### 2.1.3 Neural Gas

The Neural Gas (NG) was introduced by Martinetz, Berkovich, & Schulten and is inspired by the SOM [Martinetz et al., 1993]. In Neural Gas a neighborhood cooperativeness between prototypes is also integrated in the local cost to speed up the convergence compared to CM. Yet, the NG neighborhood function $h_\sigma^{NG}$ is evaluated in the input space $V$ and not on an external grid like in SOM. More precisely, $h_\sigma^{NG}$ depends on the *winner rank function*

$$rk_j\left(\mathbf{v}, W\right) = \sum_{k=1}^{N_W} \mathtt{H}\left(d\left(\mathbf{v}, \mathbf{w}_j\right) - d\left(\mathbf{v}, \mathbf{w}_k\right)\right) \tag{2.13}$$

where

$$\mathtt{H}\left(x\right) = \begin{cases} 0 & if \quad x \leq 0 \\ 1 & else \end{cases} \tag{2.14}$$

is the Heaviside function. The winning prototype $\mathbf{w}_{s(\mathbf{v})}$ has the rank $rk_{s(\mathbf{v})}(\mathbf{v}, W) = 0$, the second closest prototype $\mathbf{w}_j$ the rank $rk_j(\mathbf{v}, W) = 1$ and so one. The neighborhood function yield to:

$$h_\sigma^{NG}\left(rk_j\left(\mathbf{v}, W\right)\right) = \exp\left(-\frac{rk_j\left(\mathbf{v}, W\right)}{2\sigma^2}\right) \tag{2.15}$$

with the neighborhood range $\sigma \geq 0$. For $\sigma > 0$, beside the winning prototype $\mathbf{w}_{s(\mathbf{v})}$ also the prototypes of higher ranks are updated. In general, $\sigma$ slowly decreased to zero during learning [Martinetz et al., 1993].

The local costs are:

$$lc^{NG}(\mathbf{v}, W, \sigma) = \sum_{j=1}^{N_W} h_\sigma^{NG}\left(rk_j\left(\mathbf{v}, W\right)\right)\ d\left(\mathbf{v}, \mathbf{w}_j\right) \tag{2.16}$$

and the cost function of the neural gas yields to:

$$E_{\mathrm{NG}}(V, W) = \frac{1}{C(\sigma)} \int P\left(\mathbf{v}\right) lc^{NG}(\mathbf{v}, W, \sigma)\ d\mathbf{v} \tag{2.17}$$

with data density $P\left(\mathbf{v}\right)$ and the normalization constant $C(\sigma) = \sum_{j=1}^{N_W} h_\sigma^{NG}\left(rk_j\right)$. The cost function corresponds to the energy function of a gas with the particles $\mathbf{w}$, the potential $V$ and the viscosity $\sigma$.

The learning takes place as stochastic gradient descent on $E_{NG}$ according to

$$\mathbf{w}_j \leftarrow \mathbf{w}_j - \alpha\, \Delta\mathbf{w}_j \tag{2.18}$$

$$\Delta\mathbf{w}_j \sim h_\sigma^{NG}\left(rk_j\left(\mathbf{v}, W\right)\right) \cdot \frac{\partial d\left(\mathbf{v}, \mathbf{w}_j\right)}{\partial \mathbf{w}_j} \tag{2.19}$$

where $\alpha > 0$ is the learning rate. During the adaptation process the shifting of the prototypes can be interpreted as the dynamic of the diffusing particles moving in a potential determined by the data density [Martinetz et al., 1993].

In a nutshell, the NG is an efficient tool for uVQ. Like SOM the NG takes also the neighborhood cooperativeness into account. Thus, the method is not sensitive to the initialization like c-means and therefore is robust. Further, due to the neighborhood cooperatives dead units are avoided. Compared to SOM, an advantage is the abolition of the topological restriction in the neighborhood cooperativeness because of the winner ranks. Therefore, Neural Gas is a good choice for unsupervised vector quantization problems. Yet, NG does not provide a feasible possibility for visualization.

### 2.1.4 Common Generalizations

So far only the basic prototype based uVQ methods were explained. Yet, there exist a lot of extensions and modifications. A few of them are summarized in this section and exemplified on the Neural Gas method.

#### Batch Variants

In the last section only the *online* variants of the uVQ were presented, i. e. during each prototype update step only one data point is taken into account. The notation *batch* indicate another optimization strategy. Though, for an averaged update of the prototypes all data points are taken into account in one iteration step and therefore, all data points have to be presented in advance. In general, an alternating optimization scheme adapted from the Expectation-Maximization strategy (EM, [Bishop, 2006]) is applied and in each iteration step prototypes are set to an weighting average over the data samples (Batch c-means [Ball and Hall, 1967], Batch SOM [Cheng, 1997], Batch NG [Cottrell et al., 2006]).

**Batch NG** For Batch NG the discrete form of the NG cost function is defined by

$$E_{\mathrm{NG}_{disc}}(V, W) = \frac{1}{C(\sigma)} \sum_{i=1}^{N_V} lc^{NG}(\mathbf{v}_i, W, \sigma) \tag{2.20}$$

with the local costs $lc^{NG}(\mathbf{v}_i, W, \sigma)$ from (2.16). The ranks are interpreted as hidden variables $k_{ji} = rk_j(\mathbf{v}_i, W)$ where the vectors $\mathbf{k}_i = \{k_{ji} \in \{0, \ldots, N_W - 1\} | \mathbf{v}_i \in V\}$ constitute a permutation of the set $\{0, \ldots N_W - 1\}$ and includes all winner ranks. Thus, the new parameters of the cost function (2.20) are $\mathbf{k}_i$ and $\mathbf{w}_j$. The optimization of $E_{\mathrm{NG}_{disc}}(V, W)$ might be done by an alternating optimization strategy [Bishop, 2006], which consist of the following two alternating adaptation steps:

**step A** Calculate the ranks of $\mathbf{w}_j$ for all data points:

$$k_{ji} = \sum_{k=1}^{N_W} \mathtt{H}\left(d\left(\mathbf{v}_i, \mathbf{w}_j\right) - d\left(\mathbf{v}_i, \mathbf{w}_k\right)\right)$$

**step B** Fix the parameter vector $\mathbf{k}_i$. Update prototype $\mathbf{w}_j$ by the weighted average:

$$\mathbf{w}_j = \frac{\sum_{k=1}^{N_V} h_\sigma^{NG}\left(k_{jk}\right) \cdot \mathbf{v}_i}{\sum_{k=1}^{N_V} h_\sigma^{NG}\left(k_{jk}\right)} \tag{2.21}$$

Step B can be interpreted as a second order optimization according to Newton optimization. The derivation of the formula (2.21) can be found in [Cottrell et al., 2006]. It has to be mentioned that this batch variant is formulated for the squared Euclidean distance. The distance $d_E^2$ is implicitly involved in (2.21).

The Batch SOM is analog to the Batch NG, only the neighborhood function and the winner determination is changed accordingly [Cheng, 1997]. Usually, only a few adaptation steps are necessary for convergence in the batch variants [Cottrell et al., 2006].

In general, the batch variants are faster in convergence especially on high dimensional data, because only one adaptation is necessary in each epoch. However, the prototypes might get stuck in local minima and in particularly in Batch SOM, the problem of topological mismatches and the dependency on the initialization is much more pronounced as for online learning [Fort et al., 2001].

#### Median and Relational Variants

In some fields of application like in the analysis of protein structures and text documents, the data are not given in the standard Euclidean vector space. Instead, a discrete representation of the data or even only a (dis-)similarity matrix are available. In these cases the standard shifting (2.6) of the prototypes is not possible due to the derivations of $d(\mathbf{v}, \mathbf{w})$ do not exist.

In the median variants the prototypes are restricted to be data points, i. e. an index mapping $\Psi : l \in \{1, \ldots N_W\} \longmapsto \tilde{l} \in \{1, \ldots N_V\}$ is learned (Median NG [Cottrell et al., 2006], Median SOM [Kohonen and Somervuo, 2002]). Hence, only the dissimilarity matrix $D \in \mathbb{R}_+^{N_V \times N_V}$ of the data points has to be given. Moreover, $D$ has to be a quasi metric [3][Cottrell et al., 2006].

Extensions of the median variants are the relational methods, where the prototypes are assumed to be linear combination of the data points $\mathbf{w} = \sum_{i=1}^{N_V} \alpha_i \mathbf{v}_i$. Thus, the linear factors $\alpha_i$ are adjusted during the learning (Relational NG [Cottrell et al., 2006],

[3]The dissimilarity measure $D$ can be embedded in a pseudo-Euclidean vector space (see Note 2.1 or [Pekalska and Duin, 2005]).

Relational SOM [Hasenfuss and Hammer, 2007]). Like in the median variants, only the dissimilarity matrix $D$ has to be given and which has to be a quasi metric [Hasenfuss, 2009].

**Median NG** In Median NG ([Cottrell et al., 2006]) a prototype $\mathbf{w}_l$ is mapped on a data point $\mathbf{v}_{\tilde{l}}$ with $d_{i\tilde{l}} = d(\mathbf{v}_i, \mathbf{v}_{\tilde{l}})$. To learn this index mapping step B for Batch NG (2.21) is replaced by:

$$\tilde{l} \leftarrow \underset{l' \in \{1,\ldots,N_V\}}{\operatorname{argmin}} \sum_{i=1}^{N_V} h_\sigma^{NG}(k_{li})\, d_{i,l'}$$

with the fixed vector $\mathbf{k}_i = \{k_{li} \in \{0, \ldots, N_W - 1\} | \mathbf{v}_i \in V\}$.

### Fuzzy Variants

Until now the assignment of a data point $\mathbf{v}_i$ to a prototype $\mathbf{w}_j$ is unique and crisp. Let, $U \in \mathbb{R}_+^{N_V \times N_P}$ be an assignment matrix and in fuzzy variants of uVQ (fuzzyVQ) these assignments $u_{ij}$ can be possibilistic $u_{ij} \in [0, 1]$ or probabilistic ($u_{ij} \geq 0$ and $\sum_{j=1}^{N_W} u_{ij} = 1$). For crisp VQ yields $u_{ij} = 1$ if $\mathbf{v}_i$ is mapped to $\mathbf{w}_j$ and $u_{ij} = 0$ otherwise.

The discrete cost function of the fuzzyVQ variant has the form:

$$E_{fuzzyVQ} = \sum_{i=1}^{N_V} \sum_{j=1}^{N_W} u_{ij}^m \cdot lc(\mathbf{v}_i, \mathbf{w}_j, W) \tag{2.22}$$

where $lc(\mathbf{v}_i, \mathbf{w}_j, W)$ are the local costs and $m$ is the *fuzzyness* parameter usually set between $1.2 \leq m \leq 2$ [Bezdek, 1980]. If $m \to 1$, the assignments converge to the crisp solution and for $m \to \infty$ a unique distribution is forced [Bezdek, 1980]. For fuzzyVQ variants, both the position of the prototypes and the assignments $u_{ij}$ have to be adapted. Frequently, the learning of both parameters is done by alternating optimization [Bezdek and Hathaway, 2003]. This optimization principle is similar to the batch variants.

The Fuzzy-C-Means (FCM) was proposed by `Dunn` [Dunn, 1973] and elaborately discussed and improved by `Bezdek` [Bezdek, 1980]. Later, the idea of fuzzyness was transfered to the SOM (FSOM [Tsao et al., 1994]) and NG (FNG [Geweniger et al., 2013]). Again, we specify the basic principle for NG.

**Fuzzy Neural Gas** The cost function of the Fuzzy Neural Gas (FNG) yields to:

$$E_{FNG} = \sum_{i=1}^{N_V} \sum_{j=1}^{N_W} u_{ij}^m \cdot lc^{FNG}(\mathbf{v}_i, \mathbf{w}_j, W, \sigma) \tag{2.23}$$

with $lc^{FNG}$ similar to the local costs of the original NG (2.16):

$$lc^{FNG}(\mathbf{v}, \mathbf{w}_j, W, \sigma) = \sum_{l=1}^{N_W} h_\sigma^{FNG}\left(rk_j\left(\mathbf{w}_l, W\right)\right)\ d_E^2\left(\mathbf{v}, \mathbf{w}_j\right). \tag{2.24}$$

Yet, the rank is determined according to the best matching prototype and not according to the data points:

$$h_\sigma^{FNG}\left(rk_j\left(\mathbf{w}_l, W\right)\right) = \exp\left(-\frac{rk_j\left(\mathbf{w}_j, W\right)}{2\sigma^2}\right) \tag{2.25}$$

is the neighborhood function of the FNG with the winner rank function

$$rk_j\left(\mathbf{w}_l, W\right) = \sum_{k=1}^{N_W} \mathtt{H}\left(d_E^2\left(\mathbf{w}_l, \mathbf{w}_j\right) - d_E^2\left(\mathbf{w}_l, \mathbf{w}_k\right)\right). \tag{2.26}$$

Analog to Batch NG, the optimization of the FNG cost function with the parameters prototypes $\mathbf{w}$ and fuzzy assignments $u_{ij}^m$ instead of the hidden variables $\mathbf{k}_j$ is based on the alternating optimization strategy. Thus, we obtain the following steps, which have to be processed alternatingly:

**step A** Calculate the fuzzy assignments $u_{ij}$ for all data points:

$$u_{ij} = \frac{1}{\sum_{l=1}^{N_W}\left(lc^{FNG}(\mathbf{v}, \mathbf{w}_j, W, \sigma)\right)^{1-m} \cdot \left(lc^{FNG}(\mathbf{v}, \mathbf{w}_l, W, \sigma)\right)^{m-1}}$$

**step B** Fix the fuzzy assignments $u_{ij}$ and update prototype $\mathbf{w}_j$: Using the squared Euclidean distance the update equation can be formulated as:

$$\mathbf{w}_j = \frac{\sum_{i=1}^{N_V}\sum_{l=1}^{N_W} u_{il}^m \cdot h_\sigma^{FNG}\left(rk_j\left(\mathbf{w}_l, W\right)\right) \cdot \mathbf{v}_i}{\sum_{i=1}^{N_V}\sum_{l=1}^{N_W} u_{il}^m \cdot h_\sigma^{FNG}\left(rk_j\left(\mathbf{w}_l, W\right)\right)} \tag{2.27}$$

A more detailed description and the proof of convergence can be found in [Geweniger et al., 2013].

## 2.2 Supervised Vector Quantization

As already mentioned, for sVQ beside the training samples $\mathbf{v} \in V \subset \mathbb{R}^D$ also the class assignments $c(\mathbf{v}) \in \mathcal{C} = \{1, \ldots, N_C\}$ are given. The task is to find a model which assign a data point $\mathbf{v}$ to a predicted label $\hat{c}(\mathbf{v}) \in \mathcal{C}$ under the aspect of correctness, i. e. good classification performance.

A common measure to evaluate the classification performance is the classification accuracy $acc(V, W)$ or the classification error $cerr(V, W)$, respectively. The accuracy is determined by the relative number of data points which are classified correctly by the model, i. e.:

$$acc(V, W) = \frac{1}{N_V} \sum_{\mathbf{v} \in V} \delta_{c(\mathbf{v}), \hat{c}(\mathbf{v})} \tag{2.28}$$

where $\delta_{i,j}$ is the Kronecker delta with

$$\delta_{i,j} = \begin{cases} 1 & , \text{ if } i = j \\ 0 & , \text{ else} \end{cases} \tag{2.29}$$

The classification error is simply $cerr(V, W) = 1 - acc(V, W)$.

Two well known prototype based classifier models are the Support Vector Machine (SVM, [Schölkopf and Smola, 2002]) and the Learning Vector Quantization (LVQ, [Kohonen, 1998, Sato and Yamada, 1996]). The SVM is an extension of a hyperplane classifier to solve non-linear separable binary classification problems. The hyperplane is described by the prototypes, called Support Vectors, and therefore, the prototypes are class border typical. The cost function of the SVM maximizes the margin between two classes and is model as quadratic optimization problem with convex constraints. It has to be pointed out that the SVM is a binary classifier. Extensions to multi-class problems are only based on heuristics. A more detailed description can be found in the Appendix A.2 or in [Schölkopf and Smola, 2002].

The Learning Vector Quantizers are well established Bayesian classifiers with an intuitive learning principle. A detailed description of the LVQ methods follows.

### 2.2.1 The Family of Learning Vector Quantizers - LVQ

The basic method LVQ 1 was introduced by `Kohonen` with the goal to approximate a *Bayes* classification scheme in an intuitive manner [Kohonen, 1986]. At the beginning of the training phase data points $\mathbf{v} \in V$ with their labels $c(\mathbf{v})$ and at least one initialized prototype $\mathbf{w}$ per class with $y(\mathbf{w}) \in \mathcal{C}$ are given. LVQ 1 is based on the Hebbian learning scheme and is an iterative method. In one iteration step a data point $\mathbf{v}$ is chosen randomly and the nearest prototype, the winner $\mathbf{w}_{s(\mathbf{v})}$, is determined using the squared

Euclidean distance $d_E^2$ (2.1). The update of the winner $\mathbf{w}_{s(\mathbf{v})}$

$$\mathbf{w}_{s(\mathbf{v})} \longleftarrow \mathbf{w}_{s(\mathbf{v})} - \alpha \cdot \Delta\mathbf{w}_{s(\mathbf{v})} \tag{2.30}$$

depends on the label matching

$$\Delta\mathbf{w}_{s(\mathbf{v})} = (-1)^{1+\delta_{c(\mathbf{v}),y(\mathbf{w}_{s(\mathbf{v})})}} \cdot (\mathbf{v} - \mathbf{w}_{s(\mathbf{v})})$$

where $0 < \alpha \ll 1$ is a decreasing learning rate. The adjusting of the prototypes follows the intuitive principle of *attraction and repulsing*:

**attraction** If the winning prototype $\mathbf{w}_{s(\mathbf{v})}$ is of the same class as the presented data point $\mathbf{v}$, i. e. $c(\mathbf{v}) = y(\mathbf{w}_{s(\mathbf{v})})$, $\mathbf{w}_{s(\mathbf{v})}$ will be attracted due to: $(-1)^{1+\delta_{c(\mathbf{v}),y(\mathbf{w}_{s(\mathbf{v})})}} = 1$.

**repulsing** If the label of the winning prototype $\mathbf{w}_{s(\mathbf{v})}$ is different compared to $c(\mathbf{v})$, the prototype will be pushed away, i. e. $(-1)^{1+\delta_{c(\mathbf{v}),y(\mathbf{w}_{s(\mathbf{v})})}} = -1$.

At the end of the training, the prototypes should describe their classes as precise as possible.

The assignment of a data point with an unknown label is done by nearest prototype classification (NPC):

$$NPC: \quad y(\mathbf{w}_{s(\mathbf{v})}) \mapsto \hat{c}(\mathbf{v}). \tag{2.31}$$

The simple principle of LVQ is very effective in a lot of applications ([Umer and Khiyal, 2007], [Bashyal and Venayagamoorthy, 2008]). However, there are some disadvantages, like slow convergence or instability in some cases: If prototypes and data points are in an inappropriate constellation e. g. unbalanced data or unfavorably initialized, the prototypes might be pushed away always. Therefore, some improvements are developed, e. g. LVQ 2.1 and LVQ 3 [Kohonen, 1998].

**LVQ 2.1** For LVQ 2.1 beside the winning prototype $\mathbf{w}_{s(\mathbf{v})}$ also the second best matching prototype $\mathbf{w}_{s_2(\mathbf{v})}$ with

$$s_2(\mathbf{v}) = \underset{k \in \{1,\ldots,N_W\}, k \neq s(\mathbf{v})}{\operatorname{argmin}} d(\mathbf{v}, \mathbf{w}_k) \tag{2.32}$$

is considered. Thereby, the update of these prototypes has to be distinguished into three cases:

**case I** $c(\mathbf{v}) = y(\mathbf{w}_{s(\mathbf{v})}) = y(\mathbf{w}_{s_2(\mathbf{v})})$: No update is performed because of the robust decision.

**case II** $c(\mathbf{v}) \neq y(\mathbf{w}_{s(\mathbf{v})})$ and $c(\mathbf{v}) \neq y(\mathbf{w}_{s_2(\mathbf{v})})$: Both prototypes are pushed away

$$\begin{aligned} \Delta\mathbf{w}_{s(\mathbf{v})} &= -\alpha \cdot (\mathbf{v} - \mathbf{w}_{s(\mathbf{v})}) \\ \Delta\mathbf{w}_{s_2(\mathbf{v})} &= -\alpha \cdot (\mathbf{v} - \mathbf{w}_{s_2(\mathbf{v})}) \,. \end{aligned}$$

**case III** $k, l \in \{s(\mathbf{v}), s_2(\mathbf{v})\}$ and $k \neq l$: If $c(\mathbf{v}) = y(\mathbf{w}_k) \neq y(\mathbf{w}_l)$ and both prototypes are located within the *window*

$$\min\left(\frac{(\mathbf{v}-\mathbf{w}_l)^2}{(\mathbf{v}-\mathbf{w}_k)^2}, \frac{(\mathbf{v}-\mathbf{w}_k)^2}{(\mathbf{v}-\mathbf{w}_l)^2}\right) > \frac{1-\omega}{1+\omega}\,, \tag{2.33}$$

the prototype $\mathbf{w}_k$ is attracted and $\mathbf{w}_l$ is pushed away

$$\begin{aligned} \Delta\mathbf{w}_k &= +\alpha \cdot (\mathbf{v} - \mathbf{w}_k) \\ \Delta\mathbf{w}_l &= -\alpha \cdot (\mathbf{v} - \mathbf{w}_l) \,. \end{aligned}$$

The window parameter $0 < \omega \leq 1$ effects the number of training samples taking into account during the learning. A good choice for $\omega$ is $0.2$ to $0.3$ [Kohonen, 1998]. The window rule assures that only prototypes around the decision border are considered, without this rule the prototypes may diverge.

The LVQ 2.1 should be applied subsequently to LVQ 1 and run for only a few iteration steps, because it may cause instable behavior [Kohonen, 1998]. If we consider only one prototype per class, LVQ 1 tries to arrange the prototypes in the class centers. Otherwise, LVQ 2.1 is more class border sensitive, because of window rule (2.33).

**LVQ 3** The variant LVQ 3 is based on LVQ 2.1 and only differs in *case I*. Hereby, both prototypes are attracted:

$$\begin{aligned} \Delta\mathbf{w}_{s(\mathbf{v})} &= \alpha \cdot (\mathbf{v} - \mathbf{w}_{s(\mathbf{v})}) \\ \Delta\mathbf{w}_{s_2(\mathbf{v})} &= \alpha \cdot (\mathbf{v} - \mathbf{w}_{s_2(\mathbf{v})}) \end{aligned}$$

The LVQ 3 should only be applied in addition to LVQ 1 like for LVQ 2.1

All mentioned heuristics and combinations thereof typically have a good performance in practical applications. However, they are only heuristics and, therefore, no predictions about the learned models and local or global optima are possible. An extended version is the Generalized Learning Vector Quantization which is based on a cost function and, therefore, is mathematically more sound.

### 2.2.2 Generalized Learning Vector Quantization

The Generalized Learning Vector Quantization (GLVQ) was introduced by `Sato and Yamada` in 1995 [Sato and Yamada, 1996]. The goal of the GLVQ method is to minimize the classification error by keeping the principle of prototype learning known from LVQ. For that purpose, `Sato and Yamada` introduced the *classifier function*:

$$\mu_W(\mathbf{v}) = \frac{d^+(\mathbf{v}) - d^-(\mathbf{v})}{d^+(\mathbf{v}) + d^-(\mathbf{v})}, \tag{2.34}$$

where $d^+(\mathbf{v}) = d(\mathbf{v}, \mathbf{w}^+)$ is the distance between data point $\mathbf{v}$ and its the best matching prototype $\mathbf{w}^+$ with the same label, i. e. $c(\mathbf{v}) = y(\mathbf{w}^+)$. Otherwise, $d^-(\mathbf{v}) = d(\mathbf{v}, \mathbf{w}^-)$ is the distance between data point $\mathbf{v}$ and the closest, wrong labeled prototype $\mathbf{w}^-$, i. e. $c(\mathbf{v}) \neq y(\mathbf{w}^-)$. More precisely:

$$\mathbf{w}^+ = \underset{\mathbf{w} \in W^+(\mathbf{v})}{\operatorname{argmin}}\ d(\mathbf{v}, \mathbf{w}) \quad \text{with } W^+(\mathbf{v}) = \{\mathbf{w} \in W \mid y(\mathbf{w}) = c(\mathbf{v})\} \tag{2.35}$$

$$\mathbf{w}^- = \underset{\mathbf{w} \in W^-(\mathbf{v})}{\operatorname{argmin}}\ d(\mathbf{v}, \mathbf{w}) \quad \text{with } W^-(\mathbf{v}) = \{\mathbf{w} \in W \mid y(\mathbf{w}) \neq c(\mathbf{v})\} \tag{2.36}$$

Obviously, the classifier function (2.34) is non-positive iff the data point is correct classified, i. e. $d^+(\mathbf{v}) \leq d^-(\mathbf{v})$ is valid. Due to the normalization term $d^+(\mathbf{v}) + d^-(\mathbf{v})$ the range of $\mu_W(\mathbf{v})$ is in the interval $\mu_W(\mathbf{v})$ is $[-1, 1]$.

The cost function is a soft version of the classification error and has the form of:

$$E_{GLVQ} = \frac{1}{N_V} \sum_{\mathbf{v} \in V} f_\theta(\mu_W(\mathbf{v})) \tag{2.37}$$

with the *transfer function* $f_\theta$. The transfer function has to be monotonically increasing. Common choices are the identity or the sigmoid function

$$f_\theta(x) = \frac{1}{1 + e^{-\theta \cdot x}}. \tag{2.38}$$

which depends on the parameter $\theta$. Choosing the latter one, the range of the sum terms in (2.37) is $[0, 1]$ and, hence, the cost function approximates the classification error for $\theta \nearrow \infty$ (see Sec. 5.1).

The minimization of the differentiable cost function (2.37) can be done by stochastic gradient descent. The update rules result in

$$\mathbf{w}^\pm \leftarrow \mathbf{w}^\pm - \alpha_W \frac{\partial E_{GLVQ}}{\partial \mathbf{w}^\pm} \tag{2.39}$$

with learning rate $0 < \alpha_W \ll 1$ and derivatives

$$\frac{\partial E_{GLVQ}}{\partial \mathbf{w}^{\pm}} = \xi_{\theta}^{\pm}(\mathbf{v}) \cdot \frac{\partial d^{\pm}(\mathbf{v})}{\partial \mathbf{w}^{\pm}} \tag{2.40}$$

$$\text{with} \quad \xi_{\theta}^{\pm}(\mathbf{v}) = f'_{\theta}(\mu_W(\mathbf{v})) \cdot \frac{\pm d^{\mp}(\mathbf{v})}{(d^{+}(\mathbf{v}) + d^{-}(\mathbf{v}))^2} \,. \tag{2.41}$$

As it can be seen in (2.40), the updates depend on the derivatives of the distance measure according to the prototypes. Therefore, one requirement on the distance measure is the differentiability. A common choice is the squared Euclidean distance which ends up in a vector shift of the prototypes during the update:

$$\frac{\partial d^{\pm}(\mathbf{v})}{\partial \mathbf{w}^{\pm}} = -2(\mathbf{v} - \mathbf{w}^{\pm}) \,. \tag{2.42}$$

It turns out, the GLVQ with squared Euclidean distance and optimization by SGD is alike the LVQ 2.1 with data dependent factors in the update (see Sec. 5.1).

The GLVQ belongs to the margin optimizers like the SVM. Yet, the GLVQ optimizes the hypothesis margin $\frac{1}{2}(d^{+}(\mathbf{v}) - d^{-}(\mathbf{v}))$ [Crammer et al., 2003]. The hypothesis margin is related to the distance a prototype can be altered without changing the classification decision. Thus, this fact indicates an efficient structural risk optimization in the learning phase [Hammer et al., 2001]. Further, in case of overlapping classes the GLVQ is more robust than LVQ, since convergence is assured.

Moreover, LVQ or GLVQ are flexible related to the distance measures: any dissimilarity measure which is differentiable according to the prototypes can be applied and distance adaption is also feasible (see Sec. 2.4).

Further, GLVQ is designed for multi-class problems, i. e. $N_C > 2$. The only requirement is to provide at least one prototype per class and therefore, the complexity of the resulting model has to be determined in advance. The correct number of prototypes is hard to identify and depends on the data set. However, *less is more*: In many applications one prototype per class is adequate if additionally distance adaption is applied ([Biehl et al., 2012],[Biehl et al., 2013]). Note, if the standard GLVQ with Euclidean distance and one prototype per class is performed, it is equivalent to a linear classifier.

Analog to the methods for unsupervised VQ, different variants of GLVQ are available: Median GLVQ [Nebel et al., 2013], Relation GLVQ [Hammer et al., 2011] and also fuzzy variants [Villmann et al., 2008]. Further, probabilistic variants of the LVQ methods exist: Soft Nearest Prototype Classifier (SNPC) [Seo et al., 2003] and Robust Soft LVQ (RSLVQ) [Seo and Obermayer, 2003] with median, relational variants, and so on ([Nebel et al., 2014],[Hammer et al., 2014a]). Yet, in [Nebel and Villmann, 2013] it is discussed that the result of SNPC and RSLVQ is similar, beside the different motivations.

Moreover, the combination of unsupervised NG and supervised GLVQ, the so called

*Supervised Neural Gas* (SNG), is proposed in [Villmann et al., 2003]. The SNG provides a more stable convergence behavior, because the neighborhood cooperativeness of NG is integrated into the GLVQ principle.

To summarize: GLVQ is a crucial tool solving classification problems and has a lot of extensions and modifications. The GLVQ or variants thereof can be used for a lot of different classification problems with various properties of the data sets, e. g. low or high dimensional data, two- or multi-class problems, unbalanced data sets as well as data sets with application properties like models with complexity constrains, model interpretation or visualization ability ([Biehl et al., 2012], [Bashyal and Venayagamoorthy, 2008], [T. Villmann and Riedel, 2012], [Kästner et al., 2013]).

The GLVQ is a classical supervised learning method and can only handle labeled data. Yet, in practical applications the labeling of each data point can be very expensive. Therefore, there exists data sets with labeled and unlabeled data samples. Here, semi-supervised methods come into contribution. They are described in the next section.

## 2.3 Semi-Supervised Vector Quantization

As already mentioned, the term *semi-supervised* is used for methods which can handle labeled as well as unlabeled data points. Thereby, the number of unlabeled data samples is often much higher compared to the number of labeled samples. Thereby, it has to be distinguished between transduction and inductive models. Transduction methods only predict the classes of the unlabeled data. On the other side, inductive methods generating a model for label-prediction of labels of unseen data. In the following, we only consider inductive models. A summarization semi-supervised learning in general and different approaches like EM-strategies or the Transduction-SVM can be found in [Chapelle et al., 2006].

Further, in several approaches of semi-sVQ, the results end up with fuzzy labels for the prototypes, i. e. the prototypes have no crisp labels, but probabilistic or possibilistic assignments (cf. fuzzy clustering on page 29). In this section, semi-supervised approaches integrating label information in a classical uVQ method are explained.

### 2.3.1 Learning Associations by Self-Organization

Learning Associations by Self-Organization (LASSO)[4] combines the dissimilarity measure between the data points and prototypes in the data space and further the dissimilarity measure between the label information in the distance function. Suppose $N_C$ classes are given. Each training data vector $\mathbf{v}$ is accompanied by a label vector $\mathbf{c}(\mathbf{v}) = \left(c(\mathbf{v})^1, \ldots, c(\mathbf{v})^{N_C}\right) \in [0,1]^{N_C}$ whose entries are taken as probabilistic or possibilistic class assignments. Analogously, each prototype $\mathbf{w}$ is equipped with a class label vector $\mathbf{y}(\mathbf{w}) \in [0,1]^{N_C}$. Crisp classification is obtained by the additional requirements of $c(\mathbf{v})^j \in \{0,1\}$ and the use of the Euclidean norm $||\mathbf{c}(\mathbf{v})||_E = 1$, otherwise the assignments are denoted as *fuzzy*.

Basically, the idea of LASSO is as follows: the input of the `Heskes`-SOM model is a vector $\tilde{\mathbf{v}}$ extended by the class information $\tilde{\mathbf{v}} = [\mathbf{v}; c(\mathbf{v})]^T$:

$$E_{LASSO} = \int P(\mathbf{v}) \; lc^{SOM}(\tilde{\mathbf{v}}, W, \sigma) \; d\mathbf{v}$$

Therefore, beside the location of the prototypes also the class assignments of the prototypes are learned. Another interpretation of this idea is: the distance measure $D(\mathbf{v}, \mathbf{w}, \mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}))$ used for the winner determination (2.10) during learning is an additive combination of the distance between data and prototypes in the data space and

[4]Note, this LASSO approach should not be confused with the regularization method *least absolute shrinkage and selection operator* (Tip-Lasso, [Midenet and Grumbach, 1994]). Despite the latter one is more popular, due to historical reason we decided to use the original abbreviation.

the distance of the label vectors:

$$D\left(\mathbf{v}, \mathbf{w}, \mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w})\right) = d_E^2(\mathbf{v}, \mathbf{w}) + d_E^2(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w})) \tag{2.43}$$

Yet, in the recall phase, the class information of the data points are not available and therefore cannot be contributed.

LASSO is a classical semi-supervised method because unlabeled and labeled data can be used to train the model. Further, this method is also suitable to handle fuzzy labeled data sets. In general the resulting prototypes have fuzzy assignments, too.

Hence, LASSO has a high variety in applications, but is very restricted concerning the influence of the class labels during learning. A generalization and extension of LASSO is the Fuzzy Labeled Self-Organizing Map model (FLSOM, see next part) or Fuzzy Semi-Supervised SOM (FSSOM, see Sec. 4.2.1).

### 2.3.2 Fuzzy Labeled Self-Organizing Map

The Fuzzy Labeled Self-Organizing Map (FLSOM) is introduced in [Villmann et al., 2007]. For FLSOM the original cost function of `Heskes`-SOM:

$$E_{SOM} = \int P(v)\; lc^{SOM}\left(\mathbf{v}, W, \sigma\right) d\mathbf{v}$$

is extended by a term including the class information:

$$E_{FLSOM} = (1-\gamma) \cdot E_{\mathrm{SOM}} + \gamma \cdot E_{\mathrm{FL}}^{SOM} \tag{2.44}$$

with

$$E_{\mathrm{FL}}^{SOM} = \sum_{\mathbf{r}\in\mathbf{A}} \int P\left(\mathbf{v}\right) \cdot g^{SOM}\left(\mathbf{v}, \mathbf{w_r}, \beta\right) \cdot \delta\left(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w_r})\right)\; d\mathbf{v} \tag{2.45}$$

where $\delta\left(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w_r})\right)$ is any dissimilarity measure for the class assignment vectors. The parameter $\gamma \in [0, 1]$ determines the influence of the class information whereby $\gamma = 0$ yields the `Heskes`-SOM.

For the choice of the neighborhood cooperativeness function $g^{SOM}$ in (2.45) exists to possibilities:

**case I** The neighborhood cooperativeness function $g^{SOM}$ is based on the distance of the prototypes in the grid like in $E_{SOM}$:

$$g_I^{SOM}\left(\mathbf{v}, \mathbf{w_r}, \beta\right) = \exp\left(-\frac{d_A\left(\hat{\mathbf{s}}(\mathbf{v}), \mathbf{r}\right)}{2\beta}\right)$$

If additionally $\sigma = \beta$ is valid, both neighborhood functions of $E_{SOM}$ and $E_{FL}^{SOM}$ are identically and the cost function (2.47) can be rewritten:

$$E_{FLSOM} = \int P(V) \sum_{\mathbf{r} \in A} h_\sigma^{SOM}(\hat{\mathbf{s}}(\mathbf{v}), \mathbf{r}) \left[ (1-\gamma) \cdot d(\mathbf{v}, \mathbf{w_r}) + \gamma \cdot \delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w_r})) \right] \, d\mathbf{v}$$

Therefore, it is a generalization of LASSO with the additive dissimilarity term:

$$D_{add}(\mathbf{v}, \mathbf{w}, \mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w})) = (1-\gamma) \cdot d(\mathbf{v}, \mathbf{w}) + \gamma \cdot \delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}))$$

However, the disadvantages of distinguish winner detection in training and recall phase known from LASSO remains.

**case II** The approach which is presented in [Villmann et al., 2007] chooses the neighborhood cooperativeness function $g^{SOM}(\mathbf{v}, \mathbf{w_r}, \beta)$ as Gaussian kernel

$$g_{II}^{SOM}(\mathbf{v}, \mathbf{w_r}, \beta) = \exp\left( -\frac{d(\mathbf{v}, \mathbf{w_r})}{2\beta} \right) \tag{2.46}$$

based on the distance $d(\mathbf{v}, \mathbf{w_r})$ in the data space. It is assumed if the classification problem is fairly smooth, data points next to a prototype determine the corresponding label.

In **case II** the updates of the prototype depends on both cost function terms of (2.44):

$$\Delta \mathbf{w_r} = \left( (\gamma - 1) h_\sigma^{SOM}(\hat{\mathbf{s}}(\mathbf{v}), \mathbf{r}) + \frac{\gamma}{4\beta^2} g_{II}^{SOM}(\mathbf{v}, \mathbf{w_r}, \beta) \left( \mathbf{c}(\mathbf{v}) - \mathbf{y}(\mathbf{w_r}) \right) \right) \frac{\partial d(\mathbf{v}, \mathbf{w_r})}{\partial \mathbf{w_r}}$$

Further, the updates of the labels are:

$$\Delta \mathbf{y}(\mathbf{w_r}) = \gamma \cdot g_{II}^{SOM}(\mathbf{v}, \mathbf{w_r}, \beta) \cdot \frac{\partial \delta\left(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w_r})\right)}{\partial \mathbf{y}(\mathbf{w_r})}$$

While the latter case with $g_{II}^{SOM}$ is more complicated and not easy to interpret, the winner detection in the learning and recall phase is the same, i. e. the winner detection is independent from the labels.

As well as in LASSO, the data points can be equipped without, with crisp or fuzzy labels. For visualization of the results component planes as well as the label maps can be used. The label map show the fuzzy labels of each neuron in the grid. An example can be found in Application 4.5.

### 2.3.3 Fuzzy Labeled Neural Gas

The same idea like in FLSOM can be transfered to the Neural Gas [Villmann et al., 2006a] called Fuzzy Labeled Neural Gas (FLNG). The cost func-

tion of FLNG has the same structure

$$E_{\mathrm{FLNG}} = (1-\gamma)\, E_{\mathrm{NG}} + \gamma E_{\mathrm{FL}}^{NG} \tag{2.47}$$

but is based on Neural Gas and with

$$E_{\mathrm{FL}}^{NG} = \sum_{j=1}^{N_W} \int P(\mathbf{v})\, g^{NG}(\mathbf{v}, \mathbf{w}_j)\, \delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_j))\, . \tag{2.48}$$

Again, $\delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_j))$ is a dissimilarity measure for the class assignment vectors and $\gamma \in [0,1]$ determines the influence of the class information. The neighborhood cooperativeness function $g^{NG}(\mathbf{v}, \mathbf{w}_j)$ in FLNG explicitly takes into account the dissimilarity $d(\mathbf{v}, \mathbf{w}_j)$ between the prototype $\mathbf{w}_j$ and the data vector $\mathbf{v}$. Once more, the choice of $g^{NG}(\mathbf{v}, \mathbf{w}_j)$ has to be divided into two cases. This time the cases regard the properties of the data distribution: discrete or continuous. For continuous data the neighborhood function becomes

$$g_{cont}^{NG}(\mathbf{v}, \mathbf{w}_j) = \exp\left(-\frac{d(\mathbf{v}, \mathbf{w}_j)}{2\sigma^2}\right)$$

whereas

$$g_{discr}^{NG}(\mathbf{v}, \mathbf{w}_j) = h_\sigma^{NG}(k_j(\mathbf{v}, \mathbf{w}_j))$$

is valid for the discrete setting. In the latter case, the cost function (2.47) can be further simplified to

$$E_{\mathrm{FLNG}} = \sum_{j=1}^{N_W} \sum_{\mathbf{v} \in V} P(\mathbf{v})\, h_\sigma^{NG}(k_j(\mathbf{v}, \mathbf{w}_j))\, D_{add}(\mathbf{v}, \mathbf{w}_j, \mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_j), \gamma)$$

with a new *additive* distortion measure

$$D_{add}(\mathbf{v}, \mathbf{w}_j, \mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_j), \gamma) = (1-\gamma)\, d(\mathbf{v}, \mathbf{w}_j) + \gamma\, \delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_j)). \tag{2.49}$$

To simplify the idea of the discrete FLNG, only the distance of the Neural Gas $d(\mathbf{v}, \mathbf{w})$ is replaced by the *additive* distortion measure $D_{add}(\mathbf{v}, \mathbf{w}, \mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}), \gamma)$ which include the class information.

The necessity for the distinction of the discrete and continuous cases in the neighborhood cooperativeness for the labels is the consequence to assure a valid convergence proof for the algorithm. Details can be found in [Villmann et al., 2006a]. Another similar method the Fuzzy Supervised Neural Gas (FSNG) is presented in section 4.2.2. In the FSNG the discrete and continuous cases has not be distinguished and this method is a good alternative for the FLNG.

**Parts of the section are based on**

*T. Villmann, S. Haase, and M. Kaden:* Kernelized Vector Quantization in Gradient-Descent Learning, *Neurocomputing, 2014.*

*M. Riedel, F. Rossi, M. Kästner, and T. Villmann:* Regularization in Relevance Learning Vector Quantization Using $l_p$-Norm. *ESANN, 2013.*

## 2.4 Dissimilarity Measures

In this section, we take a closer look to the dissimilarity or distance measures, respectively (see Note 2.1). Until now, the phrase '$d(\mathbf{v}, \mathbf{w})$ is any distance measure is written very often. Nearly all methods in VQ were developed using the (squared) Euclidean distance. However, not for every data set this Euclidean measure is the best choice. Therefore, e. g. in [Hammer and Villmann, 2005] the idea of using other distance measures or the adaptation of distance parameters came up. Possible distances may be: $l_p$-distances, Sobolev distance, divergences, kernel based distances or covariances. The applied distance measure do not have to be a metric (see Note 2.1). Yet, an important requirement is the differentiability according to the second argument, because all described online VQ methods in this theses using the stochastic gradient descent to optimize their cost functions.

A frequently ask question is: which measure should be used for a given data set. Unfortunately, no general answer exists. Though, certain properties of the data set should be considered when a measurement is chosen. In the following, some distance measures are described closer.

### Minkowski Distances

A generalization of the Euclidean distance is the fundamental *Minkowski* distance or also called $l_p$-distance:

$$d_{l_p}(\mathbf{v}, \mathbf{w}) = \left( \sum_{k=1}^{D} |v_k - w_k|^p \right)^{\frac{1}{p}} \tag{2.50}$$

where $p > 0$. If $p \geq 1$, it is even a metric and it yields

$$d_{l_p}(\mathbf{v}, \mathbf{w}) = ||\mathbf{v} - \mathbf{w}||_{l_p} \tag{2.51}$$

with $||\cdot||_{l_p}$ is the $l_p$-norm. Special cases are $p = 2$ as Euclidean distance and the Manhattan metric with $p = 1$. Moreover, if $0 < p < 1$, $(d_{l_p})^p = ||\mathbf{v} - \mathbf{w}|^p|_{l_p}$ is a metric, too and small distinctions have a higher weight and the other way around [Lange et al., 2014]. Further, it has to point out that for special values of $p$, $d_{l_p}(\mathbf{v}, \mathbf{w})$ is not differentiable due

to the absolute value function. However, it exists several differentiable approximations, which were applied in [Lange and Villmann, 2013].

**Sobolev Distance**

A special kind of data are functional data, i.e. the data points are discrete representations of functions $v_t = v(t)$ (see Note 1.1). The Sobolev distance $d_S^{\gamma,k}$ implies beside the standard Euclidean distance of the data points also the Euclidean distance of the derivatives up to the $k^{th}$ order of a data sample $\mathbf{v}^{(k)}$ or prototype $\mathbf{w}^{(k)}$, respectively:

$$d_S^{\gamma,k}(\mathbf{v},\mathbf{w}) = \gamma_0 \cdot d_E(\mathbf{v},\mathbf{w}) + \sum_{l=1}^{k} \gamma_l \cdot d_E(\mathbf{v}^{(l)},\mathbf{w}^{(l)}) \tag{2.52}$$

with the linear combination factors $\gamma_l > 0$ and $\sum_{l=0}^{k} \gamma_l = 1$. The factors $\gamma_l$ weight the influence of the single derivatives.

The generalization of (2.52) is the p-Sobolev distance analog to the Minkowski distance:

$$d_{S_p}^{\gamma,k}(\mathbf{v},\mathbf{w}) = \gamma_0 \cdot d_{l_p}(\mathbf{v},\mathbf{w}) + \sum_{l=1}^{k} \gamma_l \cdot d_{l_p}(\mathbf{v}^{(l)},\mathbf{w}^{(l)}) \tag{2.53}$$

whereby for $p = 2$ yields the Sobolov distance $d_S^{\gamma,k}$.

In general, the derivatives of the discrete representation of the data points are not given and has to be approximated e. g. by a difference operator. Due to the approximation and consequential approximation error, $k$ is often set to $k = 1$. A theoretical integration of the p-Sobolev norm into VQ can be found in [Villmann, 2007] and [Villmann and Hammer, 2009].

### Functional Distance

A further distance for functional data is presented by `Lee & Verleysen` [Lee and Verleysen, 2005]. This dissimilarity measure is based on a quasi-norm[5] and is motivated by geometrical considerations. The functional distance is obtained as:

$$d^{\tau}_{F_p}(\mathbf{v},\mathbf{w}) = \left( \sum_{k=1}^{D} \Big( A_{k-1}(\mathbf{v},\mathbf{w}) + A_{k+1}(\mathbf{v},\mathbf{w}) \Big)^p \right)^{\frac{1}{p}} \tag{2.54}$$

with

$$A_{k\pm1}(\mathbf{v},\mathbf{w}) = \begin{cases} \frac{\tau}{2}|\mathbf{v}_k - \mathbf{w}_k| & \text{, if } 0 \leq (\mathbf{v}_k - \mathbf{w}_k)(\mathbf{v}_{k\pm1} - \mathbf{w}_{k\pm1}) \\ \frac{\tau}{2} \frac{([\mathbf{v}-\mathbf{w}]_k)^2}{|\mathbf{v}_k - \mathbf{w}_k| + |\mathbf{v}_{k\pm1} - \mathbf{w}_{k\pm1}|} & \text{, if } 0 > (\mathbf{v}_k - \mathbf{w}_k)(\mathbf{v}_{k\pm1} - \mathbf{w}_{k\pm1}) \end{cases} \tag{2.55}$$

with the same restriction to $p$ known from the $l_p$-distance, i. e. $p \geq 1$. An application of this distance measure by a respective Supervised Neural Gas can be found in [Schleif et al., 2007b].

### Divergences

The divergences are based on information theory and calculate the information content between two probability distributions $\mathbf{v}$ and $\mathbf{w}$, i. e. $\sum_{k=1}^{D} v_k = 1$, $\sum_{k=1}^{D} w_k = 1$ and $v_k, w_k \geq 0 \, \forall k$. In general, divergences are hollow metrics, i. e. the symmetry property is not fulfilled (see Note 2.1). The family of divergences can divided at least into three main groups: Bregmann, Csiszar-f, and $\gamma$-divergences (see Fig 2.3) [Cichocki and Amari, 2010]. The intersection of all contains the well known Kullback-Leibler-Divergence for a particular parameter (discrete case):

$$d_{KL}(\mathbf{v},\mathbf{w}) = -\sum_{k=1}^{D} v_k \cdot \log\left(\frac{v_k}{w_k}\right) \tag{2.56}$$

An elaborate summarization of divergences and the integration in VQ methods can be found in [Haase, 2014].

### Kernels

*Kernel* functions are very popular and widely used especially in classification methods. A kernel is a similarity measure, more precisely an inner product in a sufficiently special kind of feature space. The background is that every binary classification problem

[5] In [Villmann and Lange, 2015] it is shown that the original Lee&Verleysen term is not a norm and therefore, the according dissimilarity measure is not a metric.

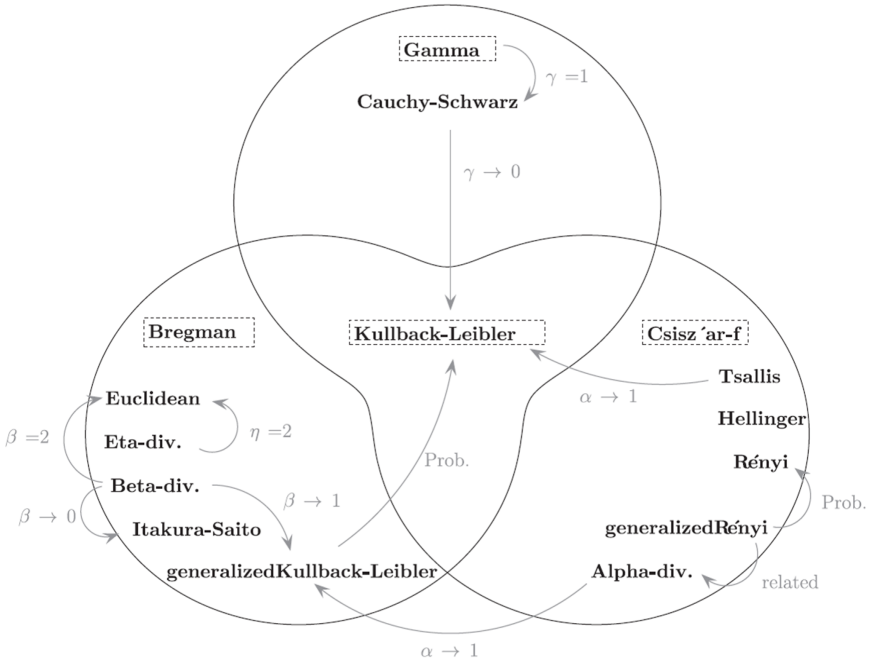


**Figure 2.3**: Illustration of the family of divergences [Bunte, 2011]

is linear separable if the data are mapped into a high (may be infinite) feature space non-linearly [Schölkopf and Smola, 2002]. The principle of kernel mapping is based on the theoretic work by `Mercer` and `Aronzaijn` about positive semi-definite and reproducing kernels [Aronszajn, 1950, Mercer, 1909].

We consider a Hilbert space $\mathcal{H}$ equipped with an inner product

$$< \cdot, \cdot >_{\mathcal{H}} \colon \mathcal{H} \times \mathcal{H} \to \mathbb{C}.$$

The fundamental theorem of Moore-Aronzaijn implies: For a set $V$ the kernel function $\kappa : V \times V \to \mathbb{C}$ is symmetric and positive semi-definite[6] if and only if the mapping $\Phi : V \to \mathcal{H}$ exists and the feature space $\mathcal{H}$ is a functional Hilbert space with the inner product

$$\kappa(\mathbf{v}, \mathbf{w}) = \langle \Phi(\mathbf{v}), \Phi(\mathbf{w}) \rangle_{\mathcal{H}} \tag{2.57}$$

where $\mathbf{v}, \mathbf{w} \in V$ and $\Phi(V) = \mathcal{I}_\kappa \subseteq \mathcal{H}$ [Christmann and Steinwart, 2010]. In the following, we consider positive semi-definite and real-valued kernels.

So far, the functional Hilbert space, denoted as *Feature Mapping Hilbert Space (FMHS)*, as well as the mapping $\Phi$ are not unique. They become unique if the kernel holds

[6]symmetric: $\kappa(\mathbf{v}_1, \mathbf{v}_2) = \kappa(\mathbf{v}_2, \mathbf{v}_1)\ \forall \mathbf{v}_1, \mathbf{v}_2 \in V$; positive semi-definite in case of a finite set $V$: the according Gram matrix $K$ with the entries $k_{ij} = \kappa(\mathbf{v}_i, \mathbf{v}_j)$ is positive semi-definite $\forall \mathbf{v}_i, \mathbf{v}_j \in V$ and $i, j = 1, \ldots, N_V$.

the *reproducing* property, i. e. for the kernel applies $\kappa(\mathbf{v}, \mathbf{w}) = \langle f_{\mathbf{v}}, f_{\mathbf{w}} \rangle_{\mathcal{H}}$ $(\forall \mathbf{v}, \mathbf{w} \in V)$ with $f_{\mathbf{v}}(\cdot) = \kappa(\mathbf{v}, \cdot)$ [Zhang et al., 2009, Mercer, 1909]. Further, a reproducing kernel is positive definite[7] and imply injectivity of their related feature maps [Aronszajn, 1950]. The FMHS associated by a positive definite kernel $\kappa$ is denoted by $\mathcal{H}_\kappa$.

Note, the kernel property (2.57) allows a direct calculation of the inner product with the elements in $V$ without explicit processing the mapping $\Phi$. Thus, the mapping is only implicitly performed whereby in the most cases $\Phi$ as well as $\mathcal{H}$ are unknown. This concept is called *kernel trick*. A summarization of definitions and theorems about the kernel theory can be found in [Steinwart and Christmann, 2008] or [Schölkopf and Smola, 2002].

The principle of kernel mapping can be transfer to the prototype based Vector Quantizations methods, too. As mentioned before, the kernels itself determine real inner products and consequently similarities. Thus, they can not be applied to the VQ methods directly. Yet, each inner product determines a distance measure according to

$$d_\kappa(\mathbf{v}, \mathbf{w}) \quad = \quad \langle \Phi(\mathbf{v}) - \Phi(\mathbf{w}), \Phi(\mathbf{v}) - \Phi(\mathbf{w}) \rangle_{\mathcal{H}}, \tag{2.58}$$

which leads after elementary calculations to:

$$d_\kappa(\mathbf{v}, \mathbf{w}) \quad = \quad \sqrt{\kappa(\mathbf{v}, \mathbf{v}) - 2\kappa(\mathbf{v}, \mathbf{w}) + \kappa(\mathbf{w}, \mathbf{w})} \tag{2.59}$$

If the symmetric kernel is only positive semi-definite, the $d_\kappa$ is only a semi metric (see Note 2.1).Though, a positive definite symmetric kernel induces a metric $d_\kappa$ [Steinwart, 2001a]. Examples of differentiable kernel induced distances and their derivatives are listed in Table A.1.

To recapitulate this part; Several dissimilarity measures exist with different theoretical backgrounds. Some of them are restricted to special structures of the data others can applied on every kind of data. Yet, if the distance measure is differentiable, it can be plugged into the prototype based Vector Quantization methods. Due to the non-trivial theory of the kernels, we have a closer look to the integration of them into GLVQ.

### 2.4.1 Differentiable Kernels in Generalized LVQ

A well established classifier using the *kernel trick* is the Support Vector Machine (SVM). The basic (Euclidean)-SVM optimizes the separation margin of a binary class problem and according to the minimization of the *structural risk* the model provides a good generalization performance ([Vapnik, 1995], [Schölkopf and Smola, 2002]). Yet, this statement applies only for linear separable problems. In many cases the used SVM is based

[7] The Gram matrix is strict positive.

on a soft-margin optimization and a kernel mapping (see appendix A.2). Consequently, the structural risk minimization is not longer guaranteed. Nevertheless, in many applications the SVM or a modification of it outperforms other classifiers. The SVM is used in many practical application due to two main reasons:

a) fast computation: the modeling as a quadratic optimization problem with convex constraints and

b) good separation property: the integration of kernels.

The first aspect a) ensures the determination of an unique global minimum in adequate time [Bordes et al., 2005]. Quadratic optimizations problems with convex constraints are sufficiently investigated and thus, fast and efficient algorithms exist like the sequential minimal optimization (SMO, [Platt, 1998]).

The second reason b) is associated by the *kernel trick* as already explained in the last section. The mapped prototypes $\Phi(\mathbf{w})$ are linear combinations of the mapped data points $\Phi(\mathbf{v})$. The prototypes are called *support vectors* (SV) due to the fact: the SV are determine the separation margin. Yet, the interpretability of the SV and the resulted model using the kernel mapping is at least difficult, because the class borders are defined in the FMHS $\mathcal{H}_\kappa$.

There are other limits for the SVM, too. It is common to give the test accuracy to evaluate a model determine the generalization ability. Yet, the difference between test and training accuracy and, especially, the number of support vectors (SV) are also important for a statement about the generalization ability. In many applications the relation between number of SV and data points are really high (sometimes greater than $25\%$) [Lange et al., 2013], i. e. nearly every data point is needed for the separation decision (classification). This high model complexity can causes an over-fitting of the classification problem. A more detailed description of the SVM can be found in the Appendix A.2 or in [Schölkopf and Smola, 2002].

For GLVQ the number of prototypes has to be defined before and hence, the complexity of the resulted model is determined by the user in advance. However, the standard GLVQ with the Euclidean distance often can not achieve such a great flexibility in class separation like the kernel methods. Thus, ideas came up to integrate the kernel trick into the GLVQ principle. One idea, presented by `Quin and Suganthan` in [Qin et al., 2004] and called Kernelized GLVQ (KGLVQ), pursued the same principle like SVM: the data points as well as the prototypes are described and handled in the FMHS $\mathcal{H}$. More precisely, here the prototypes $\mathbf{w}_j \in \mathcal{H}$ are linear combinations of the mapped data points:

$$\mathbf{w}_j = \sum_{i=1}^{N_V} \alpha_{ij} \Phi(\mathbf{v}_i) \tag{2.60}$$

with the linear combination vector $\alpha_{:j} \in \mathbb{R}^{N_V}$. Thus, instead of shifting the prototypes, the factors $\alpha_{ij}$ are adapted like it is known from the relational variants (see page 29). Yet, in the relation variants the prototypes are still in the data space. In the KGLVQ the prototypes are elements of the FMHS. Again, the knowledge of the mapping $\Phi(\mathbf{v}_i)$ is not required, because the squared distance in the feature space $\mathcal{H}$ can be computed by:

$$\begin{aligned} d^2_\kappa(\mathbf{v}_l, \mathbf{w}_j(\alpha)) &= \langle \Phi(\mathbf{v}_l) - \Phi(\mathbf{w}_j(\alpha)), \Phi(\mathbf{v}_l) - \Phi(\mathbf{w}_j(\alpha)) \rangle^2_{\mathcal{H}} \\ &= \left\langle \Phi(\mathbf{v}_l) - \left( \sum_{i=1}^{N_V} \alpha_{ij} \Phi(\mathbf{v}_i) \right), \Phi(\mathbf{v}_l) - \left( \sum_{i=1}^{N_V} \alpha_{ij} \Phi(\mathbf{v}_i) \right) \right\rangle^2_{\mathcal{H}} \\ &= \kappa(\mathbf{v}_l, \mathbf{v}_l) - 2 \sum_{i=1}^{N_V} \alpha_{il}\ \kappa(\mathbf{v}_i, \mathbf{v}_l) + \sum_{s=1}^{N_W} \sum_{t=1}^{N_V} \alpha_{sj}\ \alpha_{tj}\ \kappa(\mathbf{v}_s, \mathbf{v}_t)\,, \end{aligned}$$

which is only based on the inner products between data points on $\mathcal{H}$ calculated by the kernel. This approach has the advantage that the kernels $\kappa(\mathbf{v}_i, \mathbf{v}_l)$ do not have to be calculated in each iteration step. Moreover, only the similarity matrix between the data has to be given for calculation of the Gram-Matrix $G = \{\kappa(\mathbf{v}_j, \mathbf{w}_j)\}_{ij}$. The data points itself are not longer required. Yet, an assumption on the similarity matrix G is that an Euclidean embedding exists [Hammer et al., 2014a]. A sparse and improved enhancement of the KGLVQ concerning the learning complexity can be found in [Schleif et al., 2011]. However as pointed out for SVM, the prototypes are not longer interpretable due to the fact of *living* in the unknown FMHS (2.60) and the principle of adapting the prototypes by shifting is lost.

**GLVQ applying Differentiable Kernels**

A further idea to integrate the kernel concept in the GLVQ is the DK-GLVQ [Villmann et al., 2015]. Thereby, the distance measure $d(\mathbf{v}, \mathbf{w})$ in the cost function (2.37) is replaced by the distance measure deduced from the kernel $\kappa(\mathbf{v}, \mathbf{w})$;

$$d_\kappa(\mathbf{v}, \mathbf{w}) = \sqrt{\kappa(\mathbf{v}, \mathbf{v}) - 2\ \kappa(\mathbf{v}, \mathbf{w}) + \kappa(\mathbf{w}, \mathbf{w})}. \tag{2.61}$$

Applied to the famous *radial basis kernel* (RBF)

$$\kappa_{RBF}(\mathbf{v}, \mathbf{w}, \varsigma) = e^{\frac{-d_E(\mathbf{v}, \mathbf{w})}{2\varsigma^2}} \tag{2.62}$$

we obtain the according distance:

$$d_{RBF}(\mathbf{v}, \mathbf{w}, \varsigma) = \sqrt{2 - 2 \cdot \kappa_{RBF}(\mathbf{v}, \mathbf{w}, \varsigma)}\,. \tag{2.63}$$

As already mentioned, for symmetric and positive semi-definite kernels $\kappa$, $d_\kappa$ is only

a semi metric. If the mapping $\Phi_\kappa : V \to \mathcal{I}_\kappa \subseteq \mathcal{H}$ is injective, i. e. the kernel is positive definite, $d_\kappa(\mathbf{v}, \mathbf{w})$ becomes a metric and it yields:

$$d_\kappa(\mathbf{v}, \mathbf{w}) = ||\Phi(\mathbf{v}) - \Phi(\mathbf{w})||_\mathcal{H} \,. \tag{2.64}$$

It can be shown under some circumstances that the compact metric space $(V, d_\kappa)$ equipped with the kernel induced distance $d_\kappa$ provides the same topological richness like the FMHS $(\mathcal{I}_\kappa, d_\kappa)$ [Villmann et al., 2015]:

A kernel is continuous iff the mapping $\Phi$ is continuous [Steinwart, 2001b]. Further, a scalar function $f_g : V \to \mathbb{R}$ is induced by $\kappa$ if there exist an element $g$ in the Hilbert space $\mathcal{H}_\kappa$ with $f_g(\mathbf{v}) = \langle g, \Phi_\kappa(\mathbf{v}) \rangle_{\mathcal{H}_\kappa}$. A continuous kernel on a compact metric space $(V, d)$ is *universal* if the space $\mathcal{F}_\kappa$ of all functions induced by $\kappa$ is dense in the space $\mathcal{C}(V)$ of continuous functions over V with the supremum norm $||g||_\infty = \sup_{\mathbf{v} \in V}\{|g(\mathbf{v})|\}$ [Steinwart, 2001a]. Due to the fact that universal kernels are positive definite, `Steinwart` proves the injectivity of the corresponding feature map $\Phi_\kappa$. Furthermore, the identity map $\Psi$:

$$\Psi : (V, d_E) \to (V, d_\kappa)$$

is also continuous.

Concluding, it can be reasoned [Villmann et al., 2015]:

**Remark 2.1.** Let $(V, d_E)$ a compact metric space, $\kappa$ a universal kernel with the feature map $\Phi_\kappa : V \to I_\kappa \subseteq \mathcal{H}$, and $d_\kappa$ is the metric induced by $\kappa$, then $(V, d_\kappa)$ and $(\mathcal{I}_\kappa, d_\kappa)$ are topological equivalent and isometric. The mapping $\Phi_\kappa \circ \Psi^{-1} : (V, d_\kappa) \to (\mathcal{I}_\kappa, d_\kappa)$ is continuous and bijective.

This remark is summed up in Figure 2.4 visually.

Consequently, the DK-GLVQ provides high application variety in class separation like the KGLVQ or SVM do. In Application 2.3 it is shown that the DK-GLVQ achieve similar accuracy values like the SVM applied on a non-linear separation problem, but with an underlying linear model, i. e. the number of prototypes in DK-GLVQ is set to one per class. Thus, in this example the DK-GLVQ model is very sparse compared to the SVM model, which ends up with more support vectors.

To test if a kernel is universal is challenging as it can be concluded from the definition. Yet, the most famous kernels like RBF-kernel, exponential kernel, the polynomial kernel or a special kind of radial kernels fulfill these requirements ([Micchelli et al., 2006], [Sriperumbudur et al., 2011]). The radial kernels are a class of kernels with

$$\kappa_g(\mathbf{v}, \mathbf{w}) = g(d(\mathbf{v}, \mathbf{w})) \tag{2.65}$$

where $g : \mathbb{R}^+ \to \mathbb{R}$ is a scalar function and $d(\mathbf{v}, \mathbf{w})$ is a distance measure. In

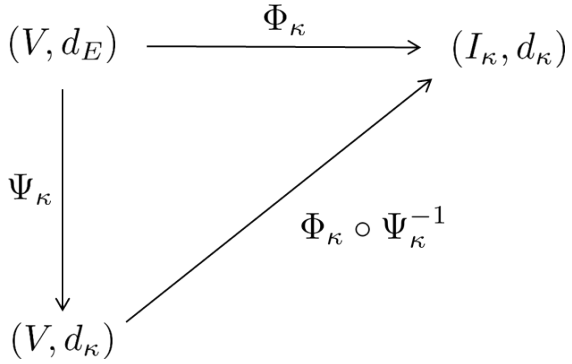


**Figure 2.4**: Visualization of the topologically equivalence and the isometric of $(V, d_\kappa)$ and $(I_\kappa, d_\kappa)$.

[Sriperumbudur et al., 2011] it is shown; if the radial kernel is positive definite, the kernel is also universal. This holds for radial kernels where $d(\mathbf{v}, \mathbf{w})$ is a metric.

The properties of be universal or positive definite have not to be fulfilled in general. In this case the measure $d_\kappa(\mathbf{v}, \mathbf{w})$ is still a semi metric (see Note 2.1). Yet, the isometry between $(V, d_\kappa)$ and $(\mathcal{I}_\kappa, d_\kappa)$ does not longer holds for those kernels.

Obviously, one further requirement on the kernel $\kappa(\mathbf{v}, \mathbf{w})$ for the integration in the DK-GLVQ is the differentiability according to the prototypes $\mathbf{w}$. Yet, the most common kernels like the radial basis function, polynomial or the Student-type gaussian kernel fulfill this requirement (see Tab. A.1). However, e. g. structural kernels like sequence and tree kernel can not be applied [Suzuki and Isozaki, 2005].

Naturally, differentiable kernels can be also plugged into the unsupervised gradient based Vector Quantization methods like NG, SOM, or other extensions of the GLVQ like the border sensitive (see Sec. 5.1) or the attention based one (see Sec. 5.2.1).

### 2.4.2 Dissimilarity Adaptation for Performance Improvement

In the last subsections several distance measures were discussed. Some of them are parameterized like the distance of the RBF-kernel $d_{RBF}(\mathbf{v}, \mathbf{w}, \varsigma)$ (2.63). A big advantage in prototype based VQ is the possibility of distance adaptation for performance improvement, i. e. additional learning of the distance parameters by stochastic gradient descent. A pioneering work in supervised Vector Quantization is [Hammer and Villmann, 2002] about *relevance learning*. The approach is denoted as Generalized Relevance LVQ (GRLVQ). Thereby, a weighted squared Euclidean distance is used:

$$d_E^2(\mathbf{v}, \mathbf{w}, \lambda) = \sum_{k=1}^{D} (\lambda_k (v_k - w_k))^2 \tag{2.66}$$

with weighting vector $\lambda^2 = \left(\lambda_1^2, \ldots, \lambda_D^2\right)$ also denotes as *relevance profile* and the normalization constraint

$$\sum_{k=1}^{D} \lambda_k^2 = 1 \,. \tag{2.67}$$

In GRLVQ, these relevance parameters are adjusted also be SGD in parallel to the prototypes. Later, this idea was also transfer to unsupervised VQ [Kästner et al., 2011].

A generalization of this relevance learning is the *matrix* variant. Instead of applying a relevance vector, a matrix is introduced:

$$d_E^2(\mathbf{v}, \mathbf{w}, \Omega) = \sum_{k=1}^{m} \left([\Omega(\mathbf{v} - \mathbf{w})]_k\right)^2 \tag{2.68}$$

where $\Omega \in \mathbb{R}^{m \times D}$ can be interpreted as a linear mapping of the data points and prototypes [Schneider et al., 2009]. The term $d_E^2(\mathbf{v}, \mathbf{w}, \Omega)$ is still a quasi metric, because it can be rewritten to

$$d_E^2(\mathbf{v}, \mathbf{w}, \Omega) = (\mathbf{v} - \mathbf{w})^T \; \Omega^T \Omega \; (\mathbf{v} - \mathbf{w}) \tag{2.69}$$

and $\Omega^T \Omega$ is sure positive semi definite. If

$$\Omega = \begin{pmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_D \end{pmatrix}$$

the distance $d_E^2(\mathbf{v}, \mathbf{w}, \Omega)$ is equivalent to $d_E^2(\mathbf{v}, \mathbf{w}, \lambda)$.

Summarized, any parameterized differential distance can plugged into a gradient based VQ method and the distance parameters can be determined automatically via SGD. Further examples of distance parameter adaptation are the learning of the kernel parameter $\varsigma$ in RBF-kernel [Villmann et al., 2015], relevance/matrix learning in divergences ([Kästner et al., 2011],[Haase, 2014]) or learning of the influence of the derivatives $\gamma$ in the Sobolev distance [Harth, 2012].

A more detailed description of the principle of distance adaptation will be given in the following section as exemplified by the relevance/matrix learning in GLVQ and DK-GLVQ.

#### Generalized Relevance/Matrix Learning Vector Quantization

In GRLVQ the update of the prototypes is analog to (2.40) applying $d^{\pm}(\mathbf{v}, \lambda)$ where $d^{\pm}(\mathbf{v}, \lambda) = d(\mathbf{v}, \mathbf{w}^{\pm}, \lambda)$ are the weighted distances to the winner prototypes according (2.35) and (2.36), respectively. Beside the prototypes the relevance profile $\lambda^2$ is adapted

in each iteration step:

$$\lambda_k \leftarrow \quad \lambda_k - \alpha_\lambda \cdot \Delta\lambda_k \tag{2.70}$$

$$\text{with} \quad \Delta\lambda_k = \quad \xi_\Theta^+(\mathbf{v}) \cdot \frac{\partial d^+(\mathbf{v},\lambda)}{\partial \lambda_k} + \xi_\Theta^-(\mathbf{v}) \cdot \frac{\partial d^-(\mathbf{v},\lambda)}{\partial \lambda_k} \tag{2.71}$$

$$\text{and} \quad \tfrac{\partial d^\pm(\mathbf{v},\lambda)}{\partial \lambda_k} = \quad 2 \cdot \lambda_k \cdot (v_k - w_k^\pm)^2 \tag{2.72}$$

where $\xi_\Theta^\pm(\mathbf{v})$ is determined analog to standard GLVQ (2.41).

For choosing the learning rate $\alpha_\lambda$ exists two strategies: From the point of minimization of the cost function both the prototypes as well as the relevance profile $\lambda^2$ are parameters to be optimized. Thus, $\alpha_\lambda$ can be initialized independent from the learning rate of the prototypes $\alpha_W$ and has to be decreased during learning (see Appendix A.1). Another point of view is the interpretation of the update as Hebbian learning scheme. Here, the distance should be adapted much slower (adiabatic) than the prototypes at least one magnitude, because a quasi stationarity of the dissimilarity measure is assumed [Hammer and Villmann, 2002].

After learning GRLVQ the resulted relevance profile $\lambda^2$ weights each dimension focusing a good class separation, i. e. if $\lambda_k^2$ has a high value, then the dimension $k$ is important for the class separation. Yet, the reverse case does not apply in general, because single features can dependent on each other and if a feature with high $\lambda_k^2$ will be dropped off, other features might compensate it [Hammer and Villmann, 2002]. The number of parameters $\lambda_k^2$ are directly dependent on the dimensionality of the data. If a high dimensional data set is given, the number of parameters to be optimized is also high and this might cause instabilities [Mendenhall and Merényi, 2008]. Frequently, for high dimensional data sets the features depend on each other like in functional data sets. In Section 4.1 possibilities to integrate this dependency of the dimensions into the learning scheme is shown up.

As already mentioned, the relevance profile weights each dimension independently from each other for a better class separation. If the relevance profile is interpret as a linear mapping only the axes are distorted, i. e. the corresponding mapping matrix is a diagonal matrix. The generalization would be a fully occupied mapping matrix. In Generalized Matrix LVQ (GMLVQ) such a linear mapping matrix for better class separation is learned [Schneider et al., 2009].

Once again, the distance $d^\pm(\mathbf{v})$ in the cost function (2.37) is replace by the *mapping distance* $d^\pm(\mathbf{v}, \Omega)$ (2.68) and beside the prototypes the *mapping matrix* $\Omega \in \mathbb{R}^{m \times D}$ is up-

dated:

$$\Omega_{kl} \leftarrow \quad \Omega_{kl} - \alpha_{\mathbf{\Omega}} \cdot \Delta\Omega_{kl} \tag{2.73}$$

$$\text{with} \quad \Delta\Omega_{kl} = \quad \xi_{\Theta}^{+}(\mathbf{v}) \, \frac{\partial d^{+}(\mathbf{v},\Omega)}{\partial \Omega_{kl}} + \xi_{\Theta}^{-}(\mathbf{v}) \, \frac{\partial d^{-}(\mathbf{v},\Omega)}{\partial \Omega_{kl}} \tag{2.74}$$

$$\text{and} \quad \tfrac{\partial d^{\pm}(\mathbf{v},\Omega)}{\partial \Omega_{kl}} = \quad [\mathbf{v} - \mathbf{w}^{\pm}]_m \left[\Omega(\mathbf{v} - \mathbf{w}^{\pm})\right]_l \, . \tag{2.75}$$

where $\xi_{\Theta}^{\pm}(\mathbf{v})$ and $d^{\pm}(\mathbf{v},\Omega) = d_E^2(\mathbf{v},\mathbf{w}^{\pm},\Omega)$ is determined analog to standard GLVQ (2.41), (2.35), and (2.36), respectively. For learning rate $\alpha_{\Omega}$ the same argumentation applies like for $\alpha_{\lambda}$ in GRLVQ [Schneider et al., 2009]. The natural number $m$ gives the dimensionality of the resulted mapping space: $\phi(\mathbf{v}) = \Omega\mathbf{v}$ with $\phi : \mathbb{R}^D \to \mathbb{R}^m$.

The GMLVQ is versatile in application and interpretation of the learned model. For example the matrix

$$\Lambda = \Omega^T \Omega \tag{2.76}$$

can be interpreted as *classification correlation matrix* [Kaden et al., 2014]. The matrix $\Lambda$ involves beside the weighting of each feature on the diagonal, the classification correlation values between the features besides the diagonal. This means if $\Lambda_{kl} > 0$, the dimension $k$ and $l$ are positive correlated in terms of classification improvement and if $\Lambda_{kl} < 0$, they are negative correlated. The GMLVQ is applied on the *Coffee data* set to illustrate the mapping $\Omega\mathbf{v}$ and interpretation of the classification correlation matrix $\Lambda$ in Application 2.1.

Like the normalization of the relevance profile a normalization constraint according $\Omega$ or $\Lambda$ can be required, respectively. Analog to (2.67), the trace of $\Lambda$ is set to one, i. e.

$$\sum_{i=1}^{D} \Lambda_{ii} = 1 \tag{2.77}$$

This constraint can be transferred to $\Omega$ by:

$$\Lambda_{ii} = \sum_{k=1}^{m} \Omega_{ki}^2 \tag{2.78}$$

and therefore, each element of $\Omega$ have to be divided by:

$$\Omega_{lj} \leftarrow \frac{\Omega_{lj}}{\left(\sum_{k=1}^{m} \sum_{i=1}^{D} (\Omega_{ki})^2\right)^{\frac{1}{2}}} \, . \tag{2.79}$$

This normalization does not affect the value of the cost function and can be applied after each iteration step [**?**].

Another application possibility is to use the mapping matrix $\Omega$ for visualization of the

classification result denoted as limited rank GMLVQ (LiRaM LVQ, [Bunte et al., 2012]). For that purpose the matrix has to be initialized rectangular with the mapping dimension $m = 2$ or $m = 3$ (see Fig. 2.12 in Appl. 2.2). It should be emphasized that $\Omega$ differs from the mapping matrix determined by the Principle Component Analysis (PCA, [Haykin, 1994]). In general, the PCA method is an eigenvalue decomposition of the data covariance matrix. Afterwards, the first two or three eigenvectors are used to map the data and, hence, to visualize the data set. Yet, the PCA can be seen as unsupervised methods since only the data points get into account. Beside the PCA further and more established visualization methods exists like Multi-Dimensional Scaling (MDS), Sammon Mapping or the t-Distributed Stochastic Neighbor Embedding (t-SNE) [v. d. Maaten et al., 2009]. However, all of them are unsupervised methods. In contrast, the mapping $\Omega$ in GMLVQ is determined under the aspect of good class separation and thus, it is learned supervised. An illustration of such a mapping can be found in Application 2.2.

Further, in [Biehl et al., 2009] it is pointed out that the number of non-vanishing eigenvalues of $\Lambda$ and, therefore, the number of relevant eigendirections are very small. Moreover, there is a slight tendency of the squared matrix $\Omega$ (i. e. $m = D$) to converge to a degenerated state where the rows represent the first eigenvector. Thus, despite the huge number of free parameters, the solution of GMLVQ tends to yield simple. In face of this insights, the learning of a huge number of independent parameters like in GMLVQ can course instability. Therefore, the Enhancement-GMLVQ was developed to integrate knowledge about the dependency of the dimension in the learning scheme, e. g. the lateral dependency of the dimensions in functional data. In section 4.1.2 this idea is described more detailed.

A useful tool in classification is the feature extraction, i. e. beside a good classification the model identifies the relevant features considering the class separation. On the one side, in many application the number of features are directly correlated with explicit costs. On the other side, the classification rate may depend on the number of used features. Thus, a compromise between classification rate and number of used features is necessary. On pages 67ff we consider possibilities for obtaining a sparse relevance profile or classification correlation matrix.

In the following, different aspect of the GRLVQ and GMLVQ are demonstrated on the Coffee data set (see section 1.2).

**Application 2.1** (Relevance Learning in GLVQ)**.**

**data set** Coffee

**methods** GRLVQ, GMLVQ

**parameter settings**

| | |
|---|---|
| number of prototypes | $N_W = 5$ (one per class) |
| distance functions | $d_E^2(\mathbf{w}, \mathbf{v}, \lambda)$, $d_E^2(\mathbf{w}, \mathbf{v}, \Omega)$ |
| learning rates | $\alpha_W = 0.0001$ |
| | $\alpha_\lambda = 0.1 \cdot \alpha_W$ |
| | $\alpha_\Omega = 0.1 \cdot \alpha_W$ |

**experimental settings**

| | |
|---|---|
| number of training/test data | $5,000/20,000$ |
| epochs | 5,000 |
| initialization | prototypes: mean spectra per class |
| | relevance profile: $\lambda_i = \frac{1}{256}$ |
| | mapping matrix: $\Omega = \mathbb{I}_{256}$ |

**results in numbers**

| **accuracy** | GLVQ | GRLVQ | GMLVQ |
|---|---|---|---|
| training | 76.8% | 84.0% | 87.0% |
| test | 76.8% | 83.9% | 86.6% |

**results in figures**

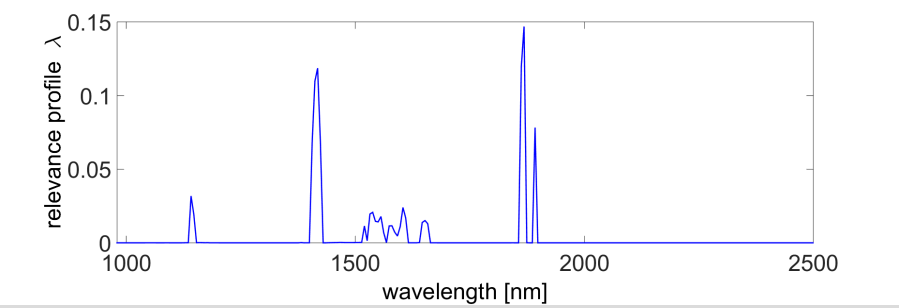


**Figure 2.5**: Relevance profile $\lambda$ learned by the GRLVQ.

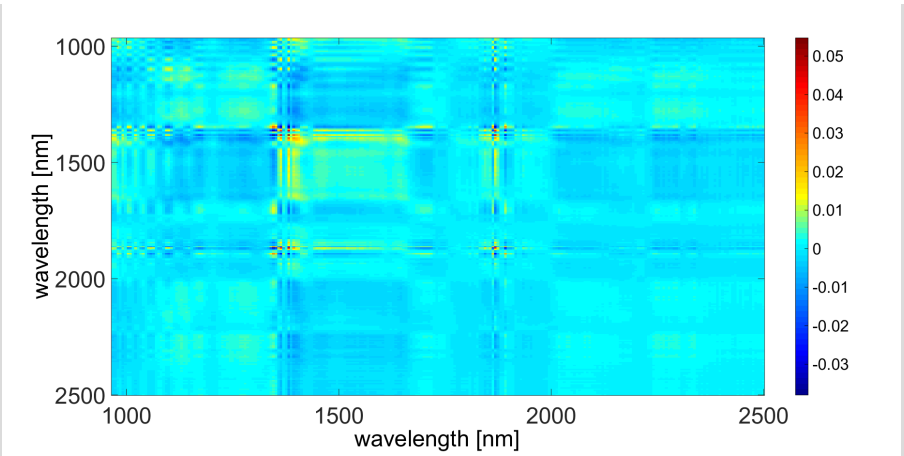


**Figure 2.6**: Relevance matrix $\Lambda$ learned by the GMLVQ.

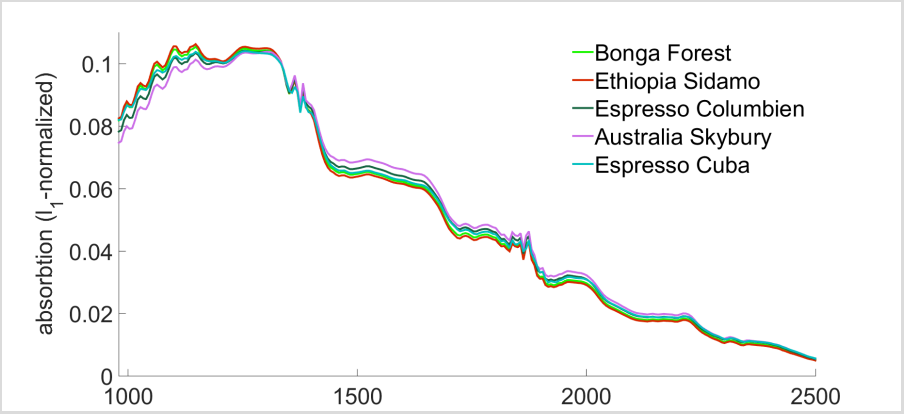


**Figure 2.7**: Original class mean spectra $\bar{\mathbf{v}}$

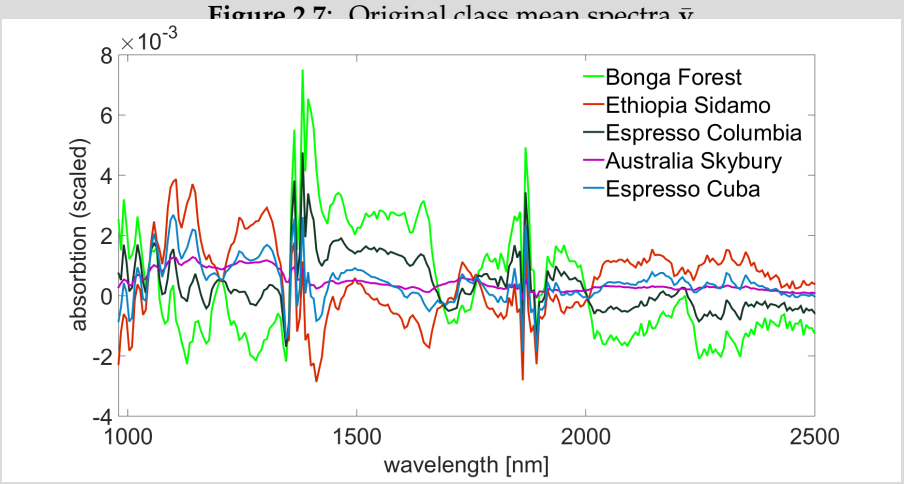


**Figure 2.8**: Mapped mean spectra by $\Omega\bar{\mathbf{v}}$

**resume** The classification of the Coffee spectra is challenging. Yet, the data set is homogeneous, due to nearly the same training and test accuracies. The dimension of the spectra is 256 and therefore, it is a high dimensional data set. The GLVQ get only about $77\%$ accuracy. Moreover, applying GRLVQ we get up to $87\%$, whereat the GMLVQ reaches the best accuracy. In the relevance profile in Fig. 2.5 only a few wavelengths seems to be important for the classification. Only $44$ wavelengths have a higher relevance value than $10^{-4}$. Yet, the relevance values of the other wavelength do not vanishing. To obtain a more sparse relevance profile a regularization term can be added (see page 67). The same behavior of small but non-vanishing entries occurs for the classification correlation matrix $\Lambda$ or the mapping matrix $\Omega$, respectively. Yet, $\Lambda$ provides some further information about the interaction of the wavelengths (see Fig. 2.5), e. g. the wavelengths in the interval $1350 - 1400nm$ and $1850 - 1875nm$ are highly correlated which each other. Moreover, in Fig. 2.7 and 2.8 the original mean spectra and the mean spectra mapped using the matrix $\Omega$ are depicted. The original data are difficult to distinguish visually. In contrast, the discrimination of the mapped mean spectra is obviously better.

**Application 2.2** (Visualization with GMLVQ (LiRaM LVQ))**.**

**data set** Coffee

**methods** PCA, GLVQ, GMLVQ

**parameter settings**

| | |
|---|---|
| number of prototypes | $N_W = 5$ (one per class) |
| distance function | $d_E^2(\mathbf{w}, \mathbf{v}, \Omega)$ |
| learning rates | $\alpha_W = 0.0001$ |
| | $\alpha_\Omega = 0.1 \cdot \alpha_W$ |
| mapping dimension | $m \in \{2, 256\}$ (note: $D = 256$) |

**experimental settings**

| | |
|---|---|
| initialization | prototypes: mean spectra per class |
| | relevance profile: $\lambda_i = \frac{1}{256}$ |
| number of training/test data | 5.000/20.000 |
| epochs | 5000 |

**results in numbers**

| | GLVQ | GMLVQ | |
|---|---|---|---|
| m | - | 2 | 256 |
| test accuracy | 83.3% | 88.5% | 89.0% |

**results in figures**

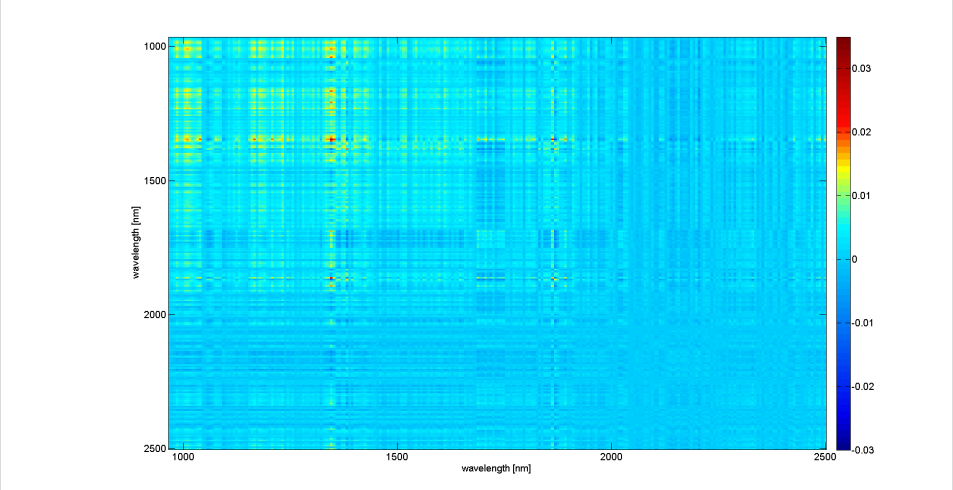


**Figure 2.9**: Classification correlation matrix $\Lambda$ for GMLVQ with the mapping dimension $m = 2$

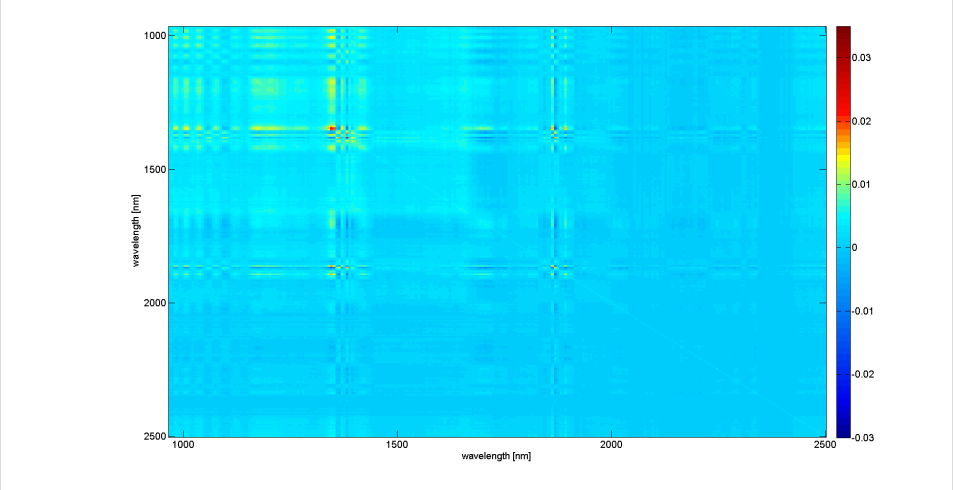


**Figure 2.10**: Classification correlation matrix $\Lambda$ for GMLVQ with the mapping dimension $m = D = 256$

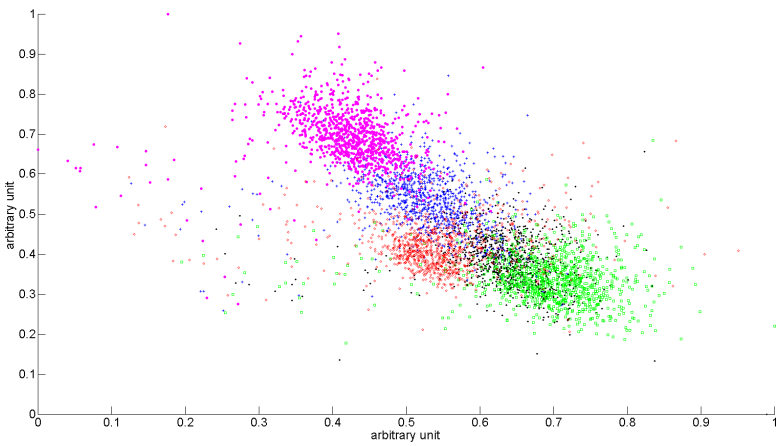


**Figure 2.11**: Mapping of the the coffee spectra by the first two eigenvectors determined by the unsupervised PCA

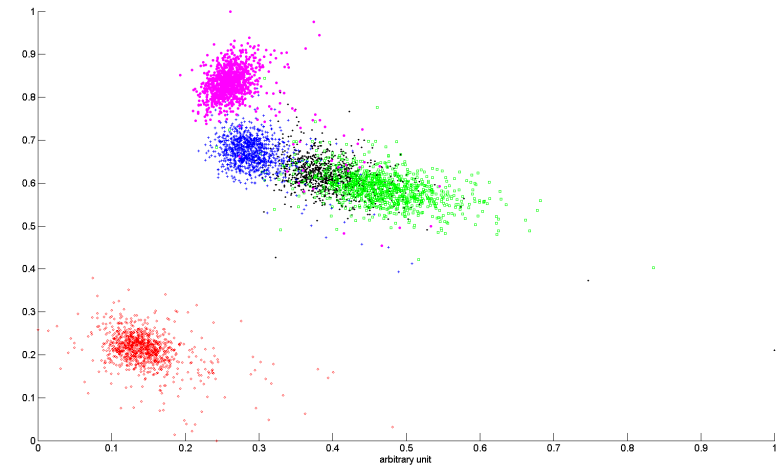


**Figure 2.12**: Mapping of the spectra by $\Omega\mathbf{v}$ and $\Omega \in \mathbb{R}^{2\times 256}$

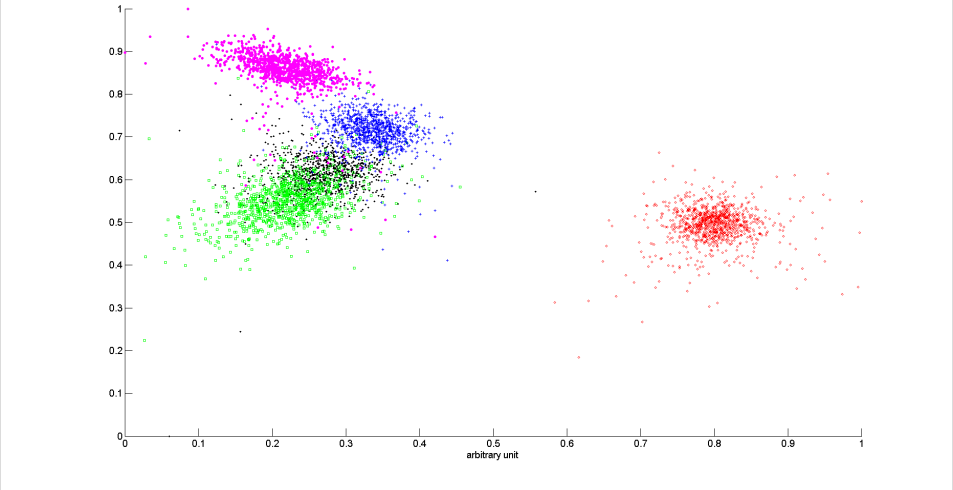


**Figure 2.13**: Projection of the data according to the eigenanalysis of $\Lambda_2$ defined by the full rank mapping $\Omega_2 \in \mathbb{R}^{256\times256}$

**resume** As mentioned in the Application 2.1 the classification of the coffee spectra is challenging. A standard PCA with mapping on the first two main components underline this certainty (see Fig. 2.11). Yet, applying LiRaM LVQ we obtain better results. First the mapping dimension is limited to $m = 2$ and the visualization can directly be performed by $\Omega_1\mathbf{v} \rightarrow \tilde{\mathbf{v}} \in \mathbb{R}^2$. Then, we learned a full rank matrix $\Omega_2$ and visualized the data set by the first two principle components of $\Lambda_2 = \Omega_2^T\Omega_2$. Visually, this two mappings are very similar (see Fig. 2.12 and 2.13), which is also reflected by the similar accuracies of GMLVQ with the limited and full rank mapping matrix. Yet, compared to the unsupervised PCA visualization the supervised visualization with $\Omega_1$ or $\Omega_2$ is considerably better. Especially, the *red* class (Espresso Cuba) separates perfectly from the other 4 classes. Further, it can be seen that the green (Bonga Forest) and *black* (Espresso Columbia) classes are hard to separate. Moreover, we observe that the GMLVQ with the full rank $\Omega_2$ leads to a smother correlation matrix (see Fig. 2.9). Hence, the essential structure information is preserved in $\Omega_1$.

### Distance Adaptation in DK-GLVQ

Many kernel are parameterized like the RBF kernel. These parameters have to be adjusted in advance for good model performance. This parameter determination can be very costly and challenging. In the SVM a suitable parameter combination is usually identified by the *systematic search*, i. e. all suitable parameters are simple sequential attempted. Other approaches estimate a parameter initialization. Yet, the estimation is not always satisfactory for the applications ([**?**, **?**]). One advantages in DK-GLVQ

is the possibility of automatically determination of the kernel parameters applying the stochastic gradient descent. During the learning the kernel parameter is adapted beside the prototypes like it is known from the relevance/matrix learning. Furthermore, the kernel parameter can be localized, i. e. each prototype is assigned by its own parameter.

We illustrate this strategy for the RBF kernel in DK-GLVQ; The update for the kernel parameter $\varsigma_j$ of prototype $\mathbf{w}_j$ is:

$$\begin{aligned} \varsigma_j &\leftarrow \varsigma_j - \alpha_\varsigma \cdot \Delta\varsigma_j \\ \text{with} \quad \Delta\varsigma_j &= \xi_\Theta^+(\mathbf{v}) \, \frac{\partial d^2_{RBF}(\mathbf{v}, \mathbf{w}^+, \varsigma)}{\partial \varsigma_j} + \xi_\Theta^-(\mathbf{v}) \, \frac{\partial d^2_{RBF}(\mathbf{v}, \mathbf{w}^-, \varsigma)}{\partial \varsigma_j} \\ \text{and} \quad \frac{\partial d^2_{RBF}(\mathbf{v}, \mathbf{w}_k, \varsigma_k)}{\partial \varsigma_j} &= \begin{cases} \frac{d_E(\mathbf{v}, \mathbf{w}_k)}{\varsigma_j^3} \, d^2_{RBF}(\mathbf{v}, \mathbf{w}_k, \varsigma_k) & \text{, if k=j} \\ 0 & \text{, else} \end{cases} . \end{aligned}$$

where $\xi_\Theta^\pm(\mathbf{v})$ is determined analogously to the standard GLVQ (2.41) applying the kernel distance. Experiments show, the learning rate $\alpha_\varsigma$ has to be chosen carefully (see Application 2.3).

Moreover, the relevance learning of GRLVQ/GMLVQ can also be plugged in the DK-GLVQ. Again, we demonstrate the relevance learning for the RBF kernel (2.62). Thereby, we obtain

$$\kappa_{RBF}(\mathbf{v}, \mathbf{w}, \Omega) = e^{-\sqrt{(\Omega(\mathbf{v}-\mathbf{w}))^2}} \tag{2.80}$$

Thus, the RBF parameter $1/(\varsigma^2)$ is replaced by the mapping matrix $\Omega$. If $\Omega$ is a diagonal matrix, the diagonal corresponds to the relevance profile. It has to be pointed out that it can be not longer guaranteed that the kernel $\kappa_{RBF}$ is universal: The property depends on the classification correlation matrix $\Lambda = \Omega^T\Omega$. During the learning we can only ensure that $\Lambda$ is positive semi-definite, but not to be strict positive definite and thus $\sqrt{(\Omega(\mathbf{v}-\mathbf{w}))^2}$ may be not longer a metric. Numerically, the probability of $\Omega$ degenerating is nearly zero.

But for all that, choosing $m \in \{2, 3\}$ the resulted model can be additional visualized [Kästner et al., 2012]. An example of the DK-GMLVQ is presented in Application 2.4. Thereby, the visualization ability of the DK-GMLVQ is pointed out additionally.

**Application 2.3** (DK-GLVQ with parameter adaptation)**.**

**data set** Palau

**methods** GLVQ, DK-GLVQ, SVM

**parameter settings**

(DK)-GLVQ

| | |
|---|---|
| number of prototypes | $N_W \in \{2, 9\}$ |
| distance function | $d_E^2(\mathbf{v}, \mathbf{w}_j)$, $d_{RBF}(\mathbf{v}, \mathbf{w}_j, \varsigma_j)$ |
| learning rates | $\alpha_W = 0.01$ |
| | $\alpha_\varsigma = 10^{-8}$ |
| transfer function parameter | $\Theta = 1$ |

SVM

| | |
|---|---|
| kernel | $\kappa_{RBF}$ |
| kernel parameter | $\varsigma = 0.5$ (systematic search) |
| penalty value | $C = 1,000$ |

**experimental settings**

(DK)-GLVQ

| | |
|---|---|
| training set | $N_{V_{train}} = 5,000$ |
| test set | $N_{V_{test}} = 20,000$ |
| epochs | GLVQ: 1000 DK-GLVQ: 5000 |
| initialization | prototypes assigned to a data point randomly |
| | kernel parameter: $\varsigma = 0.05$ |

**results in numbers**

| | GLVQ | | DK-GLVQ | | SVM |
|---|---|---|---|---|---|
| $N_W$ | 2 | 9 | 2 | 9 | 16 |
| accuracy | 67.1% | 97.8% | 95.5% | 99.5% | 100% |

**results in figures**

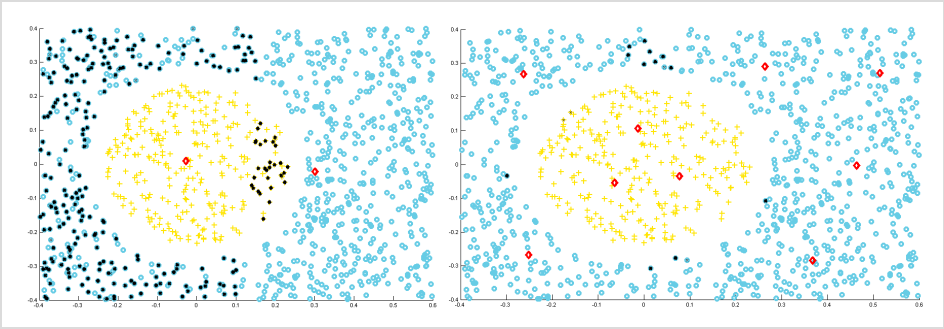

**Figure 2.14**: GLVQ results with 2 prototypes (left) and 9 prototypes (right); legend: data points of class 1 - yellow + and of class 2 - blue ○, misclassified samples - black ×, prototypes - red ◇

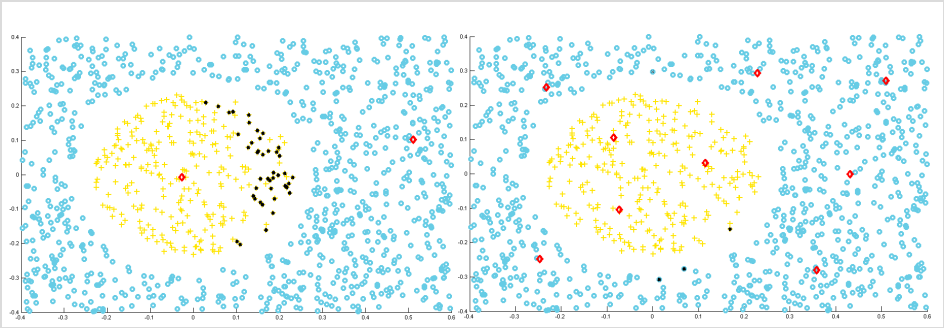

**Figure 2.15**: DK-GLVQ results with 2 prototypes (left) and 9 prototypes (right); legend: data points of class 1 - yellow + and of class 2 - blue ○, misclassified samples - black ×, prototypes - red ◇

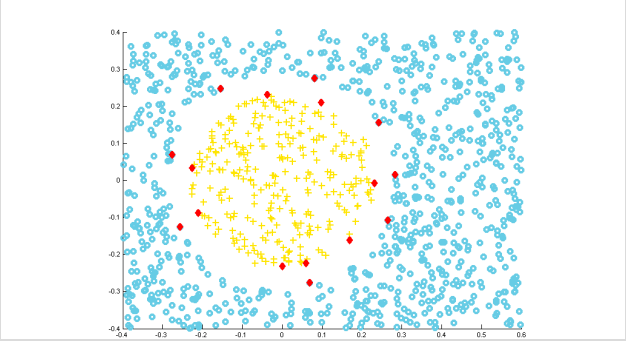

**Figure 2.16**: SVM result with 16 SV; legend: data points of class 1 - yellow + and of class 2 - blue ○, SV - red ◇

**resume** This exemplary two-dimensional example shows that a linear classification in $(V, d_E^2)$ is not possible, however a linear separation in $(V, d_{RBF})$. The GLVQ using only two prototypes (linear model) together with the quadratic Euclidean distance can not solve the classification problem. One possibility is to enlarge the number of prototypes per class. Nine prototypes are adequate to solve the problem. Yet, another possibility is to apply a kernel based distance like the $d_{RBF}(\mathbf{v}, \mathbf{w}_j, \varsigma_j)$ combined with localized parameter learning on $\varsigma_j$. The DK-GLVQ with two prototypes ends up with a satisfying result. The values for the parameters are $\varsigma_1 = 0.03$ and $\varsigma_2 = 0.07$. In the experiments, it turns out that the parameter learning is very sensible concerning the learning rate. Thus, the parameters have to be adapted very carefully (learning rate $\alpha = 10^{-8}$). The sparsest model of the SVM[a] ends up with 16 SV and a training accuracy of $100\%$.

[a] The experiments are done using the libSVM package for Matlab®

**Application 2.4** (Visualization with DK-GMLVQ).

**data set** Coffee

**methods** GMLVQ, DK-GMLVQ, SVM

**parameter settings**

| | |
|---|---|
| (DK-)GMLVQ | |
| number of prototypes | $N_W = 5$ (one per class) |
| distance function | $d_E^2$, $d_{RBF}^{\Omega}$ |
| learning rates | $\alpha_W = 0.01$ |
| | $\alpha_\Omega = 0.01\ \alpha_W$ |
| transfer function parameter | $\Theta = 10$ |
| SVM | |
| kernel | $\kappa_{RBF}$ |
| kernel parameter | $\varsigma_1 = 7$, $\varsigma_2 = 5$ (systematic search) |
| penalty value | $C_1 = 200,000$, $C_2 = 10,000$ |

**experimental settings**

| | |
|---|---|
| training set | $N_{V_{train}} = 5,000$ |
| test set | $N_{V_{test}} = 20,000$ |
| epochs | 10,000 |
| initialization | prototypes: mean spectra |
| | $\Omega$ : m=2 - random, m=256 - identity matrix |

**results in numbers**

| | GMLVQ | | DK-GMLVQ | | SVM | |
|---|---|---|---|---|---|---|
| $N_W$ | 5 | | 5 | | 827 | 1000 |
| m | 2 | 256 | 2 | 256 | - | - |
| test accuracy | 88.5% | 89.0% | 90.8% | 91.4% | 93.6% | 94.7% |

| confusion matrix | coffee types | black | green | blue | magenta | red |
|---|---|---|---|---|---|---|
| GMLVQ (m=256) | black | 81.5 | 14.1 | 4.4 | 0.1 | 0 |
| | green | 26.0 | 73.0 | 1.0 | 0 | 0 |
| | blue | 5.7 | 0.2 | 92.9 | 1.2 | 0 |
| | magenta | 0.8 | 0.1 | 1.6 | 97.5 | 0 |
| | red | 0 | 0 | 0 | 0 | 100 |
| DK-GMLVQ (m=256) | black | 82.8 | 15.5 | 1.5 | 0.1 | 0 |
| | green | 17.5 | 82.2 | 0.3 | 0 | 0 |
| | blue | 4.7 | 0.1 | 94.1 | 1.1 | 0 |
| | magenta | 0.7 | 0.1 | 1.5 | 97.7 | 0 |
| | red | 0 | 0 | 0 | 0 | 100 |

**results in figures**

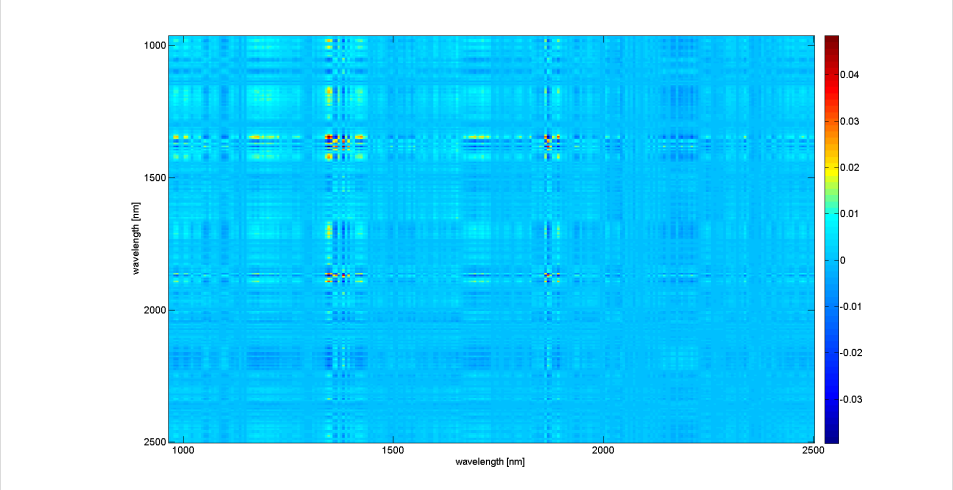


**Figure 2.17**: Classification correlation matrix $\Lambda_1$ for DK-GMLVQ with the mapping dimension $m = 2$

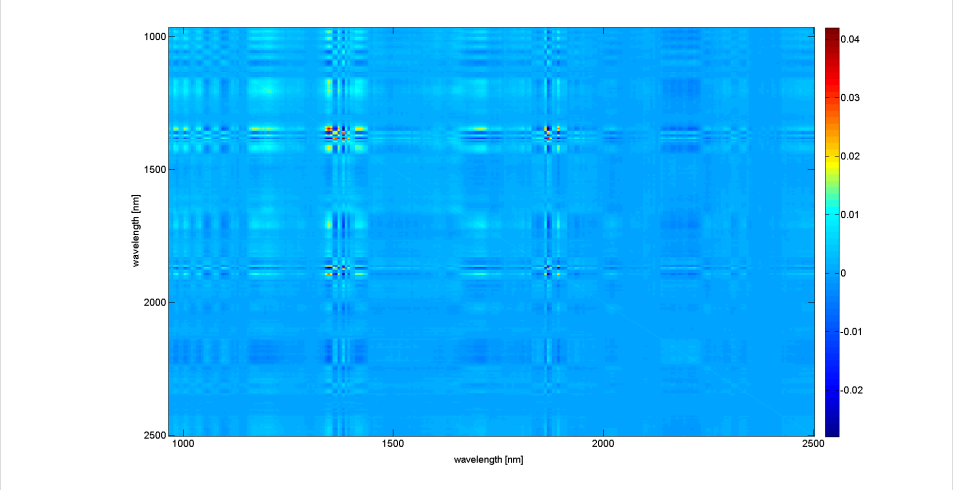


**Figure 2.18**: Classification correlation matrix $\Lambda_2$ for DK-GMLVQ with the mapping dimension $m = D = 256$

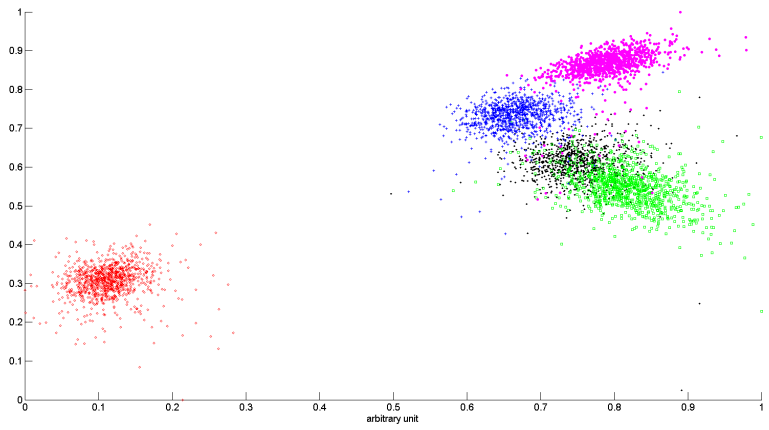


**Figure 2.19**: Visualization with DK-GMLVQ: mapping of the spectra by $\Omega_1\mathbf{v}$ and $\Omega_1 \in \mathbb{R}^{2\times 256}$

**resume** The visualization of the coffee spectra with the Euclidean distances was already done in Application 2.2. Compared to GMLVQ, the DK-GMLVQ with the weighted RBF-Kernel $\kappa_{RBF}(\mathbf{v}, \mathbf{w}, \Omega)$ obtain a better accuracy with a very sparse model, i. e. using only one prototype per class. The SVM with the RBF-Kernel get a better accuracy of about $2-3\%$. However, up to $1000$ support vectors are needed. This corresponds to $20\%$ of the training data. The improvement of the DK-GMLVQ compared to the GMLVQ reflects in better separation between the green (Bonga Forest) and *black* (Espresso Columbia) class, which on the one hand can be seen in the confusion matrix and on the other hand in the mapped data points by $\Omega$ (compare Fig.2.12 and Fig. 2.19). In Fig. 2.19 the mapped data points with the limited ranked $\Omega_1 \in \mathbb{R}^{2\times 256} : \Omega_1\mathbf{v} \rightarrow \tilde{\mathbf{v}}$ obtained by DK-GMLVQ is shown.

### Feature Sparsity in Relevance and Matrix Learning

Especially for high dimensional data sets, a typical relevance profile in GRLVQ/DK-GRLVQ consists of a few considerable large values ans a large number of small but non-vanishing values . However, in sparse models with respect to the number of used features this effect is adverse. Thus, the values of features, which are not decisively contribute to the classification result, should be forced to vanish. To achieve this goal the cost function can be extended by an additive term forcing a sparsity constraint $E_s(\lambda)$:

$$E_{sGRLVQ} = E_{GRLVQ}(W, \lambda) + \gamma(\iota)\ E_s(\lambda) \tag{2.81}$$

with the weighting $\gamma(\iota)$ of the sparsity condition depending on the iteration epoch $\iota$.

The ansatz for the sparsity condition $E_s(\lambda)$ can be diverse. One approach is the formulation of the sparsity constraint by an entropy term known form the information theory. In this context, the entropy $H(\lambda)$ measures the average amount of information of the relevance profile, i. e. the value $H(\lambda)$ is minimal, iff for one single index $k$ apply $\lambda_k = 1$ and $\lambda_j = 0 \; \forall j \neq k$. The fundamental and most famous entropy is the Shannon entropy:

$$E_s^H(\lambda) = -H_{Shannon}(\lambda) = -\sum_{i=1}^{D} \lambda_i^2 \log(\lambda_i^2) \tag{2.82}$$

The Shannon entropy is differentiable with respect to $\lambda_k$. Thus, a optimization by the SGD is feasible. The additional update for the relevance vector is based on the derivate:

$$\frac{\partial E_s^H(\lambda)}{\partial \lambda_k} = -\left(\log(\lambda_k^2) + 1\right). \tag{2.83}$$

The goal of the sparsity is to achieve $\lambda_k = 0$ for a large number of $l$. However, the Shannon entropy and its derivate are numerically sensitive for small values of $\lambda_k$ due to the logarithmic function. This might lead to instable behavior during the learning. Thus, other differentiable entropy functions are thinkable, e. g. the Rényi entropy

$$E_s^{R,\alpha}(\lambda) = -\frac{1}{\alpha - 1} \log \left( \sum_{l=1}^{K} ()\lambda_l^2)^\alpha \right) \tag{2.84}$$

or the Tsallis Entropies

$$E_s^{T,\alpha}(\lambda) = -\frac{1}{1 - \alpha} \left( 1 - \sum_{l=1}^{K} (\lambda_l^2)^\alpha \right) \tag{2.85}$$

with their parameter $\alpha > 1$ [8].

Another ansatz is the use of the $l_p$-norm as a sparsity condition. Thereby, the $l_0$-norm counts the number of features which have non-zero relevances:

$$E_s^{l_0}(\lambda) = ||\lambda||_0 = \sum_{i=1}^{D} (1 - \delta_{\lambda_i,0}) \tag{2.86}$$

with $\delta_{i,j}$ is the Kronecker delta (2.29). Unfortunately, this discrete function is non-convex and hard to minimize (NP-hard). Therefore, the $l_0$-norm is replaced by an higher order concerning $p$ [Bach et al., 2011]. If $p = 2$, the least squares of the relevance

[8] A further application of the entropy for sparsity can be found for the features or structural sparsity of the functional GRLVQ on page 94ff.

profile is considered

$$E_s^{l_2}(\lambda) = ||\lambda||_2 = \sum_{i=1}^{D} \lambda_i^2 \tag{2.87}$$

The cost function term $E_s^{l_2}$ is easy to handle, because the function is convex. Nevertheless, the minimization of (2.81) can be done by stochastic gradient descent, because $||\lambda||_2$ is differentiable according $\lambda_k$, too. However, the minimization of the $l_2$-norm is not the same like the minimization of the $l_0$-norm and therefore, is not suitable for the sparsity condition [Bach et al., 2011].

In compress sensing or in the *Least Absolute Selection and Shrinkage Operator* approach (Tib-LASSO, [Tibshirani, 1996]), the sparsity term is relaxed by the $l_1$-norm

$$E_s^{l_1}(\lambda) = ||\lambda||_1 = \sum_{i=1}^{D} |\lambda_i| \tag{2.88}$$

which imply a convex optimization problem. Moreover, in [Donoho, 2004] it was shown that under certain conditions the minimum of the $l_1$-norm is equal to the minimum of the $l_0$-norm. Yet, the $l_1$-norm is not differentiable according to $\lambda_k$ because of the absolute value function. Fortunately, in [Schmidt et al., 2007] can be found a differentiable approximation of the $l_1$-norm and thus the stochastic gradient descent on $E_{sGRLVQ}$ with (2.88) can be applied. In the following a short explanation of this method is given:

As already mentioned, in [Schmidt et al., 2007] a differential approximation for the $l_1$-norm is pinpointed. Therefore, the absolute value $|\lambda_i|$ is divided into

$$|\lambda_i| = \quad (\lambda_i)_+ + (-\lambda_i)_+ \tag{2.89}$$

$$\text{with} \quad (\lambda_i)_+ = \quad \max\{\lambda_i, 0\} \tag{2.90}$$

The equation (2.90) can be approximated by

$$(\lambda_i)_+ \approx \quad \lambda_i + \frac{1}{\zeta}\ln(1 + e^{-\zeta\lambda_i})$$

with the approximation parameter $\zeta$ explained in [Chen and Mangasarian, 1995]. Thus, for the absolute value function the differential approximation

$$|\lambda_i|_\zeta = \frac{1}{\zeta}\ln\left(2 + e^{-\zeta\lambda_i} + e^{\zeta\lambda_i}\right)$$

is obtained. An upper bound for this approximation is

$$| \, |\lambda_i| - |\lambda_i|_\zeta \, | \leq \frac{2\ln(2)}{\zeta} \tag{2.91}$$

Hence, the greater $\zeta$, the better is the approximation.

Consequently, we can apply these formulas to approximate $E_s^{l_1}(\lambda)$ and the partial derivation for the additionally update term of $\lambda_k$ results in

$$\frac{\partial E_s^{l_1}(\lambda)}{\partial \lambda_k} \approx \tanh\left(\frac{\zeta\lambda_k}{2}\right) .$$

It should be noted that the weight $\gamma(\iota)$ of the sparsity term in (2.81) should be slowly increased during learning. Just before the accuracy drastically decreasing, the algorithm should be stopped. At this point a good balance between sparsity and accuracy is obtained.

The idea of sparsity can be applied to the GMLVQ, too. In this case a matrix $l_p$-norm has to be utilized. In the example of the $l_1$-norm, we obtain:

$$||\Omega||_1 = \max_{1 \leq j \leq D} \sum_{i=1}^{m} |\Omega_{ij}| \tag{2.92}$$

The realization on $||\Omega||_1$ of the approximation mentioned above is explained in [Riedel et al., 2013]. Yet, a realization frequently is numerically too complex.

Beside the feature sparsity, the goal of sparse vectors/matrices appears in many applications and problems. Therefore, the methods to obtain sparsity are multifaceted. In this case above described ansatz is only one possibility. More detailed literature with further approaches can be found in [Donoho, 2004], [Hoyer, 2002], or [Olshausen and Field, 1997].

**Application 2.5** (Relevance Learning with sparsity conditions in GRLVQ)**.**

**data set** Coffee

**methods** sGRLVQ with Shannon Entropy and Tib-LASSO

**parameter settings**

| | |
|---|---|
| number of prototypes | 5 (one per class) |
| distance function | $d_E^2(\mathbf{w}, \mathbf{w}, \lambda)$ |
| learning rates | $\alpha_W = 0.0001$ |
| | $\alpha_\lambda = 0.1 \cdot \alpha_W$ |
| approximation constant | $\zeta = 5$ |
| sparsity weighting | $\gamma(0) = 0$ linear increasing to $\gamma(5000) = 0.0025$ |

**experimental settings**

| | |
|---|---|
| initialization | learned GRLVQ model |
| number of training/test data | $5,000/20,000$ |
| epochs | 5,000 |

**results in figures**

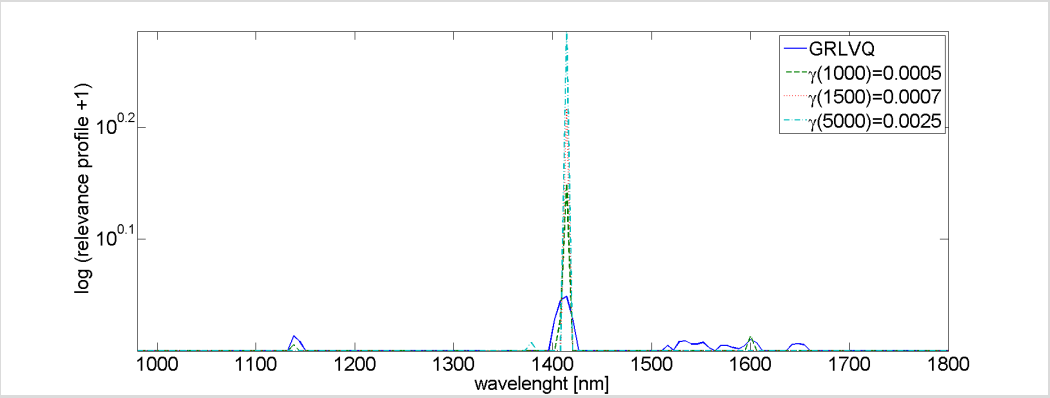


**Figure 2.20**: Relevance profile in different states of sparsity according Tib-LASSO.

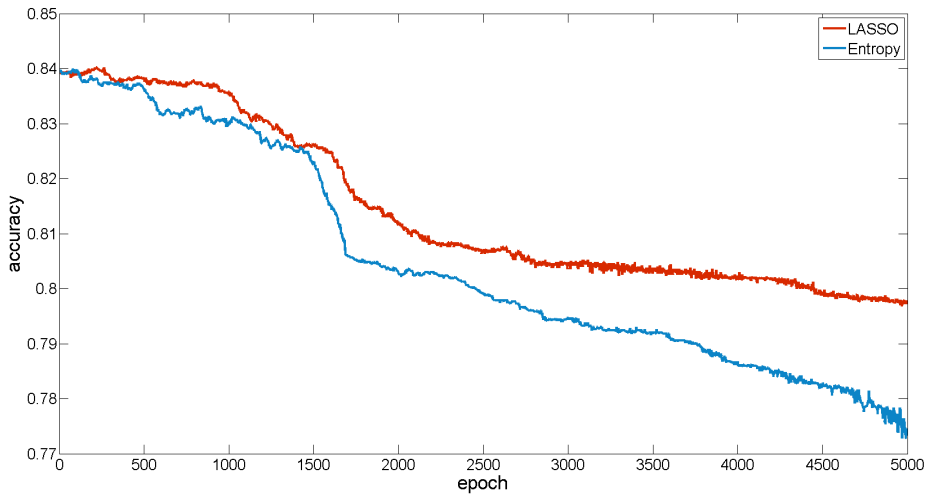


**Figure 2.21**: Development of the accuracy values during sparsity adaptation according Tib-LASSO (red) and entropy (blue).

**resume** With increasing sparsity weight $\gamma$ the relevance profile gets sparse for Tib-LASSO (see Fig. 2.20). The important wavelengthes for distinguishing the coffee types are around $1140\,nm$, $1420\,nm$ and $1600\,nm$. While the resulting relevance profile applying entropy or Tib-LASSO are similar, the accuracy decreases differ. Applying the Tib-LASSO approach, the accuracy keeps longer high values compared to the entropy approach, see Fig. 2.21. Moreover, the entropy based method shows strong instabilities (around $\iota \geq 4750$) if the relevance weights for spectral bands approach zero values at the end of the regularization process.

# Chapter 3

# Deeper Insights into Classification Problems - From the Perspective of Generalized LVQ -

## 3.1 Classification Models

In several practical applications it is too complex or even impossible to develop an exact model for a given problem like for example the distinction of soil conditions via reflectance (hyper-)spectra from the field of remote sensing. Yet, a set of such tasks can be formulated as decision or specific classification problems, respectively. Moreover, if we do so, a classification model can be generated by *learning from examples* as it is known from Machine Learning strategies.

Utilization of these strategies can be found in nearly all application fields: food industry - fat or quality level detection; automotive industry - decision about airbag deployment in a crash scenario; production - stopping of a band-conveyor in an episode; remote sensing - kind of soil conditions in an area; or in medical sciences - rating of the advanced case of a patient to name just a view. The huge amount of application fields entail a high variety of classification models in Machine Learning. The approaches can be adapted from basic statistics or statistical learning, information theory, optimization theory or by other sciences like biology or physics. In many cases, the methods combine several theories and an assignment is not unique.

Basic statistical methods are the Linear Discriminant Analysis (Linear DA) and the extensions or generalizations thereof Quadratic DA or Fisher DA [Bishop, 2006]. The DA determines the parameters of a predefined discriminant function, such that the separation of the classes is as best as possible. In case of the Linear DA, the discriminant function is linear and, therefore, it should only be applied for simple linearly separable problems.

Hidden Markov Models (HMM, [Bishop, 2006]) are stochastic models, whereby the parameters of the stochastic process have to be determined. Further, the HMM can be seen as a graph, where the nodes are the observations and the edges are equipped with transition probabilities. As already mentioned, the Support Vector Machine is a popular method based on statistical learning theory (see page 45 or appendix A.2). One extension of the SVM is the Relevance Vector Machine, which has a similar cost function like the SVM, but is based on a Bayesian formulation of a linear model.

A further family of methods are the Neural Networks (NN, [Haykin, 1994]). The mathematical models of NN are strongly simplified biological neural networks of the brain. Yet, they also can be seen as graphs like HMM or Decision Trees (DT, [Rokach and Maimon, 2008]). DT are tree-like directed graphs where each node corresponds to a feature. The final leaves (nodes without a child) represent the classes. When a data point is given, it is start in the root node (node without a parent) and the value of the specific feature decide about the next child node. The number of nodes, the features and the value bounds are learned by examples. There exist many heuristics methods . For complex data sets with a lot of features the obtained tree can be huge. Since it might be not manageable, there also exists methods to prune a DF. Further, DT are instable for small data deviations [**?**].

It can be distinguish between mathematically proven or heuristic classification approaches: In this view the classification task can be seen as an optimization problem. Yet, there are different ways to formulate the cost function. On the one side the formulation depends on the concrete goal like optimization of the accuracy or other statistical evaluation measures (see Sec. 3.3), but on the other side also on the solution strategy on the other hand. In general, the direct maximization of the classification accuracy is a discrete optimization problem which is NP-hard. Therefore, various approaches search for a continuous approximation, which is easier to handle. For example for SVMs a quadratic cost function with convex constraints has to be optimized. Thus, the solution of a SVM is unique and a lot of efficient optimization strategies exist. The SVM is mathematically established. On the other side, the DT are intuitive models with a lot of mathematical expertise. Anyway, the decision trees are heuristics. As shown in Section 2.2.1, the basic LVQ-methods like LVQ 1 or LVQ 2.1 are heuristics. Yet, `Sato & Yamada` finding a according cost function by keeping the principle of shifting of the prototypes and therefore, the generalization of the LVQ, the GLVQ, is not longer heuristically.

The prototype based Vector Quantization methods described in the prevoius section are adaptive models. Thereby, the prototypes are adapted according to a learning rule during the training process. This procedure is called *online learning*, due to the fact that not all data have to be known at the beginning of the learning. New data points can be considered any time. Other approaches directly obtain a solution by using all data points at once. If a new point is available, the whole model has to be determined anew.

Thus, the problem of classification is very comprehensive and many strategies exist regarding different perspectives. In this thesis, a full summary cannot be given. Nevertheless, several aspects of classification problems to provide better understanding of the multifariousness of the classifiers are discussed.

**Parts of the section are based on**

*M. Kaden, M. Lange, D. Nebel, M. Riedel, T. Geweniger, and T. Villmann:* Aspects in Classification Learning - Review of Recent Developments in Learning Vector Quantization. *Foundation of Computing & Decisions Sciences, 2014.*

## 3.2 The Classification Task

Although the classification task seems to be well defined, a lot of different methods exist to solve such problems as mentioned in the last pages. Reasons therefore are multifaceted. For example: the kinds of underlaying data and classes might be miscellaneous or the classification goal is not uniform.

In this section, a possible categorization of classification problems is given:

**A)** goal of the classification task,

**B)** data characteristics, and

**C)** class characteristics.

In the following, these three categories are described in more detail.

### A) Goal of the Classification Task

Beside the evaluation measure like the classification accuracy, different aspects affect the performance in practice, e. g. the model complexity, the interpretability or the usability for real time applications [Backhaus and Seiffert, 2014]. Thus, at the beginning the goal, the limits and the constraints of the given problem have to be well defined.

Thereby, the following aspects about the task should be taken into account for choosing a classifier:

**model complexity** In most applications there are no restrictions concerning the training capacity or runtime. However, the chosen model should be applicable and suitable for the given problem. Thereby, a statement of Einstein should be considered: 'Everything should be made as simple as possible, but not simpler'.

Further, since the resulting model might be implemented in an overall system with restricted memory capacity, the model size is restricted. Thus, the classifier model should be sparse.

**costs of training and testing** In praxis the training costs (in terms of time) are often non-critical as long as they are finite. If not, the chosen classifier should at least be as simple as possible concerning the calculation time. The testing costs are often more critical, especially in real time applications. Thus, additionally to the complexity of the resulting model the calculation time might be restricted as well.

**interpretability** Users in several areas of application have difficulties to accept so called *black box models* like Neural Networks. In *black box models* the goal of the concrete task might be achieved, but the learning process or the resulting model are not comprehensible for the user. Thus, the interpretability of the training procedure and also the resulting model can be an important aspect for acceptance in practical applications.

**robustness** Small deviations in the input data like adding or removing training data points should not leas to a drastic change of the model. A robust model is not sensitive according small modifications which is a crucial aspect in many applications.

**extraction of extra information about data** Beside solving the classification task, gaining further information about the data might be relevant for the user as well. Examples include feature extraction, classification certainty or visualization ability of the resulting model.

### Data Properties

Not only the variety of applications is high, also the variety of data sets. In general, data sets distinguish from each other in several aspects like the source, the complexity, dimensionality, the reason/goal for capturing of the data, the goal for analysis and so one.

In this category we can differ the following subgroups, which influence each other.

**type of data** An important issue is how the data information are given. In most applications each single data point is available. However, sometimes only the (dis-) similarity between the data might be known. If data exist explicitly, then the features can be vectorial, strings, intervals or other categories or objects. This issue is important for the choice of the classification method. Often an algorithm can only handle one type of data.

**separability** In general, it is distinguished between linear and non-linear separable classification problems. Accordingly, the classifiers are subdivided the same way, i. e. models which can only solve linear problems or models which are able to solve non-linear problems as well. Yet, in many applications the data are not

perfectly separable neither linear nor non-linear. Thus, the classifier should be able to handle overlapping classes.

**dimensionality** The number of features of the data is a crucial fact. For vectorial data it can be distinguished between low and high dimensional data. But, currently definition of *low* and *high* is intensively discussed. The difficulties of high dimensional data are due to the *course of dimensionality*[1]. Therefore, for high dimensional classification problems the preprocessing as well as the choice of the dissimilarity measure are important aspects.

**data density/number of data** The data density is directly correlated to the dimensionality and effects the same problems. The number of data points theoretically needed is exponential to the number of features.

**extra knowledge about the data** Sometimes, additionally to the data further information is available. This extra knowledge should be capable of being integrated in the classification model and, accordingly, it should also be considered choosing the classifier. One example for such extra information is available, when classifying hyperspectra in remote sensing. This kind of data are high-dimensional, but further there exist dependency between the neighbor dimensions (wavelengths). Such spectra are also denoted as functional data (see Section 1.2). The elected classifier should be able to integrate this knowledge, e. g. by using an adjusted dissimilarity measure.

### The Class Characteristics

The last properties are the class characteristics. These can be divided into the following subgroups:

**number of classes** Basically, the classification and also the models are divided into two-class or multi-class problems.

**number of data per class** Real world data sets can comprise several classes and in the most cases the number of data per class are not balanced. Many algorithms optimize indirectly the classification accuracy. But if the number of data per class are unbalanced, a accuracy value may misleading.

**labeled and unlabeled data** The labeling of each data point can be very expensive. An example can be again found in the remote sensing. The recorded area by a special spectral camera can be very large. Yet, the number of pixel with characterization

[1]The *course of dimensionality* describes the problem of distance measuring in high dimensional data. In high dimensional space all data are located at the border of a sphere and the Euclidean distance between all data points is nearly identical.

of the soil properties by an expert can be only a small fraction. Thus, sometimes data sets with a few labeled data points and a huge amount of unlabeled points exist.

**crisp or fuzzy label** In remote sensing, the resolution of a pixel can be several hundred meters squared. Thus, in such an area may be more than one vegetation type occur. The assignment to a unique soil conditions/class is not possible. At this juncture the spectra can labeled by fuzzy assignments. A transformation to crisp labels is possible, i. e. a unique assignment to only one class, but afflicted with lost of information.

**extra knowledge/conditions** Beside the fact of unbalanced number of data per class, the classes can also have different weightings in the application. Further, misclassification from class $A$ to class $B$ can be more critical than the other way around. If this extra information about the classes are available or there are special conditions on the resulted model, the classifier should be able to take these into account.

Of course, the three categories with their subgroups interact with each other and can not be seen alone. Moreover, the named aspects do not claim to be comprehensive. Yet, before any classifier is chosen these aspects among others has to be taken into account. A deep analysis of the given classification problem is indispensable.

## 3.3 Evaluation of Classification Results

A mentioned above, the classification accuracy might be not adequate for the validation of the resulted model at all. If the classes are very unbalanced concerning the data distribution, a high accuracy value might be misleading as pointed out in the following small example from the medicine: Given are $2000$ test persons with $100$ of them are disease-ridden and the other $1900$ show no symptoms. A classifier achieve an accuracy of $94.6\%$, which is might be a good result. However, if we have a closer look to the class distribution of misclassification, we can observe that nearly all patients which are disease-ridden are misclassified. In the following we discuss several classification validity measures to overcome those difficulties. First of all we focus on measures for binary class problems.

#### Binary-Class-Problems

The previous small example shows that validation using the accuracy is not suitable in all cases especially if the number of data points per class is unbalanced. Particular for two-class problems exists several more adequate measures are based on the confusion matrix. In the confusion matrix the number of correct and incorrect data points per class are itemized.

Due to the fact that the confusion matrix is often used in medicine, the two classes are typically named *positive* and *negative*, i. e. $\mathcal{C} = \{\oplus, \ominus\}$. We define the set of all data points belonging to the positive class by $V^+ = \{\mathbf{v} \,|\, c(\mathbf{v}) = \oplus\}$ and analog the set of points belonging to the negative class by $V^- = \{\mathbf{v} \,|\, c(\mathbf{v}) = \ominus\}$. They are also called the set of *real positives* and *real negatives*, respectively. All data points which are predicted to belong to the positive/negative class are collected in the set $\hat{V}^+ = \{\mathbf{v} \,|\, \hat{c}(\mathbf{v}) = \oplus\}$ and $\hat{V}^- = \{\mathbf{v} \,|\, \hat{c}(\mathbf{v}) = \ominus\}$, respectively. In Table 3.1 some basic values related to the confusion matrix are depicted.

Each value in Table 3.1 gives only a statement of the false or correct classification rate of one class, respectively. Therefore, several evaluation measures combining these basic values in different ways were introduced. Some common measures are [Fawcett, 2006], [Powers, 2011]:

**Modified Accuracy** The modified accuracy sum up all values of the confusion matrix which are weighted:

$$mAcc = \alpha \cdot tp + \beta \cdot tn - \gamma \cdot fp - \delta \cdot fn$$

with $\alpha + \beta + \gamma + \delta = 1$.

**Matthews correlation coefficient** is the correlation coefficient between observed and

| | **real positive** | **real negative** | | | |
|---|---|---|---|---|---|
| **predicted positive** | **true positive** ($tp$) | **false positive** ($fp$) | $tp + fp = \|\hat{V}^+\|$ | $ppv = \frac{tp}{tp+fp}$ (precision) | $fdr = \frac{fp}{fp+tp}$ |
| **predicted negative** | **false negative** ($fn$) | **true negative** ($tn$) | $fn + tn = \|\hat{V}^-\|$ | $for = \frac{tn}{fn+tn}$ | $npv = \frac{tn}{tn+fn}$ |
| | $tp + fn = \|V^+\|$ | $fp + tn = \|V^-\|$ | $\|V^+\| + \|V^-\| = \|\hat{V}^+\| + \|\hat{V}^-\| = N_V$ | | |
| | $tpr = \frac{tp}{tp+fn}$ (sensitivity) | $fpr = \frac{fp}{fp+tn}$ | | | |
| | $fnr = \frac{fn}{tp+fn}$ | $tnr = \frac{tn}{fp+tn}$ (specificity) | | | |

**Table 3.1**: Confusion matrix of a binary problem with:

| | | | | | |
|---|---|---|---|---|---|
| *tpr* | - | true positive rate | *fpr* | - | false positive rate |
| *fnr* | - | false negative rate | *tnr* | - | true negative rate |
| *ppv* | - | positive predicted value | *fdr* | - | false discovery rate |
| *for* | - | false omission rate | *npv* | - | negative predicted value |

predicted classification:

$$mcc = \frac{tp \cdot tn - fp \cdot fn}{\sqrt{(tp + fp) \cdot (fn + tn) \cdot (tp + fn) \cdot (fp + tn)}} \tag{3.1}$$

with $mcc \in [-1, 1]$ and $mcc = 1$ means a perfect prediction. This coefficient is related to the $\chi^2$ - statistics for a $2 \times 2$ contingency table.

**Geometric mean/G-measure** is the geometric mean between precision and recall: $g = \sqrt{tpr \cdot ppv}$.

**F-measure** is a widely used evaluation measure which is based on the parameter $\beta \in [0, \infty]$:

$$F_{\beta^2} = \frac{(1 + \beta^2) \cdot tp}{(1 + \beta^2) \cdot tp + \beta^2 \cdot fn + fp}. \tag{3.2}$$

The parameter $\beta$ controls the fraction of precision and recall, i. e. if $\beta^2 > 1$, the recall is weighted higher than precision and on the other side if $\beta^2 < 1$, the precision is weighted higher than recall [Rijsbergen, 1979].

For the special case $\beta^2 = 1$:

$$F_1 = \frac{2tp}{2tp + fn + fp}$$

is the harmonic mean between precision and recall.

**van Rijsbergen's effectiveness measure** is

$$E_\alpha = 1 - \frac{1}{\left(\frac{\alpha}{ppv} + \frac{1-\alpha}{tpr}\right)} \tag{3.3}$$

which is based on the $F_\beta$ and can be rewritten as $E_{\frac{1}{1+\beta^2}} = 1 - F_{\beta^2}$

**ROC-Curve** The *Receiver Operation Characteristic* curve (ROC-Curve) is used to visualize the results of classifiers. Thereby, the axis are the false positive rates (fpr) and true positive rates (tpr). One classifier model provides only one point in this diagram. Yet, many classifier depending on parameters influencing the fpr and tpr. The ROC-Curve is obtained by the convex hull of the rates affected by these parameters. A special point in the ROC-Curve is the *Break-Even-Point*. i. e. if recall is the same as precision [Goutte and Gaussier, 2005].

**AUROC** A validity measure for two class problems is based on the ROC-Curve is the area under the ROC-Curve (AUROC). An high AUROC-value represent a good binary class model. The value can be maximal one[Fawcett, 2006].

**PR-Curve** Beside the ROC-Curves, the *Precision-Recall curves* (PR-Curves) are used to visualize the results. Obviously, the axis are precision and recall [T. Landgrebe and Bradley, 2006].

The listed evaluation measures for binary classifiers are the most common. Obviously, many more exists and also a lot of modifications. Moreover, the advantages and disadvantages of them are discussed in many paper ([**?**],[**?**],[Sokolova and Lapalme, 2009a]). The one and only evaluation measure does not exist and the choice depends on the application problem.

### Multi-Class-Problems

The confusion matrix can be provided for multi-class problems, too. Yet, the table includes several values and the comparison of different classifier models might be difficult. Until now, some heuristics exists, which generalize the binary measures for multi-class problems and based roughly speaking on the principle one-versus-all or one-versus-one class, respectively [Sokolova and Lapalme, 2009b, Holt et al., 2010].

A further evaluation possibility is the direct comparison of two classifier results. An exemplar of such an measure is the Kappa-Index.

**Kappa-Index** The $\kappa$-Coefficient is based on the *Kappa Statistic* was established by `Cohen` [Cohen, 1960]. This coefficient is the ratio between the error rate of the classifier model to the error rate of another model. Thus, the *Cohen's* $\kappa$-Coefficient compares two classifier, at which one of them can be a random solution, too. Given are the label assignments of two classifier models $M_i$ as vectors $\hat{\mathbf{c}}_{M_i}(\mathbf{v}) \in [0,1]^{N_C}$ with $||\hat{\mathbf{c}}_{M_i}||_E = 1$ and $i \in \{1,2\}$.

The *Cohen's* $\kappa$-Coefficient $\kappa_C$ is given by:

$$\kappa_C(M_1, M_2) = \frac{P_o - P_e}{1 - P_e} \tag{3.4}$$

where $P_o$ is the relative observed agreement and $P_e$ the expected agreement of the two classifiers [Geweniger, 2012]. These values can be calculated by:

$$P_o = \frac{1}{N_V} \sum_{\mathbf{v} \in N_V} \hat{\mathbf{c}}_{M_1}(\mathbf{v})^T \cdot \hat{\mathbf{c}}_{M_2}(\mathbf{v})$$

and

$$P_e = \sum_{k=1}^{N_C} [\mathbf{p}_{M_1}]_k \cdot [\mathbf{p}_{M_2}]_k \cdot \sum_{\mathbf{v} \in V} [\hat{\mathbf{c}}_{M_1}(\mathbf{v})]_k \cdot [\hat{\mathbf{c}}_{M_1}(\mathbf{v})]_k$$

with the margin probabilities per class

$$[\mathbf{p}_{M_i}]_k = \frac{1}{N_V} \sum_{\mathbf{v} \in V} [\hat{\mathbf{c}}_{M_i}(\mathbf{v})]_k \qquad i \in \{1,2\}$$

The $\kappa_C$-Index has the maximum value of one. A high value means agood agreement of the two classifier and if $\kappa$ is small , i. e. $\kappa < 0.4$, the classifiers have only a slight or no agreement [Sachs, 1992].

Cohon's $\kappa$-Index can only compare two classifier models. An extension is the Fleiss $\kappa$ for more models [Fleiss, 2003]. Further, these indexes were modified for evaluation of fuzzy classifiers: the Fuzzy-Cohen's $\kappa$-Index [Dou et al., 2007] and the Fleiss $\kappa$-Index [Zühlke et al., 2009].

## 3.4 The Classification Task as an Ill-Posed Problem

In the previous subsection the versatility of problem related to classification is described. In this sense, classification could be taken as an ill-posed problem based on the following definition by `Hadamard` ([Hadamard, 1902],[I.Kabanikhin, 2011]:

A problem is denoted as *well-posed* problem if all of the following requirements are fulfilled:

**a** A solution for the problem exists.

**b** The solution is unique.

**c** The behavior of the solution changes continuously with the initial conditions.

If one of these requirements is violated the problem is denoted as *ill-posed*.

In general, the evaluation of the classification task by only looking at the accuracy is not appropriate. As mentioned in the previous subsections other criteria should be considered like e. g. the model complexity, additional problem specific outcome or costs for misclassification. The quality of the model depends on the application and thus, the one and only solution does not exist in classification. In view of all that, the requirement **b** of well-posed problems is not fulfilled and thus, the classification task is an ill-posed problem.

# Chapter 4

# Auxiliary Structure Information and Appropriate Dissimilarity Adaptation in Prototype Based Methods

**Parts of the section are based on**

*M. Kästner, B. Hammer, M. Biehl, T. Villmann:* Functional Relevance Learning in Generalized Learning Vector Quantization. *Neurocomputing, 2012.*

*T. Villmann, M. Kästner, D. Nebel, and M. Riedel:* Lateral Enhancement in Adaptive Metric Learning for Functional Data, *Neurocomputing, 2014.*

## 4.1 Supervised Vector Quantization for Functional Data

Different kind of data structures are exists, e. g. functional data, textural data like DNA sequences and image data set like handwritten digits. The specificity of functional data is the lateral dependency in the features (see Note 1.1). Famous examples are continuous spectra where the features are wavelengths/frequencies or time series, where the data are collected over a time interval. As mentioned, the data sets are usually given as discrete representations, i. e. as vectors $\mathbf{v} = (v_1, v_2, \ldots, v_D)^T$, which in fact are functions $v(t)$ over the wavelengths/resolution or time $t$, respectively. Moreover, in general this data are very high dimensional, i. e. the number of features $D$ can be up to hundreds or thousands.

An overview of functional data analysis (FDA) and its properties can be found in [Ramsay and Silverman, 2006]. The main property distinguishing functional data from other high dimensional vectors is that the sequence of the vector dimensions carries information and cannot be changed without information loss. Thus, this lateral information should be integrated in the learning scheme for improving the results. Several investigations take the feature dependency into account. One idea, which is mentioned in Section (2.4) is to use an adequate distance measure like the Sobolev distance [Villmann, 2007]) or functional kernels [Villa and Rossi, 2005].

Data which are densities are a special kind of functional data, i. e. $\int v(t)dt = 1$ and $v(t) \geq 0 \, \forall t$ or rather for the discrete case $\sum_{k=1}^{D} v_k = 1$ and $v_k \geq 0$. Distance measures handling only densities are the divergences (see Sec. 2.4). The divergences are based

on information theory and measure the distance between two densities. A summary of the integration of divergences in VQ methods can be found in [Haase, 2014].

A mentioned above, only a discretization of the functional data are given in general. A possibility is to approximate the discrete vectors by a linear combination of functions ([Villa and Rossi, 2005], [Rossi et al., 2005]). In a nutshell: given is a basis system of functions $\mathcal{B} = \{b_i(t)|i \in \mathcal{I}\}$ of $\mathcal{L}^1[1, D]$ with the index set $\mathcal{I}$. Examples for such a basis system are Gaussian/Lorentzian functions (see page 87), a Fourier basis system or spline functions [Ramsay and Silverman, 2006]. Thereby, the data points $v(t)$ are a superposition of basis functions [1] of a set $\mathcal{B}$:

$$v(t) = \sum_{l=1}^{|\mathcal{I}|} \beta_l(v(t))\ b_l(t) \tag{4.1}$$

with the linear combinations factors $\beta_l(v(t)) \geq 0$.

A further approach is the presentation of the prototypes as a linear combination of basis functions after the same principle mentioned above:

$$w(t) = \sum_{l=1}^{|\mathcal{I}|} \hat{\beta}_l(w(t))\ b_l(t) \tag{4.2}$$

Thereby, only the linear combination factor $\hat{\beta}_l(w(t))$ of the basis functions are learned. This idea is applied for the GLVQ in [Harth, 2012]. Thereby, the author learned beside linear combination factors also parameters of the basis functions systems, and therefore the basis function system is not fixed.

In this section we want use the natural dependency in the features, but for an improved learning in the metric adaptation (see Sec. 2.4). Thereby, the first ansatz estimate the relevance profile/matrix as a linear combination of basis functions. The second approach utilize the lateral dependency of the features in the relevance learning scheme.

### 4.1.1 Functional Relevance/Matrix LVQ

As already mentioned, the functional data can be very high dimensional. This leads to a huge number of free parameters to be adjusted in GRLVQ or GMLVQ during the metric adaptation, respectively. In [Mendenhall and Merényi, 2008] it is exemplary shown that for hyper-spectral images unstable behavior may occur in GRLVQ. The authors also identify a possibility to avoid such instabilities for this special case. Through, in this approach the functional properties are not used.

[1] It has to pointed out: the cardinality of a full basis functions system $\mathcal{B}$ is infinity in general. For approximation of the data a finite subset of $\mathcal{B}$ has to be applied.

In the standard GR/MLVQ the single relevance/matrix values are learned independently, such that the functional characteristic of the data is not considered. In the following sections, this functional property will be incorporated in the relevance profile adaption of the GRLVQ to reduce the number of free relevance parameters. Afterwards, this idea is transfered to the GMLVQ.

**Functional Relevance Learning**

In the Generalized Functional Relevance Learning Vector Quantization (GFRLVQ) beside the data points $v(t)$ and prototypes $w(t)$, the relevance profile $\lambda$ is interpreted as a function $\lambda(t)$ with $\lambda_j = \lambda(t_j)$.[2] Thus, the weighted Euclidean distance 2.66 yields to

$$d_E^2\left(v(t), w(t), \lambda\left(t\right)\right) = \int \lambda\left(t\right)\left(v\left(t\right) - w\left(t\right)\right)^2 dt \tag{4.3}$$

with the normalization constraint:

$$\int \lambda(t)\, dt = 1\,. \tag{4.4}$$

Compared to the standard GRLVQ the weighted Euclidean distance (2.66) is chosen with simple $\lambda\left(t\right)$ instead of $\lambda^2\left(t\right)$. The principle is the same. An additional restriction is $\lambda\left(t\right) \geq 0$.

The idea is to approximate the *relevance function* by a superposition of simple positive basis functions $\mathcal{K}_l$ depending on only a small parameter set $\Upsilon_l$:

$$\lambda\left(t\right) = \sum_{l=1}^{K} \beta_l\, \mathcal{K}_l\left(t, \Upsilon_l\right). \tag{4.5}$$

The weighting parameters $\beta_l \geq 0$ are restricted to be convex $\sum_{l=1}^{K} \beta_l = 1$ and describe the influence of a certain $\mathcal{K}_l$. The positive weighting values also guaranty a positive relevance profile $\lambda\left(t\right) \geq 0$ for non-negative $\mathcal{K}_l$.

Famous examples for $\mathcal{K}_l$ are the standard Gaussians:

$$\mathcal{K}_l^G\left(t, \tau_l, \sigma_l\right) = \frac{1}{\sigma_l\sqrt{2\pi}} \exp\left(-\frac{\left(t-\tau_l\right)^2}{2\sigma_l^2}\right) \tag{4.6}$$

or the Lorentzians

$$\mathcal{K}_l^L\left(t, \tau_l, \eta_l\right) = \frac{1}{\eta_l \pi} \frac{\eta_l^2}{\eta_l^2 + \left(t-\tau_l\right)^2}\,, \tag{4.7}$$

[2] It has to be pointed out again, that the data points or prototypes do not has to be given as functions, respectively. This approach also holds for the discrete representations, but the underlaying data might be fulfilled the functional characteristic.

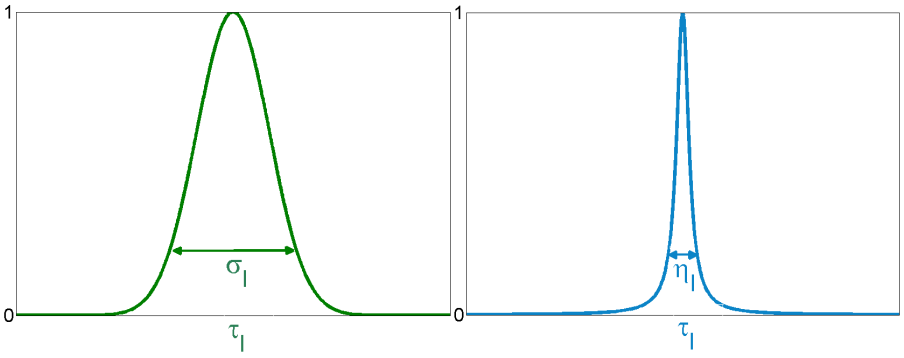


**Figure 4.1**: Shape of the basis function Gaussian (left) and Lorentzian(right)

with the shifting parameter $\tau_l$ and the shaping parameter $\sigma_l \geq 0$ or $\eta_k \geq 0$, respectively (see Fig 4.1).

The Gaussian or Lorentzian basis functions generate an orthogonal basis system of quadratic integrable functions in the Hilbert space $\mathcal{L}_2$. Such orthogonal basis systems guaranty that any arbitrary approximation precision can achieve for an adequate large $K$-value ([Pekalska and Duin, 2005],[Kantorowitsch and Akilow, 1978]). If the basis system is orthonormal additionally, the normalization constraint (4.4) is fulfilled. Yet, any basis system or even a mixture of several basis function can be applied. This might lead to reduced precision and additional requirements concerning the normalization. Obviously, a complete basis system is not realizable or constructive, respectively. The cardinality of $K$ has to be chosen in trade off good approximation and reduction of free parameters. This might be treated as a sparsity problem as explained on page 94ff.

The weighted Euclidean distance (4.3) with the relevance functions (4.5) is

$$d_E^2\left(v(t), w(t), \lambda\left(t\right)\right) = \sum_{l=1}^{K} \beta_l \int \mathcal{K}_l\left(t, \Upsilon_l\right)\left(v\left(t\right) - w\left(t\right)\right)^2 dt\,. \tag{4.8}$$

The adaptation of the relevance profile takes place by the learning of the weighting parameter $\beta_l$ and the parameters of the basis function $\Upsilon_l$, respectively. For this purpose, again the stochastic gradient scheme is applied to the cost function of GFRLVQ:

$$E_{GFRLVQ}(W, \lambda(t)) = \sum_{v(t)\in V} f_\theta\left(\mu_{W,\lambda(t)}\left(v(t)\right)\right) \tag{4.9}$$

$$\text{with} \qquad \mu_{W,\lambda(t)}\left(v(t)\right) = \frac{d^+\left(v(t), \lambda\left(t\right)\right) - d^-\left(v(t), \lambda\left(t\right)\right)}{d^+\left(v(t), \lambda\left(t\right)\right) + d^-\left(v(t), \lambda\left(t\right)\right)} \tag{4.10}$$

and the monotone increasing transfer function $f_\theta$ (2.38) and the distances $d^\pm\left(v(t), \lambda\left(t\right)\right) = d_E^2\left(v(t), w^\pm(t), \lambda\left(t\right)\right)$. The update of an arbitrary parameter $\vartheta_l \in$

$(\Upsilon_l \cup \beta_l)$ results in

$$\vartheta_l \leftarrow \vartheta_l - \alpha_\vartheta \frac{\partial E_{GFRLVQ}}{\partial \vartheta_l} \tag{4.11}$$

with the learning rate $0 < \alpha_{\vartheta_l} << 1$ and the partial derivatives of $E_{GFRLVQ}$ according to

$$\frac{\partial E_{GFRLVQ}}{\partial \vartheta_l} = \xi_\theta^+(v(t)) \cdot \frac{\partial d^+(v(t), \lambda(t))}{\partial \vartheta_l} + \xi_\theta^-(v(t)) \cdot \frac{\partial d^-(v(t), \lambda(t))}{\partial \vartheta_l} \tag{4.12}$$

for a random chosen data point $v(t)$.

More precisely, for the weighting factor $\beta_l$ it is obtained:

$$\frac{\partial d^\pm(v(t), \lambda(t))}{\partial \beta_l} = \int \mathcal{K}_l(t, \Upsilon_l) \left(v(t) - w^\pm(t)\right)^2 dt \,. \tag{4.13}$$

Hereinafter, the derivatives of the exemplary Gaussian $\mathcal{K}_l^G$ and Lorentzian $\mathcal{K}_l^L$ basis functions according to each parameter are listed:

$$\frac{\partial \mathcal{K}_l^G(t, \tau_l, \sigma_l)}{\partial \sigma_l} = \frac{1}{\sigma_l}\left(\frac{(t-\tau_l)^2}{\sigma_l^2} - 1\right) \cdot \mathcal{K}_l^G(t, \tau_l, \sigma_l) \tag{4.14}$$

$$\frac{\partial \mathcal{K}_l^G(t, \tau_l, \sigma_l)}{\partial \tau_l} = \frac{1}{\sigma_l^2}(t-\tau_l) \cdot \mathcal{K}_l^G(t, \tau_l, \sigma_l) \tag{4.15}$$

$$\frac{\partial \mathcal{K}_l^L(t, \tau_l, \eta_l)}{\partial \eta_l} = \frac{1}{\eta_l}\frac{(t-\tau_l)^2 - \eta_l^2}{\eta_l^2 + (t-\tau_l)^2} \cdot \mathcal{K}_l^L(t, \tau_l, \eta_l) \tag{4.16}$$

$$\frac{\partial \mathcal{K}_l^L(t, \tau_l, \eta_l)}{\partial \tau_l} = \frac{1}{\eta_l}\frac{2(t-\tau_l)}{\eta_l^2 + (t-\tau_l)^2} \cdot \mathcal{K}_l^L(t, \tau_l, \eta_l) \,. \tag{4.17}$$

Thereby, it is assumed that integration and differentiation can be interchanged. This operation is allowed if the partial derivative $|\frac{\partial \mathcal{K}_l(t, \Upsilon_l)}{\partial \vartheta_{l,i}}|$ can be majored by an integrable function $\phi$ in the sense of Lebesgue and $\mathcal{K}_l(t, \Upsilon_l)$ is itself Lebesgue-integrable for each $\vartheta_l$ [Kantorowitsch and Akilow, 1978].

The updated of the prototypes are analog to (2.39) applying the functional relevance profile (4.5). Thus, the principle of GRLVQ is not changed.

The shape of the resulted relevance profile depends on the choice of the basis function, i. e. smooth basis functions like Gaussians lead to a smooth relevance profile or rather sharply peaked $\mathcal{K}_l$ like Lorentzian for roughly structured profiles. Beside the mentioned basis function other ones are thinkable like the locality improved kernels (LIK) [Hammer et al., 2005] or the generalized hyperbolic skew Student t-distributions [Aas and Haff, 2006]. A constraint for the basis function is the differentiability with respect to their parameters or even an approximation of them. Obviously, a mixture of

different kind of basis functions is also possible to achieve diversity in the shape.

It has to be considered that each basis function parameter has its own learning rate $\alpha_\vartheta$. These learning rates have to be chosen carefully, inter alia due to the fact that the single parameters influence each other. Frequently, in functional data not only one feature is relevant but a range of features, thus such a range can be covered by single, small basis function with a high $\beta_l$ value or by a broad one. This interaction of the parameters justifying the challenging learning and is already known from density estimation by Parzen-Rosenblatt. [Principe, 2010].

In the Applications 4.1 the GFRLVQ is used to the *Wine* and *Tecator* data set described in Section 1.2.

**Application 4.1** (Functional Metric Adaption in GLVQ)**.**

**data set** Tecator, Wine

**methods** GFRLVQ (compared with GRLVQ)

**parameter settings**

| | |
|---|---|
| number of prototypes | Tecator: $N_W \in \{10, 20, 40\}$ |
| | Wine: $N_W \in \{4, 8, 12\}$ |
| distance function | $d_E^2(\mathbf{v}, \mathbf{w}, \lambda)$, $d_E^2(\mathbf{v}, \mathbf{w}, \lambda(t))$ |
| learning rates | $\alpha_W = 0.01$ |
| GRLVQ | $\alpha_\lambda = 0.01$ |
| GFRLVQ | $\alpha_\beta = 0.003$ |
| | $\alpha_\sigma = \alpha_\eta = 0.01$ |
| | $\alpha_\tau = 0.1$ |
| transfer function parameter | $\theta = 1$ |
| number of basis functions | $K = \{1, 5, 10\}$ |
| kind of basis function | Tecator: Gaussians; Wine: Lorenzians |

**experimental settings**

| | |
|---|---|
| training/test | see data description in Section 1.2 |
| epochs | GRLVQ: 5 000 |
| | GFRLVQ: 150 000 |

**descriptions**
In these experiments we varied the number of prototypes as well as the number of basis functions. Hence, the parameters for relevance learning are drastically reduced to $3/15/30$ from $100$(Tecator) and $256$ (Wine), respectively. Further, we carefully adapt the heights, widths as well as the centers of the basis functions during learning for both experiments. More precisely, at the first $50\,000$ training epochs only the high $\beta$ and location of the centers $\tau$ are adapted. Then, the high $\beta$ was fixed and the width of the basis function $\sigma$ or $\eta$, respectively, was adapted in also $50\,000$ training epochs. At the end, again the high $\beta$ and location of the centers $\tau$ were learned while fixing the width. According to the more smooth shape of the spectra, we applied GFRLVQ using Gaussian basis functions for Tecator. For the Wine data set we used the Lorentzians due to the more peaked spectra.

**results in numbers** accuracy in $\%$ (train/test)

| | Tecator | | | Wine | | |
|---|---|---|---|---|---|---|
| $K \setminus \lvert W\rvert$ | 10 | 20 | 40 | 4 | 8 | 12 |
| 1 | 70.8 | 84.1 | 90.0 | 90.1 | 91.2 | 93.4 |
| | 70.5 | 70.5 | 85.3 | 73.3 | 80.0 | 83.3 |
| 5 | 71.1 | 81.7 | 90.0 | 91.2 | 89.0 | 93.4 |
| | 71.6 | 75.8 | 83.2 | 76.7 | 73.3 | 83.3 |
| 10 | 75.0 | 88.2 | 90.8 | 89.0 | 90.1 | 93.4 |
| | 76.8 | 80.0 | 84.2 | 80.0 | 80.0 | 86.7 |
| GRLVQ | 71.7 | 87.5 | 94.2 | 93.4 | 91.2 | 93.4 |
| | 70.5 | 77.9 | 83.2 | 83.3 | 86.7 | 80.0 |

**results in figures**

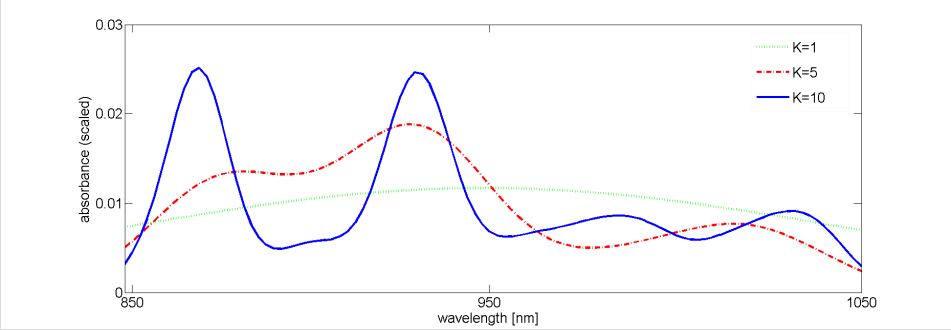


**Figure 4.2**: *Tecator*: Global relevance profiles obtained using different number $K$ of Gaussian basis functions for functional relevance learning using 20 prototypes for learning. The richness in shape increases with $K$ while keeping the relative smoothness.

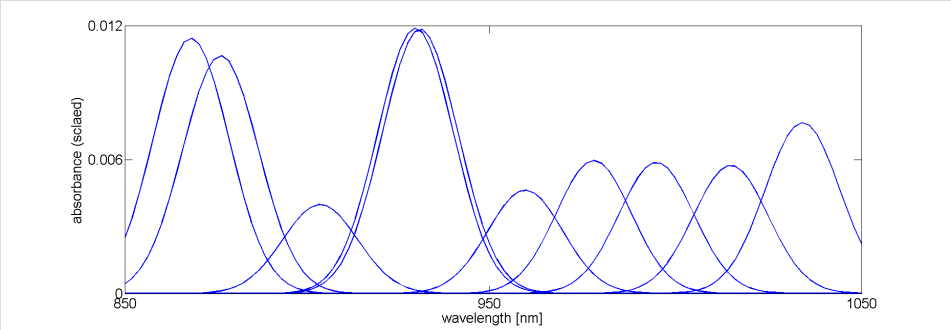


**Figure 4.3**: *Tecator*: Distribution of the $K = 10$ adapted Gaussians for the functional relevance profile for the *Tecator data set* using 20 prototypes for learning with global metric adaptation. The resulting relevance profile is shown in Fig.4.2.

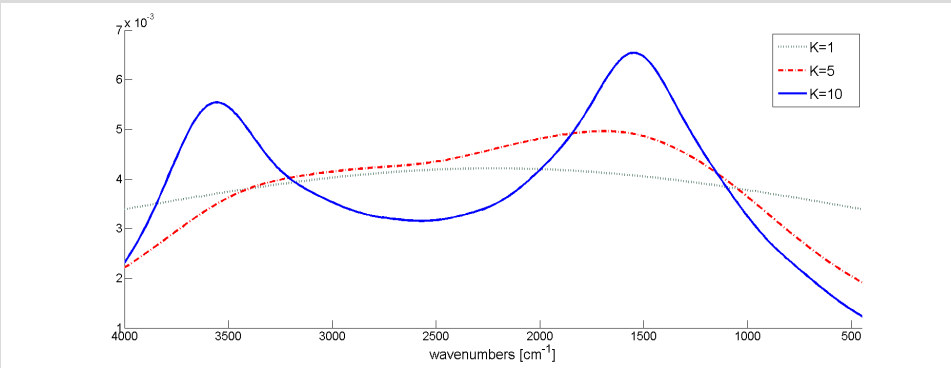

**Figure 4.4**: *Wine*: Global relevance profiles obtained using different number K of Lorentzian basis functions for functional relevance learning using 12 prototypes for learning. As for the *Tecator* data set, the richness in shape increases with K while keeping the relative smoothness.

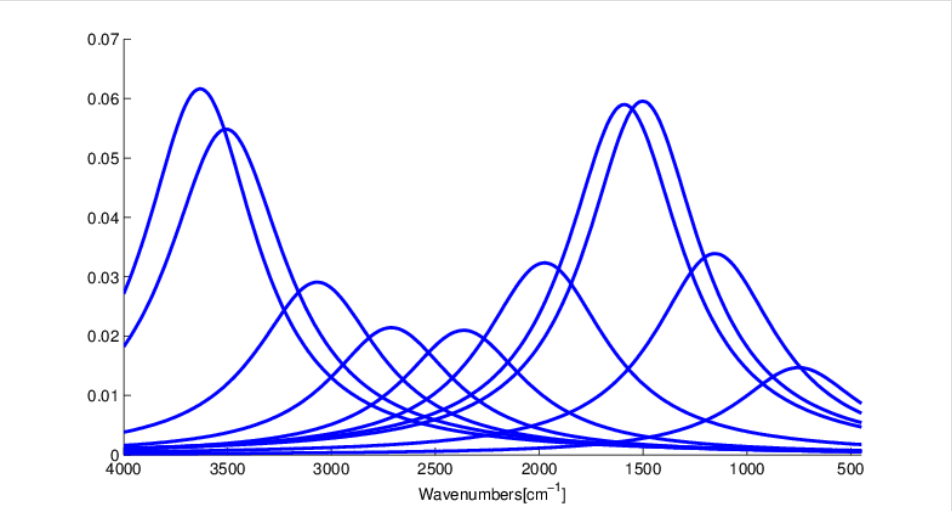


**Figure 4.5**: *Wine*: Distribution of the $K = 10$ adapted Lorentzians for the functional relevance profile using $8$ prototypes.

**resume** As we can see in the table of the accuracies, the classification accuracy of the GFRLVQ are comparable to those of standard GLVQ or other approaches like in [Krier et al., 2009]. However, the results are obtained using a considerably lower number of adapting parameters for relevance learning compared to standard GRLVQ.
As can be expected, the shape of the relevance profile becomes richer with increasing $K$-values (see Fig. 4.2 and Fig. 4.4), while keeping the smooth character according to the smooth applied basis functions. In Fig. 4.3 and Fig. 4.5 the single basis functions are pictured for $K = 10$. The superposition of these basis functions yields to the relevance profile depicted in Fig. 4.2 and Fig. 4.4, respectively.
It has to be mentioned that the number of free parameters can be reduced drastically compared to standard relevance learning, Through, the number of learning epochs is much higher in GFRLVQ. One reason for this is interaction of the basis function parameters and thus, the adaptation of the $\sigma/\eta$ and $\tau$ has to be performed carefully.

### Structural Sparsity

An open question is the suitable choice of the number of basis functions $K$. It has to be find a compromise between minimal free parameters to optimize and a good approximation of the relevance profile. Obviously, if K is set too high, the goal of reducing the number of free parameters rigorously can not be achieved. Otherwise, if the number of basis function is chosen too small, the relevance profile can not determined precisely enough and the accuracy might be decline. This problem of finding a suitable $K$ can be seen as *structural* sparseness requirement.

In a learned relevance profile the number of weighting factors $\beta_l > 0$ is equal of the number of used basis function. Therefore, if we find a possibility to reduce the number of $\beta_l > 0$, we achieve structural sparsity. Possibilities to realize sparsity in one parameter in gradient descent learning methods is already presented on page 67f for features sparsity in GRLVQ. This scheme can simply adopted applying it to the $\beta_l$-values and therefore, suitable for the structural sparsity in $E_{S-GFRLVQ}$.

However, this sparsity constraint does not lead to the effect of feature sparsity, because the basis functions may cover a broad range depending on the width. Thus, only a structural model sparsity is obtained in this way.

### Feature Sparsity

To achieve feature sparsity as in GRLVQ, we have to modify this approach in another manner:

Exemplary, we apply the Shannon entropy. The additional feature sparsity term $E_f(\lambda(t))$ yields to:

$$E_f(\lambda(t)) = -\int \left(\sum_{l=1}^{K} \beta_l \, \mathcal{K}_l(t, \Upsilon_l)\right) \log\left(\sum_{l=1}^{K} \beta_l \, \mathcal{K}_l(t, \Upsilon_l)\right) dt \tag{4.18}$$

in

$$E_{sGFRLVQ} = E_{GFRLVQ}(W, \lambda(t)) + \gamma(\iota) E_f(\lambda(t)) \, . \tag{4.19}$$

The general derivatives ends up in:

$$\frac{\partial E_f(\lambda)}{\partial \vartheta_l} = -\int \frac{\partial \lambda(t)}{\partial \vartheta_l} \left[1 + \log(\lambda(t))\right] dt \, . \tag{4.20}$$

The structural and feature sparsity is applied in the GFRLVQ on the *Tecator* data set (see Appl. 4.2).

**Generalized Functional Matrix LVQ (GFMLVQ)**

A generalization of the GRLVQ is the GMLVQ (see 2.4). Moreover, the idea of functional relevance learning can be extended to the GMLVQ. Thereby, it is assumed that the mapping matrix $\Omega \in \mathbb{R}^{m \times D}$ in (2.68) is a discrete representation of a continuous function $\Omega(t_1, t_2)$ [Riedel and Nebel, 2012].

Like in (4.8), this continuous function is approximated by a superposition of two-dimensional basis functions $\mathcal{K}_l(t_1, t_2, \Upsilon_{1,l}, \Upsilon_{2,l})$:

$$\Omega(t_1, t_2) = \sum_{l=1}^{K} \beta_l \, \mathcal{K}_l(t_1, t_2, \Upsilon_{1,l}, \Upsilon_{2,l}) \tag{4.21}$$

Let us again consider Gaussians and Lorentzians for exemplary basis functions. The two-dimensional symmetric Gaussian is

$$K_l(t_1, t_2, \tau_{1,l}, \tau_{2,l}, \sigma_{1,l}, \sigma_{2,l}) = \frac{1}{\sigma_{1,l} \cdot \sigma_{2,l} \cdot 2\pi} \exp\left(-\left(\frac{(t_1 - \tau_{1,l})^2}{2\sigma_{1,l}^2} + \frac{(t_2 - \tau_{2,l})^2}{2\sigma_{2,l}^2}\right)\right)$$

whereas the symmetric Lorentzian is gotten by

$$K_l(t_1, t_2, \tau_{1,l}, \tau_{2,l}, \eta_{1,l}, \eta_{2,l}) = \frac{1}{\eta_{1,l} \cdot \eta_{2,l} \cdot \pi^2} \left(\frac{\eta_{1,l}^2}{\eta_{1,l}^2 + (t_1 - \tau_{1,l})^2} \cdot \frac{\eta_{2,l}^2}{\eta_{2,l}^2 + (t_2 - \tau_{2,l})^2}\right).$$

The number of basis $K$ should be again smaller than the number of features $D$, more precisely $K \ll m \times D$.

It is evident, that the handling of (4.21) requires high computational effort and the learning of the single parameters is more challenging compared to the one-dimensional basis function in GFRLVQ. The authors of [Riedel and Nebel, 2012] show, that a factorized version of (4.21) leads to an easier handling. For that manner, the property of the slight self-regularization and row-wise structure known from the converged squared matrix $\Omega$ in GMLVQ (see page 53 or [Schneider et al., 2010]) is utilized: Instead of two-dimensional basis functions for the whole $\Omega(t_1, t_2)$, only the rows $\Omega_j(t_2)$ with $j = 1, ..., m$ of $\Omega(t_1, t_2)$ are generated by one-dimensional basis functions:

$$\Omega_j(t_2) = \sum_{l=1}^{K} \beta_{j,l} \, \mathcal{K}_{j,l}(t_2) \tag{4.22}$$

The one-dimensional basis functions $\mathcal{K}_{j,l}(t_2)$ are easier to handle numerically. However, the number of weighting parameter $\beta_{j,l}$ is increased, from $K$ to $K \times m$. Obviously, examples for $\mathcal{K}_{j,l}(t_2)$ are the one-dimensional Gaussian or Lorentzian functions known from the GFRLVQ. Experiments show, that there is a vanishing quality loss of

the relevance profile applying this factorized version compared to the full scheme (4.21) [Riedel and Nebel, 2012].

Further, the approaches of structured and feature sparsity for the functional GRLVQ can be easily transfered to the GFMLVQ. The cost function of GFMLVQ for structured sparsity is extended to:

$$E_{kGFMLVQ} = E_{GFMLVQ}(W, \Omega(t_1, t_2)) + \gamma(\iota)\ E_s(\beta)$$

with the weighting function $\gamma(\iota)$ and a sparsity requirement $E_s(\beta)$. Thereby, $E_s(\beta)$ can be again a entropy function or based on a $l_p$-norm (see page 67): The same is valid for the feature sparsity

$$E_{sGFMLVQ} = E_{GFMLVQ}(W, \Omega(t_1, t_2)) + \gamma(\iota)\ E_s(\Omega(t_1, t_2))\ .$$

**Application 4.2** (GFRLVQ with feature and structural sparsity)**.**

**data set** Tecator

**methods** sGRLVQ with LASSO, sGFRLVQ and kGFRLVQ

**parameter settings**

| | |
|---|---|
| number of prototypes | 20 |
| distance function | $d^2_E(\mathbf{v}, \mathbf{w}, \lambda(t))$ |
| learning rates | $\alpha_W = 0.01$ |
| | $\alpha_\beta = 0.003$ |
| | $\alpha_\sigma = 0.01$ |
| | $\alpha_\tau = 0.1$ |
| transfer function parameter | $\theta = 1$ |
| number of basis functions | $K = 10$ $(\iota = 0)$ |
| kind of basis functions | Gaussians |

**experimental settings**

| | |
|---|---|
| training/test | split (see data description in Section 1.2) |
| initialization | learned GRLVQ/GFRLVQ model |
| epochs | sGRLVQ: 5000, sGFRLVQ: 4500, kGFRLVQ: 1200 |

**descriptions** In the beginning, the GFRLVQ model were trained applying $K = 10$ Gaussians. After this basic training the influence of the sparsity constraint was slowly increased by linear growth of the weighting factor $\gamma_S(\tau)$ for structural and $\gamma_F(\tau)$ for feature sparsity, respectively. Thereby, the widths of the single basis function give information about the feature sparsity. Small Gaussians imply a small number of non-vanishing features.

**results in numbers**

| kind of sparsity | non | GFRLVQ structural | feature | GRLVQ Tib-LASSO |
|---|---|---|---|---|
| $K$ | 10 | 3 | 10 | - |
| $D$ | 100 | 100 | 75 | 74 |
| accuracy | 88.2% | 88.5% | 90.0% | 91.7% |

**results in figures**

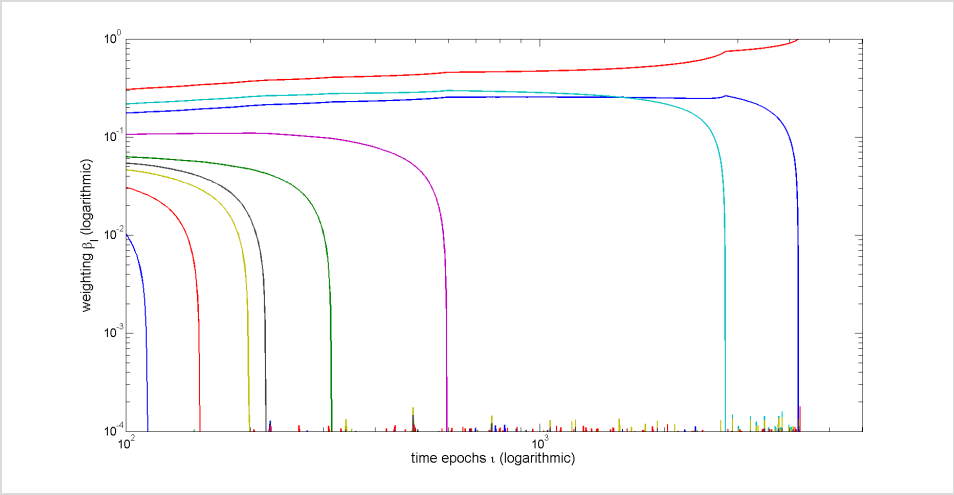


**Figure 4.6**: *Structural sparsity*: Development in time $\iota$ of weighting coefficients $\beta_l$. The weights $\beta_l$ vanish with growing sparsity pressure except only a single remaining for maximum sparseness.

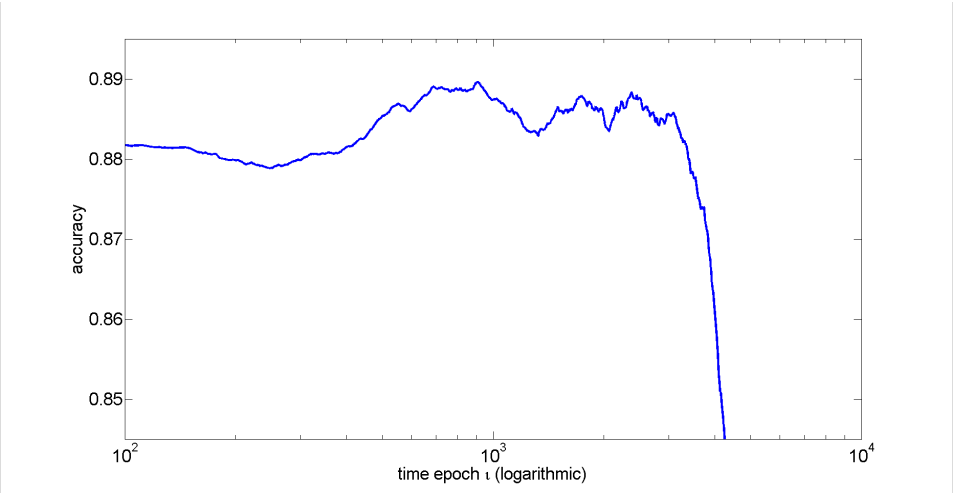


**Figure 4.7**: *Structural sparsity*: Development in time $\iota$ of the accuracy. The accuracy is still high if at least 3 basis functions are weighted to be active. After one more function becomes inactive, the accuracy decreases significantly indicating an substantial information loss.

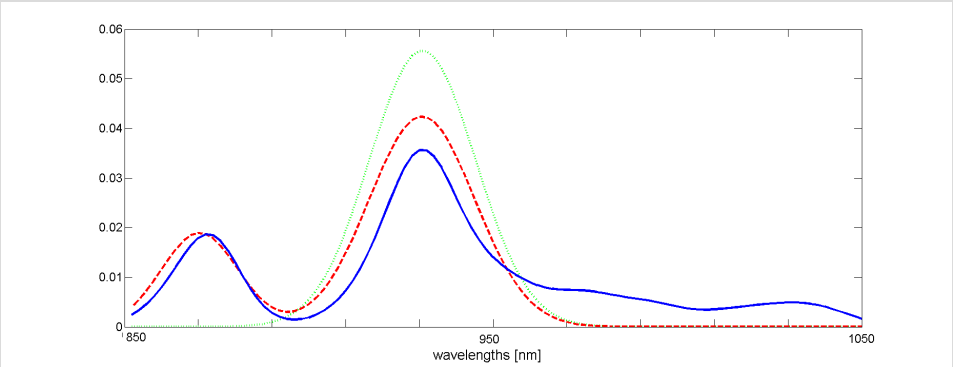


**Figure 4.8**: *Structural sparsity*: Relevance profiles at the beginning of the structural sparsity optimization (blue -solid), just before the critical transition (red - dashed) and in the final phase (green - dotted).

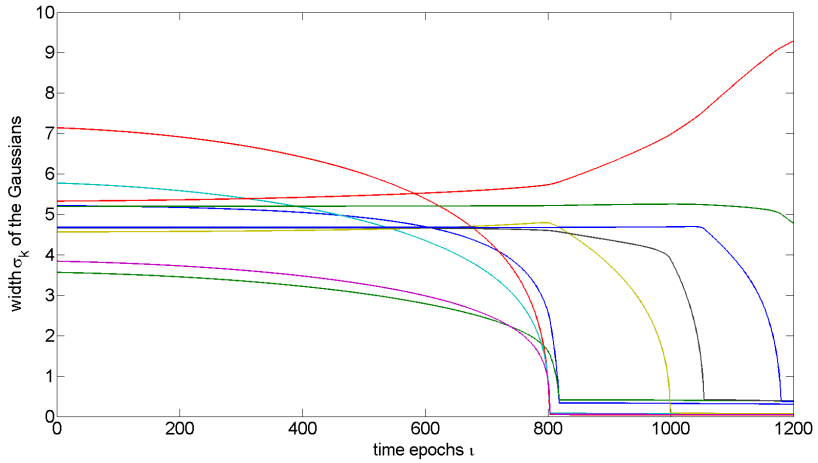


**Figure 4.9**: *Feature sparsity*: Development in time $\iota$ of the width $\sigma_l$ of the Gaussians. The coefficients vanish with growing sparsity pressure except only a single remaining for maximum sparseness.

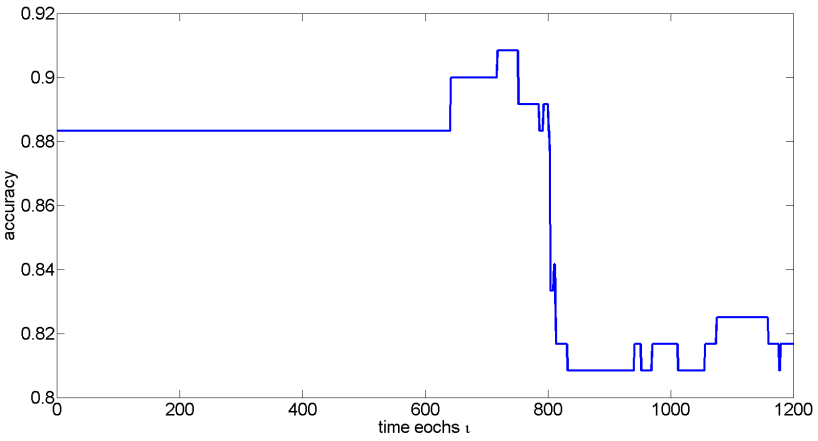


**Figure 4.10**: *Feature sparsity*: Development in time $\iota$ of the accuracy. The accuracy is still high for at least $800$ time cycles. After these, the accuracy decreases significantly indicating an substantial information loss while $5$ basis functions vanish at this time, see Figure 4.9.

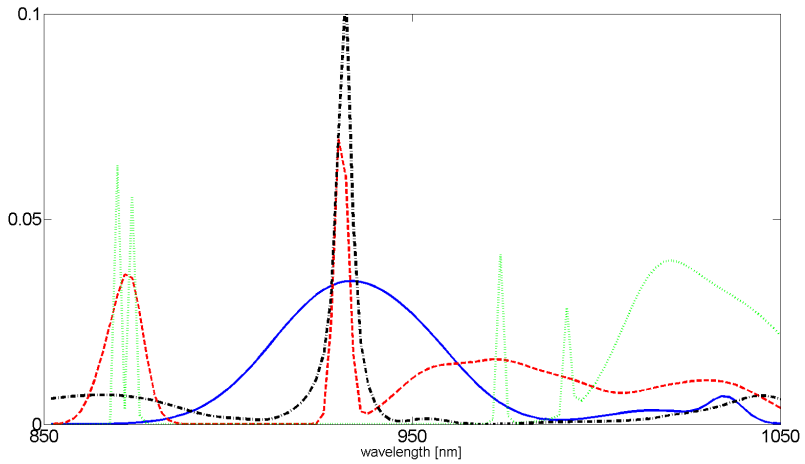


**Figure 4.11**: *Feature sparsity*: Relevance profiles for the sGRLVQ with LASSO (black - dashed) and for GFRLVQ at the beginning of the feature sparsity optimization (blue - solid), just before the critical phase transition (red - dashed) and in the final phase (green - dotted). Thereby, the number of non-vanishing features are: $95$ (blue), $50$ (red) and $40$ (green).

**resume** In case of structural sparsity judged by the entropy $H_S$ of the weighting coefficients $\beta_l$, this leads to a subsequent fading out of the several basis functions (see Fig. 4.6). In the beginning the accuracy level is kept. Above a critical sparsity, a drastic decrease of accuracy can be observed indicating the transition from sparseness optimum to a very low level (around time epoch 2500 and three non-vanishing $\beta_l$, see Fig.4.7). The related relevance profiles at the beginning of the structural sparsity optimization, just before the critical transition and in the final phase are depicted in Fig.4.8. As one can see, the loss of the relevance peak around wavelength $870nm$ leads to the breakdown.
In the second experiment the feature sparsity was investigated. Again, we observe a drastic loss of accuracy after approximately $800$ time epochs corresponding to vanishing $\sigma_l$-values for $5$ basis functions (see Fig. 4.10 and Fig. 4.11). The remaining model is too poor to keep the information. The number of non-vanishing features are $75$ at $\iota = 800$. This is comparable to the feature sparsity approach of the standard GRLVQ with LASSO.

### 4.1.2 Enhancement Generalized Relevance/Matrix LVQ

The functional relevance or matrix learning described in the section before shows good performance and together with the structural sparsity the number of parameters to learn can be drastically reduced for high dimensional data sets. However, the adaption

of the single basis function parameters is hard to handle for the GFRLVQ and even more in the matrix learning. Therefore, the functional G(R/M)LVQ is to laborious for practical applications.

In spite of this fact, the high number of independent parameters to adapt for functional data is still a challenge. The *lateral enhancement* G(R/M)LVQ (eG(R/M)LVQ) is an approach which does not reduce the number of parameters like the GF(R/M)LVQ, but still utilizes the dependency of features in functional data sets and, therefor, regularizes the learning of the relevance profile or mapping matrix, respectively. The basic idea for the eGRLVQ is: for adapting a relevance value $\lambda_i$ the lateral surrounding neighborhood of the dimension $i$ is also taken into account. The consideration of the lateral neighborhood is modeled by the Gaussian known from the neighborhood cooperativeness of the prototypes in SOM 2.1.2 or NG 2.1.3.

More precisely, for the relevance learning the Gaussian enhancement function is set to:

$$h_\sigma(i,j) = \exp\left(\frac{-(i-j)^2}{2\sigma^2}\right) \tag{4.23}$$

which should specify the influence of a dimension $j$ for the update of $\lambda_i$ depending on the width $\sigma$. If we assume the squared scaled Euclidean distance in the GRLVQ for data dissimilarity, the partial derivation in the update of the relevance value $\lambda_i$ should be of the form:

$$\frac{\partial d_\lambda(\mathbf{v},\mathbf{w},h_\sigma)}{\partial \lambda_i} = \sum_{j=1}^{D} h_\sigma(i,j)\ [\mathbf{v}-\mathbf{w}]_j^2 \tag{4.24}$$

This idea is illustrated in Fig. 4.12. Thus, the distance function is obtained by integration of (4.24):

$$d_\lambda(\mathbf{v},\mathbf{w},h_\sigma) = \sum_{k=1}^{D} \lambda_i \varphi_k(\mathbf{v},\mathbf{w},h_\sigma) \tag{4.25}$$

with the local distortion

$$\varphi_k(\mathbf{v},\mathbf{w},h_\sigma) = \sum_{l=1}^{D} h_\sigma(k,l)[\mathbf{v}-\mathbf{w}]_l^2 \tag{4.26}$$

The width $\sigma$ describes the influence range of the neighborhood cooperativeness. For $\sigma \searrow 0$ we obtain the standard GRLVQ[3]. The principle of the shifting of the prototypes is kept, only the distance measure is replaced by $d_\lambda(\mathbf{v},\mathbf{w},h_\sigma)$.

The idea of lateral neighborhood cooperativeness can be transfered to the matrix learning, too. Therefore, we let $\Omega \in \mathbb{R}^{m\times D}$ the mapping matrix (see 2.4) and $H^\sigma \in$

[3]Like in the GFRLVQ the $\lambda_i^2$ is replaced by $\lambda_i$ with the additional requirement of $\lambda_i \geq 0$.

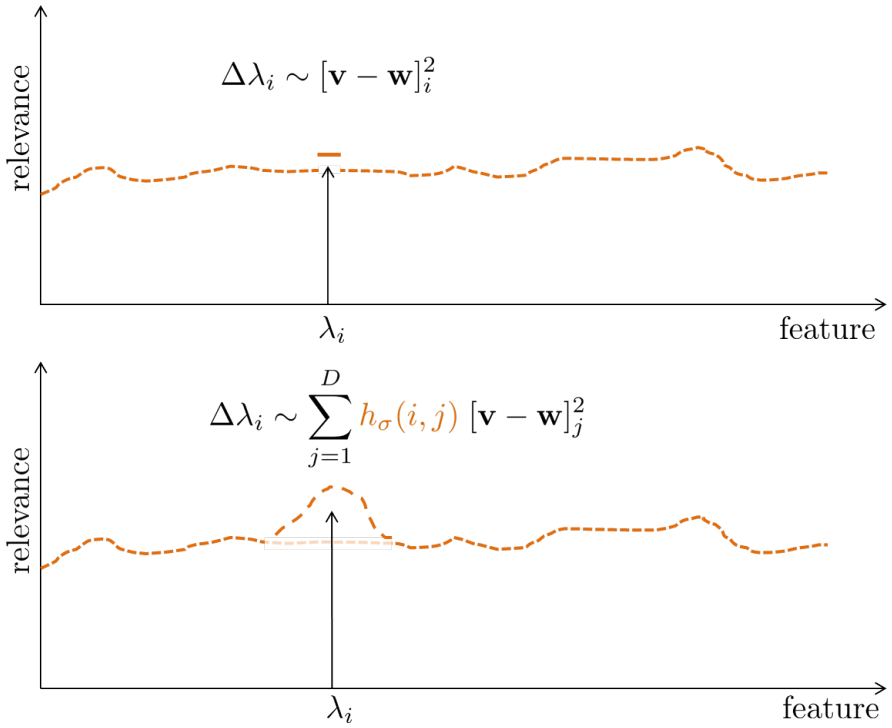


**Figure 4.12**: Illustration of the idea of lateral enhancement learning for a relevance weight $\lambda_i$ including the neighborhood cooperativeness (top: standard relevance learning, bottom: lateral enhancement according to (4.24)

$\mathbb{R}^{D\times D}$ be a lateral *enhancement matrix*, with the entries:

$$H^\sigma_{i,j} = h_\sigma(i,j) \tag{4.27}$$

are based on the enhancement function (4.23) for $i,j = 1,\ldots,D$.

Hence, the respective distance becomes:

$$d_\Omega\left(\mathbf{v},\mathbf{w},H^\sigma\right) = \sum_{i=1}^{m}\left(\sum_{k=1}^{D}\Omega_{i,k}\;\varphi_k(\mathbf{v},\mathbf{w},H^\sigma)\right)^2 \tag{4.28}$$

analog to (4.25) with the local distortion:

$$\varphi_k(\mathbf{v},\mathbf{w},H^\sigma) = \sum_{s=1}^{D} H^\sigma_{k,s}\,[\mathbf{v}-\mathbf{w}]_s \tag{4.29}$$

Moreover, the distance $d_\Omega(\mathbf{v}, \mathbf{w}, H^\sigma)$ allows a shortened notation:

$$d_\Omega(\mathbf{v}, \mathbf{w}, H^\sigma) = (\Omega H(\mathbf{v} - \mathbf{w}))^2 \tag{4.30}$$

$$= \left(\hat{\Omega}(\mathbf{v} - \mathbf{w})\right)^2 \tag{4.31}$$

which can be easily obtained by transferring the matrices multiplication into sums. Consequently, the introducing of the enhancement matrix $H$ ends up in an additional linear mapping or an modified mapping $\hat{\Omega} = \Omega H$, respectively. However, this additional linear mapping including prior knowledge about the data, e. g. the lateral neighborhood cooperativeness of the dimensions in functional data.

The update of the mapping matrix $\Omega$ in eGMLVQ is likewise the update in (2.73) with the partial derivations:

$$\frac{\partial d_\Omega(\mathbf{v}, \mathbf{w}, H^\sigma)}{\partial \Omega_{i,j}} = 2\left[H^\sigma(\mathbf{v} - \mathbf{w})\right]_j \left[\hat{\Omega}(\mathbf{v} - \mathbf{w})\right]_i . \tag{4.32}$$

The enhancement matrix $H^\sigma$ also influences the classification correlation matrix $\Lambda$ (2.76), i. e. $\hat{\Lambda} = (\Omega H^\sigma)^T(\Omega H^\sigma)$. If the Gaussian function is used to generate $H^\sigma$ with $\sigma > 0$, $\hat{\Lambda}$ is smoother, which is expected for functional data. Again, for $\sigma = 0$, $H^\sigma$ is the identity matrix and the standard GMLVQ holds.

Like in GF(R/M)LVQ, other choices of enhancement functions $h(i, j)$ or matrix $H$ are thinkable, depending on the knowledge about the data. Examples are the above mentioned Lorentzians (see 4.1) or the skew Student's t-distribution.

Moreover, the enhancement function or matrix could include further prior correlation information about the features given by an expert. We give a illustrative example in spectral classification: In general, the dimensions in spectra are wavelengths $\omega(i)$. In some data sets these wavelengths are not equidistant or in the data set some wavelengths are cut due to chemical characteristics like water bands (see the Indian Pine data set in section 1.2). In these cases, the enhancement matrix can be defined by using the distance between the neighbored dimensions, i. e.

$$\tilde{h}_\sigma(i, j) = \exp\left(\frac{-(\omega(i) - \omega(j))^2}{2\sigma^2}\right) \tag{4.33}$$

Thus this prior knowledge about the data dimension are taken into account. An application is given in Application 4.4.

Further, the eG(R/M)LVQ can be applied to non-functional data sets, too, by integration of knowledge about given correlations between the dimensions/features. An example thereof can be found in [T. Villmann and Riedel, 2012].

It is well known that the neighborhood cooperativeness in the local costs of NG or SOM stabilizes and accelerate the convergence process, beside the fact that the sensi-

tivity of the initialization is reduced. These advantages hold also for the eG(R/M)LVQ (see Appl. 4.4). Moreover, the principle of cooling the neighborhood range during the learning is suitable for eG(R/M)LVQ, too.

Additionally to the stabilization and acceleration, a better generalization might be also achieved. Indeed the number of parameters to optimized are not directly reduced like in the functional G(R/M)LVQ, but an inherent regularization is achieved due to the impact of lateral neighbored features (see Appl. 4.4).

**Application 4.3** (eGRLVQ)**.**

**data set** Wine

**methods** GRLVQ, eGRLVQ

**parameter settings**

| | |
|---|---|
| number of prototypes | $N_W = 2$ (one per class) |
| distance function | $d_\lambda(\mathbf{v}, \mathbf{w})$ (2.66) |
| | $d_\lambda(\mathbf{v}, \mathbf{w}, h_\sigma)$ (4.25) |
| learning rates | $\alpha_W = 0.01$ |
| | $\alpha_\lambda = 0.1 \cdot \alpha_W$ |
| transfer function parameter | $\theta = 1$ |
| enhancement neighborhood range | $\sigma(\iota_{start}) = 10$ |
| | $\sigma(\iota_{end}) = 0.5$ (linear decay) |

**experimental settings**

| | |
|---|---|
| training/test | 4 fold cross validation |
| number of initializations per fold | 25 |
| epochs | 2000 |

**results in numbers**

| **accuracy** | GRLVQ | eGRLVQ |
|---|---|---|
| training | $87,2\%$ | $87,0\%$ |
| test | $85,4\%$ | $87,0\%$ |

**results in figures**

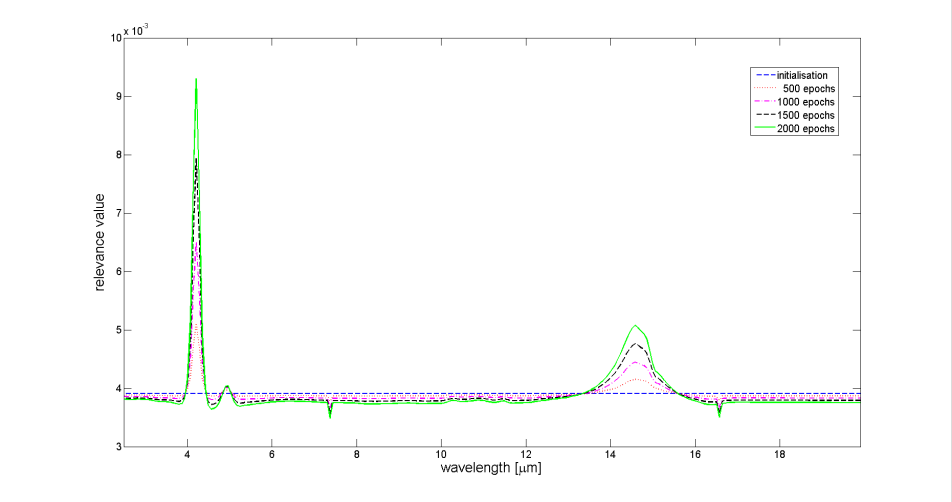


**Figure 4.13**: Relevance profiles of GRLVQ for several training epochs from a typical run during the cross-validation procedure, i. e. the other relevance profiles have an averaged relative error of about $2.05\%$ for each band.

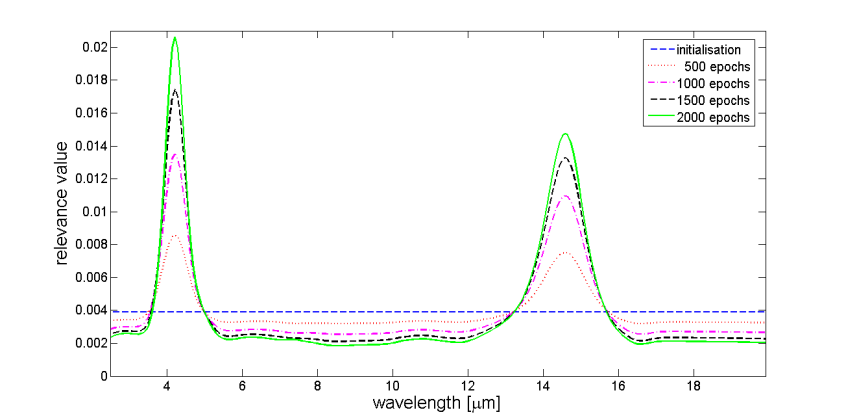


**Figure 4.14**: Relevance profiles of eGRLVQ several training epochs obtained from a typical run during the cross-validation procedure (averaged relative error of $0.41\%$ for each band). Note, in comparison to Fig. 4.13, the scale is doubled.

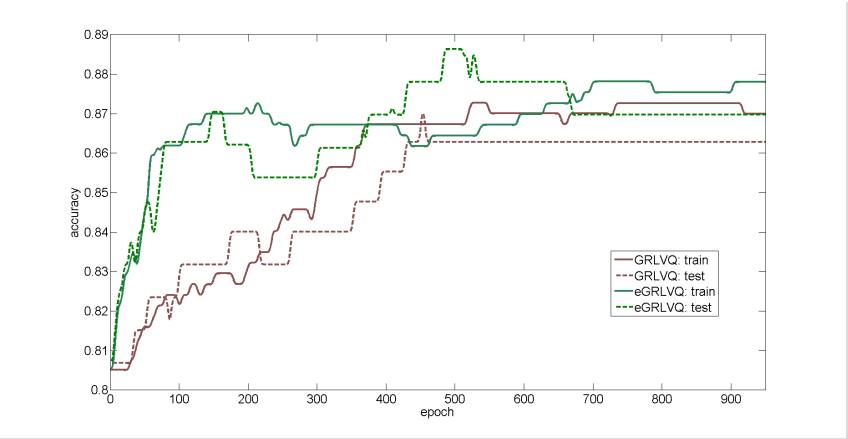


**Figure 4.15**: Averaged training and test accuracies for GRLVQ and eGRLVQ for the first 1000 epochs for 4-fold CV with one initialization. The eGRLVQ uses the enhanced relevance learning achieving a good performance faster than standard GRLVQ.

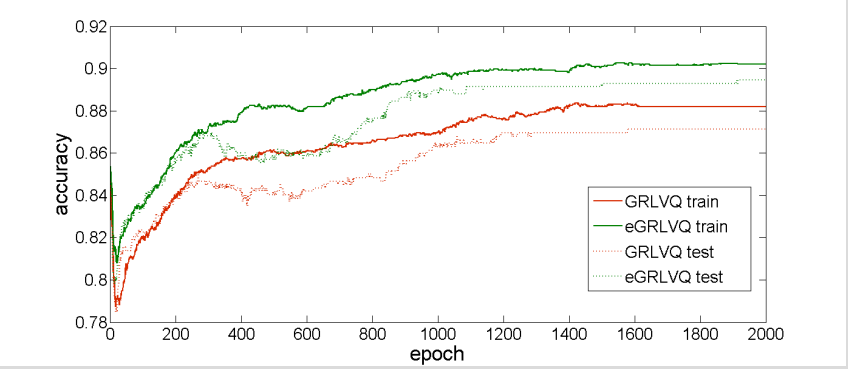


**Figure 4.16**: Averaged training and test accuracies for 25 times 4-fold CV with random initialization

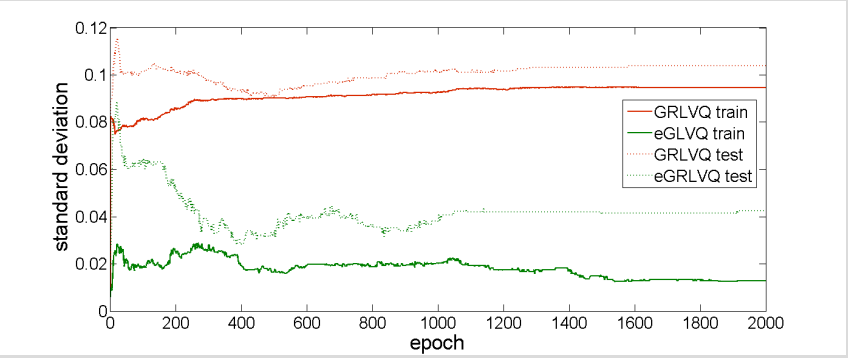


**Figure 4.17**: Standard deviations of the averaged performance of the accuracy curves (see Fig. 4.16). The eGRLVQ shows improved stability in comparison to GRLVQ indicated by lower deviations.

**resume** The wine data set, which contains absorbing spectra, is a functional data set. The spectra are relatively smooth and thus, this property is expected for the relevance profile, too. The final relevance profile (i. e. after 2000 epochs) of the eGRLVQ (see Fig. 4.14 is smoother compared to $\lambda$ learned by the GRLVQ (see Fig. 4.14). Moreover, the relevance peaks for eGRLVQ are better calibrated although both algorithms emphasize the relevances ranging around $4$ $\mu$m and $14 - 15$ $\mu$m. These spectral ranges are also marked as important for class description as it was obtained in [Krier et al., 2008]. The approach in [Krier et al., 2008] is a feature selection scheme based on information theoretic concepts and the authors pointed out, that the first peak in the relevance profiles corresponds to the absorption range of the O–H bond present in alcohol and, hence, is a clear indicator for the alcoholic level.
Further, we observe that the enhanced relevance learning leads to a slightly improved convergence behavior as expected by neighborhood cooperativeness in learning, see Fig.4.15. High performance is achieved after a few epochs for the enhanced model whereas the original relevance learning needs considerably more epochs. Both approaches achieve the final accuracies after approximately $550$ epochs. After this time only the relevance profiles are further improved (see Figures 4.13 and 4.14).
For each of the four cross folds we started with $25$ random initializations, whereas all other parameters were kept for both GRLVQ and eGRLVQ from the previous runs. The achieved accuracy curves, averaged over all cross folds and random initializations, are plotted in Fig. 4.16. The respective curves of standard deviations are depicted in Fig. 4.17. These results give at least a hint that the eGRLVQ is robust and converges faster than standard GRLVQ without enhancement learning.
Investigating these simulation results we can conclude for eGRLVQ a faster convergence with improved stability compared to GRLVQ. Thus, the enhancement learning leads to a more stable behavior with reduced sensitivity according to initialization as already proposed on page 104.

**Application 4.4** (eGMLVQ).

**data set** Indian Pine

**methods** GMLVQ, eGMLVQ

**parameter settings**

| | |
|---|---|
| number of prototypes | 108 |
| distance function | $d_\Omega(\mathbf{v}, \mathbf{w})$ (2.68) |
| | $d_\Omega\left(\mathbf{v}, \mathbf{w}, \tilde{H}^\sigma\right)$ (4.30) |
| dimension of $\Omega$ | $m = 11$ ($\Omega \in \mathbb{R}^{11\times 200}$) |
| learning rates | $\alpha_W = 0.01$ |
| | $\alpha_\Omega = 0.1 \cdot \alpha_W$ |
| transfer function parameter | $\theta = 1$ |
| basis function | GFMLVQ: $K = 10$ Gaussians |
| enhancement neighborhood range | $\sigma(\iota_{start}) = 10$ |
| | $\sigma(\iota_{end}) = 1$ |

**experimental settings**

| | |
|---|---|
| training/test | $25\%$ of data for training, $75\%$ for test |
| number of initializations per fold | 25 |

**descriptions**
The number of prototypes per class are chosen according to the class distribution of the data set and sum up to 108. As described in 1.2, around $1.33\mu\mathrm{m}$ and $1.75\mu\mathrm{m}$ we removed $20$ wavelengths, because they mainly affected by water content. Thus around these wavelengths, the neighborhood cooperativeness of the dimensions is violated. Therefore, this should be reflected in the enhancement matrix $H$, too. The according matrix at the beginning is depicted in 4.18. The value of $\sigma(\iota_{start}) = 10$ at the beginning of the training realizes a large lateral inhibitions for fast learning while the flexibility of the relevance profile is reduced. The flexibility increases while the linear decreasing of the lateral interaction range to the remaining final cooperativeness according to $\sigma(\iota_{end}) = 1$.

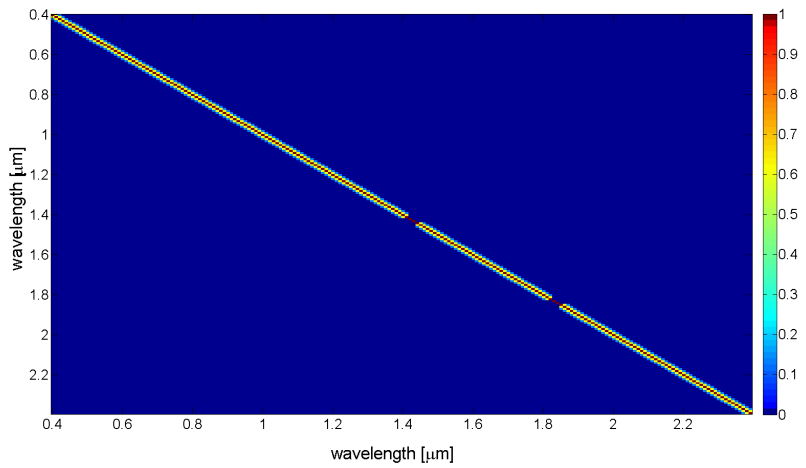


**Figure 4.18**: Enhancement matrix $\tilde{H}$ with the removed neighborhood cooperativeness around affected water bands.

**results in numbers**

| **accuracy** | GMLVQ | GFMLVQ | eGMLVQ |
|---|---|---|---|
| training | 90.0% | 67.7% | 87.4% |
| test | 81.6% | 74.0% | 81.4% |

**results in figures**

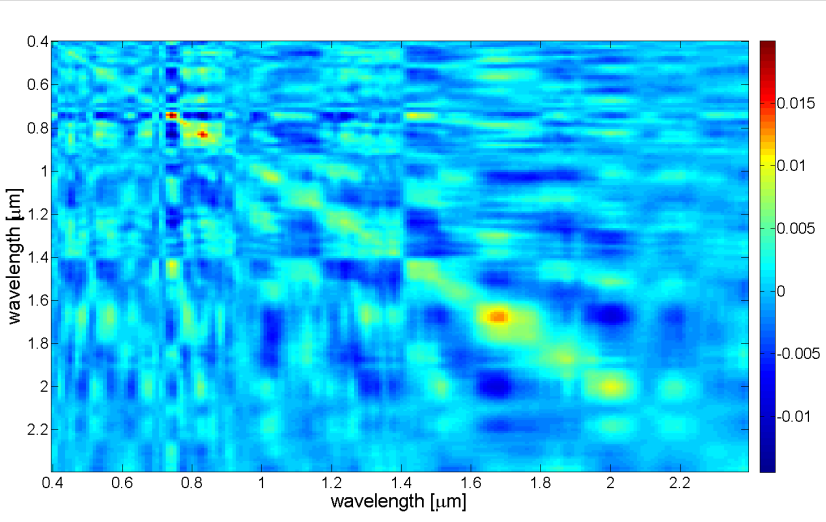


**Figure 4.19**: Classification correlation matrix $\Lambda_1$ learned by the standard GMLVQ

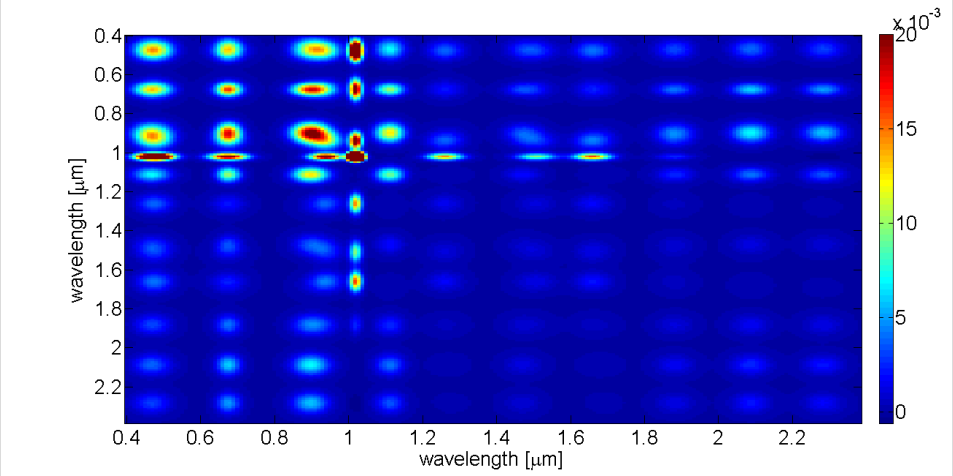


**Figure 4.20**: Classification correlation matrix $\Lambda_2$ learned by the functional GMLVQ with $K = 20$ Gaussians.

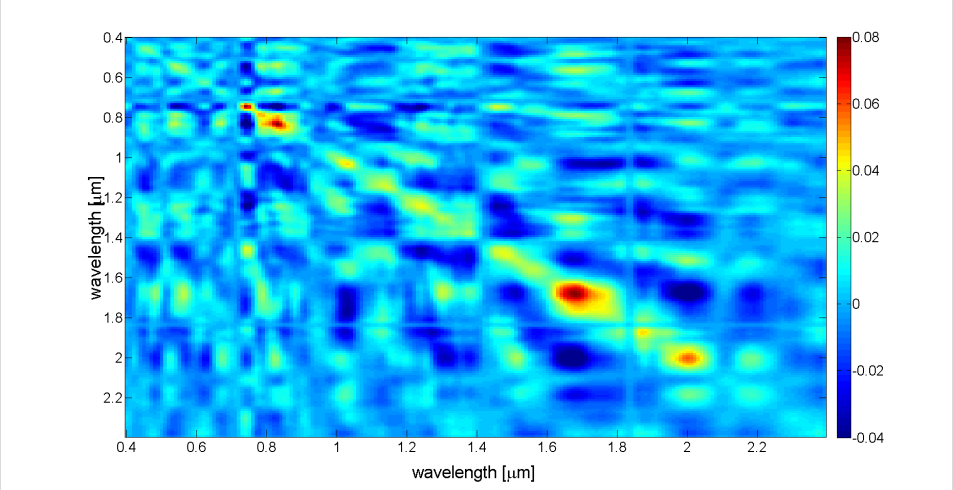


**Figure 4.21**: Classification correlation matrix $\Lambda_3$ learned with eGMLVQ and enhancement matrix $H$ (see Fig. 4.18)

**resume** The reduced accuracy of the GFMLVQ compared to standard GMLVQ shows the difficulty to learn an appropriate matrix representation by the Gaussians. This effect can be observed also in the visualization of the matrix $\Lambda_2$. The classification correlation matrix $\Lambda_2$ of the functional GMLVQ only show significant correlations lower wavelengths (see Fig. 4.20). Yet, this is in contrast to $\Lambda_1$ learned by the standard GMLVQ. In Figure 4.19 can be observed that also correlations between higher wavelengths are taken into account for a better classification. However, $\Lambda_1$ is not smooth as we would expect from the smooth shape of the data, compare Figure 1.9.

Applying the eGMLVQ with the data set adjusted enhancement matrix $\tilde{H}$ the performance is drastically improved compared to GFMLVQ. A test accuracy of $81.1\%$ is achieved similar in comparison to GMLVQ. The matrix $\Lambda_3$ is smoother compared to $\Lambda_1$, which demonstrates the regularizing ability of the enhancement learning. This property can also concluded from the decreased discrepancy between training and test error compared to GMLVQ and GFMLVQ.

**Parts of the section are based on**

*M. Kästner, M. Lange, and T. Villmann:* Fuzzy Supervised Self-Organizing Map for Semi-Supervised Vector Quantization. *Proc. of ICAISC, 2012.*

*M. Kästner and T. Villmann:* Fuzzy Supervised Neural Gas with Sparsity Constraint. *Proc. of (NC)$^2$, 2011.*

## 4.2 Fuzzy Information About the Labels

Until now, we had a closer look to the characteristics of the data, e. g. we use the functional property in the relevance learning. In this section, we concentrate on the information about the classes. On pages 37ff the semi-supervised task and some methods are described closer. A short review: The training data set is split into two subsets: the unlabeled and the labeled data points. The motivation in semi-supervised Vector Quantization is to learn a classifier which use both subsets of the data set for the training. Examples are the *Learning Associations by Self-Organizing* (LASSO [Midenet and Grumbach, 1994]) or the Fuzzy Labeled SOM/NG (FLSOM [Schleif et al., 2007a], FLNG [Villmann et al., 2006a]). The FLSOM is a generalization of the LASSO. However, the theoretical justification is tricky and raise practical difficulties [Schleif, 2014]. Thus, we develope a novel semi-supervised approach.

In the following two sections the Fuzzy Semi-Supervised SOM (FSSOM) and Fuzzy Semi-Supervised NG (FSNG) are deduced, respectively. Thereby, the distance measure in the data space of the standard `Heskes`-SOM or NG is substituted by a distance measure including the label information. The integration of the label information takes place in a multiplicative manner by preserving the structural framework of `Heskes`-SOM or NG, respectively. Thus, the proposition of convergence and stability can be transfered two the new models. The areas of application of these new approaches are manifold. On the one side it is a classical semi-supervised method, i. e. labeled as well as unlabeled data points are considered, on the other side the methods are also able to handle fuzzy information about the labels like it is known from FLSOM/FLNG. Further, extensions like metric adaptation or sparsity requirements can be applied.

### 4.2.1 Fuzzy Semi-Supervised Self-Organizing Maps

At first, in the following the class assignment for the data point $\mathbf{v}$ and prototypes $\mathbf{wr}$ are modeled by an assignment vector $\mathbf{c}(\mathbf{v}) = (c_1(\mathbf{v}), \ldots, c_{N_C}(\mathbf{v})) \in [0,1]^{N_C}$ or $\mathbf{y}(W) \in [0,1]^{N_C \times N_W}$, whereby $N_C$ is the number of classes and $N_W$ the number of prototypes, respectively. We distinguish between possibilistic, probabilistic ($\sum_{l=1}^{N_C} c_l(\mathbf{v}) = 1$) and

crisp labeling ($\sum_{l=1}^{N_C} c_l(\mathbf{v}) = 1$, $c_l(\mathbf{v}) \in \{0,1\}$). Further, we introduce the following parameter sets $\nu = \{\mathbf{v}_i, \mathbf{c}(\mathbf{v})\}$, $\omega_\mathbf{r} = \{\mathbf{w}_\mathbf{r}, \mathbf{y}(\mathbf{w}_\mathbf{r})\}$ and $\mathcal{W} = \{W, \mathbf{y}(W)\}$, respectively.

For the development of the FSSOM model we consider the cost function of the Heskes-SOM:

$$E_{FSSOM} = \int P(\mathbf{v})\ lc^{FSSOM}(\nu, \mathcal{W}, \sigma, \gamma)\ d\mathbf{v} \tag{4.34}$$

but with the local costs:

$$lc^{FSSOM}(\nu, \mathcal{W}, \sigma, \gamma) = \sum_{\mathbf{r}\in A} h_\sigma^{SOM}(\hat{\mathbf{s}}(\nu), \mathbf{r})\, D_\varepsilon(\nu, \omega_\mathbf{r}, \gamma). \tag{4.35}$$

where $\hat{\mathbf{s}}(\nu)$ is the winner determination by Heskes (2.10) depending on the novel deviation measure $D_\varepsilon(\nu, \omega_\mathbf{r}, \gamma)$. $D_\varepsilon(\nu, \omega_\mathbf{r}, \gamma)$ combines the distance $d(\mathbf{v}, \mathbf{w}_\mathbf{r})$ between data point and prototype in the data space and the distance between the labels $\delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_\mathbf{r}))$ in a multiplicative manner:

$$D_\varepsilon(\nu, \omega_\mathbf{r}, \gamma) = (\gamma \cdot \delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_\mathbf{r})) + \varepsilon_\delta) \cdot ((1-\gamma) \cdot d(\mathbf{v}, \mathbf{w}_\mathbf{r}) + \varepsilon_d) - \varepsilon_\delta \varepsilon_d \tag{4.36}$$

The weighting parameter $\gamma \in [0,1]$ determines the influence of the class information with $\gamma = 0$ yielding the standard Heskes-SOM. Further, $\varepsilon_\delta, \varepsilon_d \geq 0$ obtained from the parameter vector $\varepsilon = (\varepsilon_\delta, \varepsilon_d)$ is an offset term necessary in $D_\varepsilon$ to prevent unexpected behavior of the FSSOM under certain conditions concerning the update, which will be explained a bit later.

If the Euclidean distance is chosen for $d(\mathbf{v}, \mathbf{w}_\mathbf{r})$ as well as for $\delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_\mathbf{r}))$, the distance $D_\varepsilon$ is only a quasi metric (see Note 2.1), because the triangle inequality is not fulfilled. Obviously, other differential dissimilarity measures are conceivable for $d$ or $\delta$ (see page 116ff).

The FSSOM model leads to a prototype adaptation influenced by the class agreement $\delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_\mathbf{r}))$

$$\triangle\, \mathbf{w}_\mathbf{r} = (1-\gamma) \cdot (\gamma \cdot \delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_\mathbf{r})) + \varepsilon_\delta) \cdot h_\sigma^{SOM}(\hat{\mathbf{s}}(\mathbf{v}), \mathbf{r}) \cdot \frac{\partial d(\mathbf{v}, \mathbf{w}_\mathbf{r})}{\partial \mathbf{w}_\mathbf{r}} \tag{4.37}$$

accompanied by a label adaptation

$$\triangle\, \mathbf{y}(\mathbf{w}_\mathbf{r}) = \gamma \cdot ((1-\gamma) \cdot d(\mathbf{v}, \mathbf{w}_\mathbf{r}) + \varepsilon_d) \cdot h_\sigma^{SOM}(\hat{\mathbf{s}}(\mathbf{v}), \mathbf{r}) \cdot \frac{\partial \delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_\mathbf{r}))}{\partial \mathbf{y}(\mathbf{w}_\mathbf{r})} \tag{4.38}$$

such that both, prototype vectors and their class assignment vectors are parallel adapted.

Obviously, if the squared Euclidean distance is chosen in both cases, adaption of the

prototype and its label ends up in a vector shift:

$$\frac{\partial d\left(\mathbf{v},\mathbf{w_r}\right)}{\partial \mathbf{w_r}} = -2\left(\mathbf{v}-\mathbf{w_r}\right) \qquad \frac{\partial \delta\left(\mathbf{c}(\mathbf{v}),\mathbf{y}(\mathbf{w_r})\right)}{\partial \mathbf{y}(\mathbf{w_r})} = -2\left(\mathbf{c}(\mathbf{v})-\mathbf{y}(\mathbf{w_r})\right) \ . \tag{4.39}$$

Yet, in the recall phase the winner determination only depends on $d\left(\mathbf{v},\mathbf{w_r}\right)$ and thus differs to the determination in the learning phase. This disadvantage is already known from LASSO and FLSOM (see Section 2.3.1).

An advantages of the FSSOM is the additional visualization possibility. As mentioned on page 23, for the SOM exists a lot of visualization abilities like the component planes or the U-matrix. These visualization methods can be directly transfered to the FSSOM. An example is given in the application 4.5.

### 4.2.2 Fuzzy Semi-Supervised Neural Gas

The idea realized in the FSSOM can be directly transfered to the Neural Gas, i. e. the distance $d(\mathbf{v},\mathbf{w}_i)$ in the NG cost function (2.17):

$$E_{\mathrm{FSNG}} = \frac{1}{C(\sigma)} \int P\left(\mathbf{v}\right) lc^{FSNG}(\mathbf{v},W,\sigma)\ d\mathbf{v} \tag{4.40}$$

especially the distance between the data points and prototypes in the local costs $lc^{FSNG}$ (2.16) is replaced by the novel distance $D_\varepsilon\left(\nu,\omega_j,\gamma\right)$:

$$lc^{FSNG}(\nu,\mathcal{W},\sigma,\gamma) = \sum_{j=1}^{N_W} h_\sigma\left(rk_j\left(\nu,\mathcal{W},\gamma\right)\right)\ D_\varepsilon\left(\nu,\omega_j,\gamma\right) \tag{4.41}$$

It has to be pointed out that the rank function $rk_j\left(\nu,\mathcal{W},\gamma\right)$ based on the distance $D_\varepsilon\left(\nu,\omega_j,\gamma\right)$ in the learning phase. In the recall phase, the winner determination can only depend on $d(\mathbf{v},\mathbf{w})$ like it is known from the FSSOM or FLNG (see Section 2.3.3).

The learning procedure is the same like in FSSOM. The updates are derived in the same manner:

$$\begin{aligned}
\mathbf{w}_j &= \mathbf{w}_j - \alpha_W \cdot \triangle \mathbf{w}_j \\
\text{with } \triangle\, \mathbf{w}_j &= (1-\gamma)\cdot\left(\gamma\cdot\delta\left(\mathbf{c}(\mathbf{v}),\mathbf{y}(\mathbf{w}_j)\right)+\varepsilon_\delta\right)\cdot h_\sigma\left(rk_j\right)\cdot\frac{\partial d\left(\mathbf{v},\mathbf{w}_j\right)}{\partial \mathbf{w}_j} \\
\mathbf{y}(\mathbf{w}_j) &= \mathbf{y}(\mathbf{w}_j) - \alpha_Y \cdot \triangle \mathbf{y}(\mathbf{w}_j) \\
\text{with } \triangle\, \mathbf{y}(\mathbf{w}_j) &= \gamma\cdot\left((1-\gamma)\cdot d\left(\mathbf{v},\mathbf{w}_j\right)+\varepsilon_d\right)\cdot h_\sigma\left(rk_j\right)\cdot\frac{\partial \delta\left(\mathbf{c}(\mathbf{v}),\mathbf{y}(\mathbf{w}_j)\right)}{\partial \mathbf{y}(\mathbf{w}_j)}
\end{aligned}$$

where $\alpha_W, \alpha_Y \geq 0$ are the according learning rates.

The proof of the convergence for symmetric $D_\epsilon$ can be found in the appendix A.3. In

the following we only concentrate on the FSNG, beside the ideas are also transferable to the FSSOM.

### The Meaning of $\varepsilon$

We introduced the parameter vector $\varepsilon = (\varepsilon_\delta, \varepsilon_d)$ in the novel distance $D_\epsilon$, due to case; if either the prototype or the label perfectly match, the update should not vanishing:

1. $d(\mathbf{v}, \mathbf{w}) = 0$ and $\delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w})) \neq 0$, i. e. the prototype is perfectly placed but its label is not adequate: In that case a non-vanishing term

$$\left.\frac{\partial D_\varepsilon(\nu, \omega_j, \gamma)}{\partial \mathbf{y}(\mathbf{w}_j)}\right|_{d(\mathbf{v},\mathbf{w}_j)=0} = (1-\gamma) \cdot \varepsilon_d \cdot \frac{\partial \delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_j))}{\partial \mathbf{y}(\mathbf{w}_j)} \tag{4.42}$$

   remains, which guarantees the label adaptation.

2. $d(\mathbf{v}, \mathbf{w}_j) \neq 0$ and $\delta(\mathbf{c}_\mathbf{v}, \mathbf{y}(\mathbf{w}_j)) = 0$, i. e. the prototype label perfectly matches but the prototype itself is not optimally adjusted: In that case

$$\left.\frac{\partial D_\varepsilon(\nu, \omega_j, \gamma)}{\partial \mathbf{w}_j}\right|_{\delta(\mathbf{c}(\mathbf{v}),\mathbf{y}(\mathbf{w}_j))=0} = \gamma \cdot \varepsilon_\delta \cdot \frac{\partial d(\mathbf{v}, \mathbf{w}_j)}{\partial \mathbf{w}_j} \tag{4.43}$$

   is non-vanishing such that prototype learning is still possible.

In the case that both the prototype as well as its labeling are perfectly matching, i. e. $d(\mathbf{v}, \mathbf{w}_j) = 0$ as well as $\delta(\mathbf{c}_\mathbf{v}, \mathbf{y}(\mathbf{w}_j)) = 0$ the distance $D_\epsilon = 0$ and the update of the prototype and its label is zero, too.

### The Parameter for Controlling the Influence of the Labels During Learning

The quasi-metric $D_\varepsilon(\mathbf{v}, \mathbf{w}_i, \gamma)$ depends on the balancing parameter $\gamma$ weighting the unsupervised and supervised aspects. Experiences from earlier models like FLNG (see 2.3.3) suggest a careful control of this parameter beginning with $\gamma(0) = 0$ and later (adiabatic) increase up to a final value $\gamma_{\max}$, which should be chosen as $\gamma_{\max} < 1$ to avoid instabilities as known from FLNG. This can be interpreted as a remaining influence of unsupervised learning in the supervised learning phase of FSNG.

Furthermore, in semi-supervised tasks labeled and unlabeled data are given. Both kind of data can be used in FSNG. If the data are unlabeled, the wighting parameter is set to zero and thus, no update for the labeling is performed. An example for using the FSNG for such an data set can be found in Application 4.6.

**Integration of Different Kinds of Dissimilarity measures**

Like in other supervised or unsupervised VQ models, the distance $d(\mathbf{v}, \mathbf{w}_j)$ and $\delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_j))$ should be chosen data and task dependent. Other possibilities than Euclidean are for example kernel-, $l_p$- or weighted Euclidean distances.

Moreover, the learning of a linear mapping of the data can be improve the performance known from the GMLVQ. Therefore, the Euclidean distance in the data space $d_E^2(\mathbf{v}, \mathbf{w})$ is replaced by the matrix counterpart $d_E^2(\mathbf{v}, \mathbf{w}, \Omega)$ (2.68). The update of the mapping matrix $\Omega$ yields to:

$$\begin{aligned} \Omega_{kl} &\leftarrow \Omega_{kl} - \alpha_{\mathbf{\Omega}} \cdot \triangle\Omega_{kl} \\ \text{with } \triangle\,\Omega_{kl} &= -\sum_{j=1}^{N_W} (1-\gamma) \cdot (\gamma \cdot \delta\left(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_j)\right) + \varepsilon_\delta) \cdot \\ &\quad h_\sigma\left(rk_j\left(\nu, \mathcal{W}, \gamma\right)\right) \cdot \frac{\partial d_E^2(\mathbf{v}, \mathbf{w}_j, \Omega)}{\partial \Omega_{kl}} \\ \text{and} \quad \frac{\partial d_E^2(\mathbf{v}, \mathbf{w}_j, \Omega)}{\partial \Omega_{kl}} &= \left[\mathbf{v} - \mathbf{w}_j\right]_m \left[\Omega(\mathbf{v} - \mathbf{w}_j)\right]_l \ . \end{aligned}$$

An example of learning the mapping matrix $\Omega$ and the resulting *fuzzy* classification correlation matrix $\Lambda = \Omega^T\Omega$ is given in Application 4.6.

**Evaluation**

The output of the FSNG are fuzzy labels, i. e. the assignment of a prototype might be not unique. To ensure a probabilistic labeling, the additional requirement of $\sum_{k=1}^{N_C} y(\mathbf{w}_j)^k = 1$ has to be added after each update in the learning scheme.

To evaluate fuzzy classifier the classification error might be not longer suitable. Therefore another evaluation measure is mandatory. One simple possibility is the scaled Euclidean norm:

$$sEN = \frac{1}{\sqrt{2} \cdot N_V} \sum_{\mathbf{v} \in V} \|\mathbf{c}(\mathbf{v}) - \mathbf{y}(\mathbf{w}_{s(\mathbf{v})})\|_E \tag{4.44}$$

For crisp assignments, the $sEN$ is identical with the classification error $CE$ (2.28).

Another more adequate measure is the Cohens Kappa (see page 81), which compare two classifier and was extended to the Fuzzy Cohens Kappa [Geweniger, 2012].

Nevertheless, if a crisp classification is desirable, each prototype can be assign by an unique label:

$$y^{crisp}(\mathbf{w}) = \underset{k=1,\ldots,C}{\operatorname{argmax}} \left(y^k(\mathbf{w})\right)$$

and thus the standard classification accuracy can be used.

**Sparsity Constraints Concerning the Label Vector**

Although, fuzzy labels are given, in the resulted classification model the labels should focus only on a few classes. Until now, no requirements on the labeling are imposed and thus, the FSNG could end up with *very fuzzy* labels. To focus the labels to only a few classes per prototype, we include sparsity requirements. On page 2.4.2 we already introduce sparsity constraints. However, we used it for feature sparsity, the same concept can be applied to the labeling.

For a probabilistic labeling a suitable choice is an entropic penalty term:

$$E_s^H(\mathbf{y}(W)) = \sum_{j=1}^{N_W} \left( - \sum_{k=1}^{N_C} y^k(\mathbf{w}_j) \ \log(y^k(\mathbf{w}_j)) \right) \tag{4.45}$$

adding to the cost function of the FSNG (4.40):

$$E_{sFSNG} = E_{FSNG}(\mathcal{W}) + \gamma(\iota) E_s^H(\mathbf{y}(W)) \tag{4.46}$$

with the weighting function $\gamma(\iota) \geq 0$ increasing slowly and monotonic after convergence of the FSNG model. The extra term $E_s^H(\mathbf{y}(W))$ in the cost function leads to an additional update for the labels:

$$\begin{aligned} \triangle \mathbf{y}(\mathbf{w}_j) &= \frac{\partial E_{FSNG}(\mathcal{W})}{\partial \mathbf{y}(\mathbf{w}_j)} + \frac{\partial E_s^H(\mathbf{y}(W))}{\partial \mathbf{y}(\mathbf{w}_j)} \\ \text{with} \quad \frac{\partial E_s^H(\mathbf{y}(W))}{\partial \mathbf{y}(\mathbf{w}_j)} &= \log(\mathbf{y}(\mathbf{w}_j)) + 1 \ . \end{aligned}$$

**Application 4.5** (FSSOM with Visualization Possibilities)**.**

**data set** Colorado with $25\%$ labeled data

**methods** GLVQ, FLSOM, FSSOM

**parameter settings**

| | SOM |
|---|---|
| number of prototypes | $7 \times 6$ (rectangular lattice) |
| neighborhood range | $\sigma(\iota = 0) = 5, \sigma(\iota = end) = 0.1$ |
| distance function | squared Euclidean distance |
| learning rate | $\alpha_W = 0.001$ |

| | GLVQ |
|---|---|
| number of prototypes | 42 (class distributed) |
| transfer function | $f_\theta$, $\theta(\iota = 0) = 1$, $\theta(\iota = end) = 10$ |
| distance function | squared Euclidean distance |
| learning rate | $\alpha_W = 0.01$ |
| | **FLSOM** |
| number of prototypes | $7 \times 6$ (rectangular lattice) |
| neighborhood range | $\sigma(\iota = 0) = 0.1$,$\sigma(\iota = end) = 0.01$ |
| weighting parameter | $\gamma(\iota = 0) = 0.5$, $\gamma(\iota = 0) = 1$ |
| distance function | $D_{add}(\nu, \omega_j, \gamma)$ (2.49) |
| $\epsilon$ | $\epsilon_\delta = 0.1$, $\epsilon_d = 0.1$ |
| $d(\mathbf{v}, \mathbf{w}_j)$ | sq. Euclidean distance |
| $\delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_j))$ | sq. Euclidean distance |
| learning rates | $\alpha_W = 0.01$ |
| | $\alpha_Y = 0.1$ |
| | **FSSOM** |
| number of prototypes | $7 \times 6$ (rectangular lattice) |
| neighborhood range | $\sigma(\iota = 0) = 0.1$, $\sigma(\iota = end) = 0.01$ |
| weighting parameter $\gamma$ | $\gamma(\iota = 0) = 0$ and $\gamma(\iota = 0) = 1$ |
| distance function | $D_\varepsilon(\nu, \omega_j, \gamma)$ |
| $d(\mathbf{v}, \mathbf{w}_j)$ | sq. Euclidean distance |
| $\delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_j))$ | sq. Euclidean distance |
| learning rates | $\alpha_W = 0.001$ |
| | $\alpha_Y = 0.01$ |

**experimental settings**

| | |
|---|---|
| training set | $1\%$ of whole data set with $25\%$ are labeled |
| test set | $10\%$ of whole data set |
| initialization | SOM: prototypes assigned to data points |
| | GLVQ: prototypes assigned to data points |
| | FLSOM/FSSOM: trained SOM |
| epochs | SOM: 500 |
| | GLVQ/FLSOM: 250 |

**experimental description** For demonstrating the idea of semi-supervised learning we split the training data set ($N_V = 170\,111$) into a labeled and unlabeled part. We removed the label of $75\%$ of the data randomly, which is practical realistic. For the GLVQ we could only use the remaining $25\%$ of the data. For SOM, FLSOM, and FSSOM we use all data training points. The SOM prototypes are post labeled, i. e. the prototypes are assigned by the label of the best matching data point. The test data set consists of $340\,216$ ($10\%$ of the whole data set) labeled data points.

**results in numbers**

| | SOM | GLVQ | FLSOM | FSSOM |
|---|---|---|---|---|
| accuracy in % | 28.2 | 41.9 | 30.4 | 34.6 |
| $wEN$ | 0.74 | 0.58 | 0.63 | 0.61 |

**results in figures**

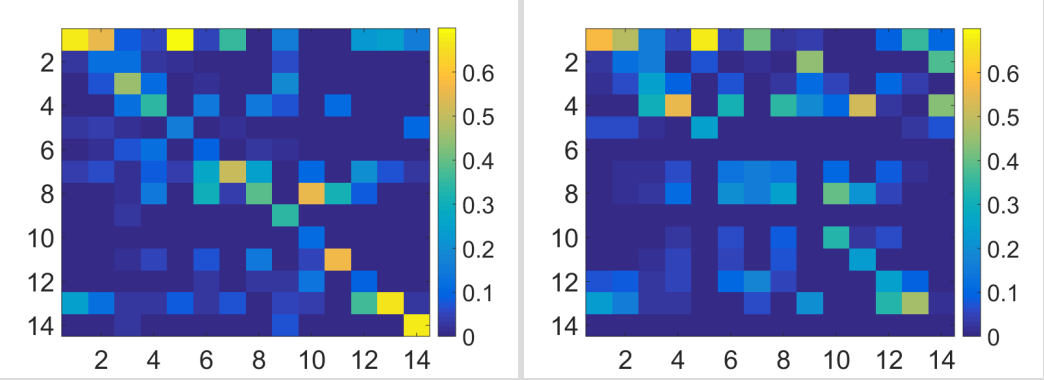


**Figure 4.22**: Confusion matrix of the GLVQ model (left) and FSSOM model (right).

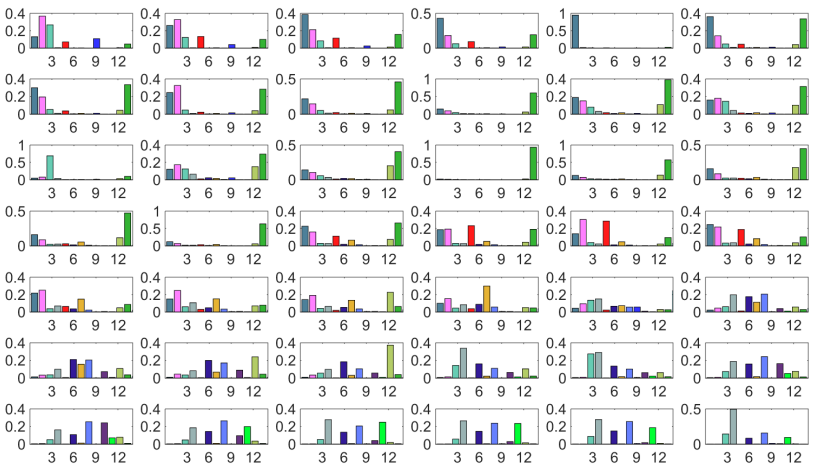

**Figure 4.23**: Barplots of the fuzzy class labels according to the FSSOM-lattice (Colors conformal to Tab. 1.1 and Fig. 1.7).

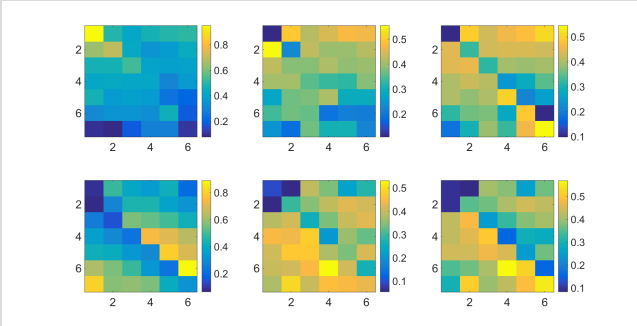

**Figure 4.24**: Component planes of the SOM model.

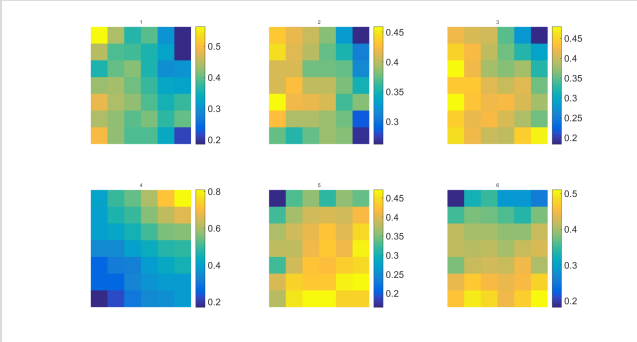

**Figure 4.25**: Component planes of the FSSOM model

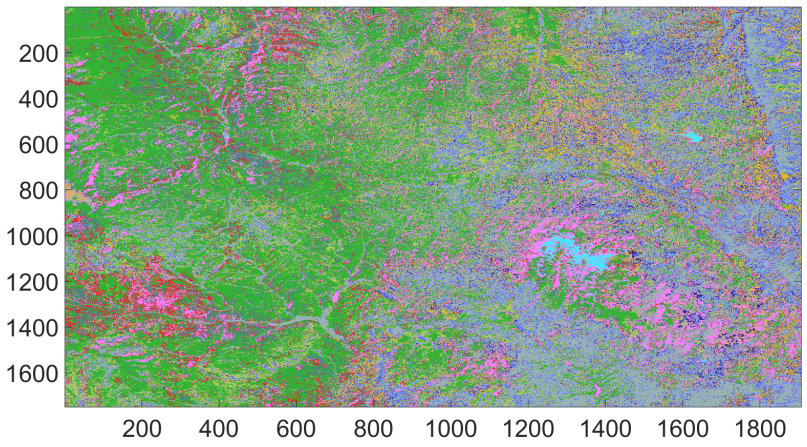


**Figure 4.26**: False color of the Colardo image, classes assigned with FSSOM

**resume** The accuracy values seems to be very small, but it has to be considered that only $1\%$ of the data were used for training and furthermore, only $25\%$ of them are labeled. Additionally, the Colorado data set is a classification problem with $14$ very unbalanced classes (see Table 1.1). The unsupervised SOM ends up with $26\%$ correct classified data samples. Adding the class information arises the accuracy values up to $30.4\%$ in FLSOM and $34.6\%$ in FSSOM. Thus, the FSSOM achieves a better accuracy value than the FLSOM. Further, the FLSOM show instable behavior in training, i. e. the results are very sensitive concerning the parameter ($\gamma$ and learning rates) selection. Yet, the result of FSSOM are lower compared to the pure classifier GLVQ which use only $25\%$ of the whole training data. One reason for this, might be the very unbalanced class distribution, e. g. for class $9$, $10$, and $14$ remain less than $100$ data points for training. This classes are represented by at least on prototype in GLVQ (initialization). Yet, the FSSOM is not able to cover these classes due to the underrepresented number of data samples. This aspect can is also reflected in the confusion matrices (see Fig. 4.22). Here the underrepresented classes like class $6$, $9$ and $14$ are not covered by the FSSOM prototypes. Instead, the GLVQ classified this classes better, which leads to the better over all accuracy of the GLVQ. Further, we demonstrate the visualization capabilities of FSSOM. For this investigation we used a two-dimensional FSSOM-lattice $A$ of $7 \times 6$ neurons. The fuzzy labels of the neurons are displayed in Fig.4.23 according to the topological structure of the neuron lattice A. Thereby, the $14$ bars in each fuzzy label subplot are assigned to different labels (the colors and sequence corresponding to Tab. 1.1). It can be seen that the prototypes ends up with very fuzzy labels, i. e. more than

one values of $\mathbf{y}(\mathbf{w})$ is significantly greater than zero. Moreover, if we concentrate on one special class, e. g. class **1** (Scotch pine), the probabilistic values $[\mathbf{y}(\mathbf{w})]_1$ are structured in the lattice. Further, the underrepresented classes like class $9$ are not considered, i. e. $[\mathbf{y}(\mathbf{w})]_9 = 0 \forall\ \mathbf{w}$. On the other side, similar classes can be also cull out of the barplots. The label values of the similar classes are simultaneously significantly greater than zero or equal to zeros. This is exemplary reflected in the values for the labels of classes $3$ (Pine/fir) and $4$ (Mixed pine fir).
The component planes of SOM and FSSOM differ from each other (see Fig. 4.24 and 4.25), which implies that the incooperation of the label information changes the prototype position. Beside the the small accuracy values the false color image is still meaningful (see Fig. 4.26), yet not such detailed like the original one (see Fig. 1.8).

**Application 4.6** (FSNG with Metric Adaptation)**.**

**data set** Colorado packed

**methods** FLNG, FSNG, FSMNG

**parameter settings**

| | |
|---|---|
| number of prototypes | 42 |
| | FLNG |
| distance function | $D_{add}(\nu, \omega_j, \gamma)$ (2.49) |
| $\epsilon$ | $\epsilon_\delta = 0.1$, $\epsilon_d = 0.1$ |
| $d(\mathbf{v}, \mathbf{w}_j)$ | sq. Euclidean distance |
| $\delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_j))$ | sq. Euclidean distance |
| learning rates | $\alpha_W = 0.01$ |
| | $\alpha_Y = 0.1$ |
| neighborhood range | $\sigma(\iota = 0) = 12$, $\sigma(\iota = end) = 0.0001$ (linear decrease) |
| weighting parameter | $\gamma(\iota = 0) = 0.05$, $\gamma(\iota = 0) = 0.75$ (linear increase) |
| | FSNG |
| distance function | $D_\varepsilon(\nu, \omega_j, \gamma)$ |
| $d(\mathbf{v}, \mathbf{w}_j)$ | sq. Euclidean distance |
| $\delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_j))$ | sq. Euclidean distance |
| $\epsilon$ | $\epsilon_\delta = 0.1$, $\epsilon_d = 0.1$ |
| learning rates | $\alpha_W = 0.01$ |
| | $\alpha_Y = 0.1$ |
| neighborhood range | $\sigma(\iota = 0) = 12$, $\sigma(\iota = end) = 0.0001$ (lin. decrease) |
| weighting parameter | $\gamma(\iota = 0) = 0.05$, $\gamma(\iota = 0) = 0.75$ (lin. increase) |
| | FSMNG |
| distance function | $D_\varepsilon(\nu, \omega_j, \gamma, \Omega)$ |
| $d(\mathbf{v}, \mathbf{w}_j, \Omega)$ | sq. Euclidean distance of mapped data ($\Omega\mathbf{v}$, $\Omega\mathbf{w}_j$) |
| $\delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_j))$ | sq. Euclidean distance |
| $\epsilon$ | $\epsilon_\delta = 0.1$, $\epsilon_d = 0.1$ |
| learning rates | $\alpha_W = 0.01$ |
| | $\alpha_Y = 0.1$ |
| | $\alpha_\Omega = 10^{-5}$ |
| neighborhood range | $\sigma(\iota = 0) = 0.5$, $\sigma(\iota = end) = 0.0001$ (lin. decrease) |
| weighting parameter | $\gamma(\iota = 0) = 0.15$, $\gamma(\iota = 0) = 0.5$ (lin. increase) |

**experimental settings**

| | |
|---|---|
| initialization | FLNG, FSNG - randomly |
| | FSMNG - FSNG model |
| training/test | 3-fold cross validation |
| epochs | 500 |

**results in numbers** scaled Euclidean norm $sEN$ (4.44)

| | FLNG | FSNG | FSMNG |
|---|---|---|---|
| training | $0.226(\pm 0.005)$ | $0.211(\pm 0.003)$ | $0.185(\pm 0.003)$ |
| test | $0.226(\pm 0.006)$ | $0.212(\pm 0.002)$ | $0.186(\pm 0.005)$ |

**results in figures**

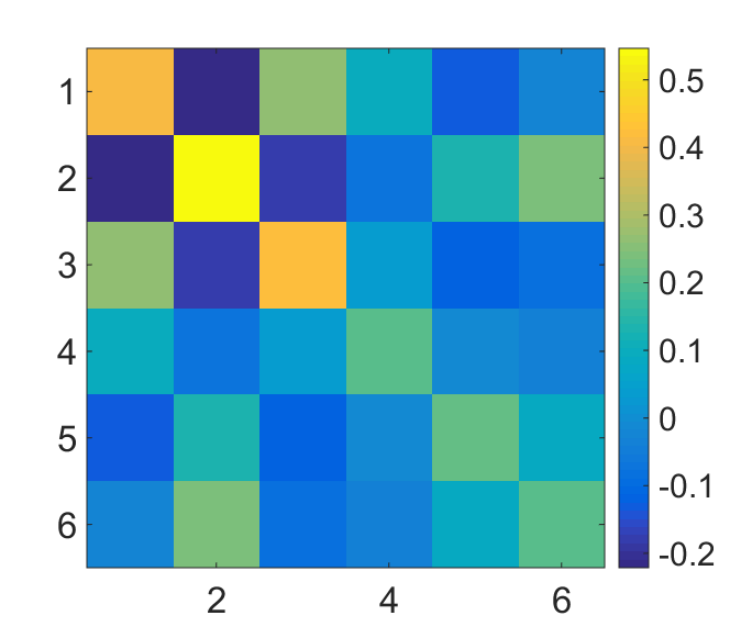


**Figure 4.27**: Mean relevance matrix learned with the FSMNG.

**resume** In data set *Colorado packed* the labels are fuzzy, i. e. a label vector contain the percentage value of the according soil condition in an area. In this case, a standard classifier like GLVQ can not be applied. Yet, the FSNG and FLNG are able to handle such fuzzy label vectors. The FLNG ends up with a mean scaled Euclidean norm of $0.226$. Applying the FSNG a slightly better value of $0.212$ is obtained. Yet, the most benefit we obtain with the matrix learning in FSNG. The $sEN$ falls down to $0.185$. In the classification correlation matrix the dimension $2$ has the highest value and additionally the dimension $1$ and $3$ are emphasized. Moreover, the dimension $2$ is negative correlated to dimension $1$ and $3$. Thus, the visible spectra play an important roll to distinguish the ground covers.

# Chapter 5

# Variants of Classification Costs and Class Sensitive Learning in Prototype Based Methods

**This section is based on**

*M. Kaden, M. Riedel, W. Hermann, and T. Villmann:* Border-sensitive Learning in Generalized Learning Vector Quantization: an Alternative to Support Vector Machines. *Soft Computing, 2015.*

## 5.1 Border Sensitive Learning in Generalized LVQ

A significant difference between the SVM and the standard GLVQ is the kind of margin which is optimized. The SVM optimizes the separation margin, i. e. the optimization concentrates to the class borders and to the finding of an optimal decision hyperplane (see appendix A.2). In contrast, the GLVQ optimizes the hypothesis margin and end ups frequently with class describing prototypes. In e. g. [**?**] it is pointed out that the task of class separation and class description might be conflicting in the general case. In theory, only the LVQ 1 generate class typical prototypes, but this statement does not hold for the GLVQ. Yet, in [Hammer et al., 2014b] are shown possibilities to achieve class typical prototypes in cost function based LVQ variants.

In this Section, we want to end up with class border sensitive prototypes for LVQ-learning and describe two approaches to modify the GLVQ accordingly. The first ansatz is adding a penalty term to force the prototypes to the decision border (P-GLVQ) and in the second approach the transfer function $f_\Theta$ (2.38) of the GLVQ cost function (2.37) is considered closer (BS-GLVQ).

### 5.1.1 Border Sensitivity by Additive Penalty Function

In [Hammer et al., 2014b] is pointed out a possibility to achieve class typical prototypes by adding a penalty function to the standard cost function of the GLVQ. This idea can be transfered to obtain class border typically prototypes like in the SVM, the Penalty-GLVQ (P-GLVQ). Therefore, the cost function of the GLVQ (2.37) is extended

by a weighted additive penalty term:

$$E_{P-GVLQ} = (1-\gamma)E_{GLVQ} + \gamma P_{NG} \tag{5.1}$$

This penalty term $P_{NG}$ should force the prototypes to the decision border and the parameter $\gamma \in [0,1]$ control the influence of this constraint. A possibility is to choose the penalty term as neighborhood attentive attraction force (NAAF):

$$P_{NG} = \sum_{\mathbf{v}\in V} \sum_{k:\mathbf{w}_k \in W^-(\mathbf{v})} h_\sigma^{NG}(rk_k(\mathbf{w}^+, W^-(\mathbf{v}))\; d(\mathbf{w}^+, \mathbf{w}_k) \tag{5.2}$$

with the set $W^-(\mathbf{v}) = \{\mathbf{w}|y(\mathbf{w}) \neq c(\mathbf{v})\}$ including all prototypes of the incorrect class of the sample $\mathbf{v}$. Further, the neighborhood function known from the Neural Gas (2.15)

$$h_\sigma^{NG}\left(rk_j\left(\mathbf{w}^+, W^-(\mathbf{v})\right)\right) = \exp\left(-\frac{rk_j\left(\mathbf{w}^+, W^-(\mathbf{v})\right)}{2\sigma^2}\right) \tag{5.3}$$

defines the rank of the prototypes in $W^-(\mathbf{v})$ via the dissimilarity function

$$rk_j\left(\mathbf{w}^+, W^-(\mathbf{v})\right) = \sum_{\mathbf{w}_k \in W^-(\mathbf{v})} \mathtt{H}\left(d\left(\mathbf{w}^+, \mathbf{w}_k\right) - d\left(\mathbf{w}^+, \mathbf{w}_k\right)\right) \tag{5.4}$$

with the Heaviside function $\mathtt{H}(x)$ (2.14).

Thus, NAAF (5.2) sums up the costs or distances for all data points, respectively, between the winning prototype and the prototypes of a different class weighted by their ranks. This term causes an additional term in the prototype update (2.40) of the standard GLVQ

$$\begin{aligned}
\frac{\partial P_{NG}}{\partial \mathbf{w}^+} &= \sum_{k:\mathbf{w}_k \in W^-(\mathbf{v})} h_\sigma^{NG}\left(rk_k\left(\mathbf{w}^+, W^-(\mathbf{v})\right)\right) \cdot \frac{\partial d(\mathbf{w}^+, \mathbf{w}_k)}{\partial \mathbf{w}^+} \\
\frac{\partial P_{NG}}{\partial \mathbf{w}_j} &= h_\sigma^{NG}\left(rk_j\left(\mathbf{w}^+, W^-(\mathbf{v})\right)\right) \cdot \frac{\partial d(\mathbf{w}^+, \mathbf{w}_j)}{\partial \mathbf{w}_j} \quad \mathbf{w}_j \in W^-(\mathbf{v})
\end{aligned}$$

which gradually attract all prototypes of $W^-(\mathbf{v})$ to the winning prototype $\mathbf{w}^+$ depending on the rank $rk_j\left(\mathbf{w}^+, W^-(\mathbf{v})\right)$ and the neighborhood range $\sigma \geq 0$. Especially, the next prototype $\mathbf{w}_j \in W^-(\mathbf{v})$ to $\mathbf{w}^+$, i. e. $d(\mathbf{w}^+, \mathbf{w}_j) \leq d(\mathbf{w}^+, \mathbf{w}_k)\; \forall \mathbf{w}_k \in W^-(\mathbf{v})$ has the rank zero and will be moved towards $\mathbf{w}^+$ strongest. Further, the parameter $\gamma$ regulates the influence of NAAF and thus the strength of the border typical constrain.

The additive NAAF term in 5.1 forces the prototypes closer to the decision border and hence, the prototypes become more border sensitive implicitly. An advantage of this method is that only a subset of prototypes can be restricted by this penalty term. Therewith, on the one side there are prototypes learned with the standard GLVQ, which

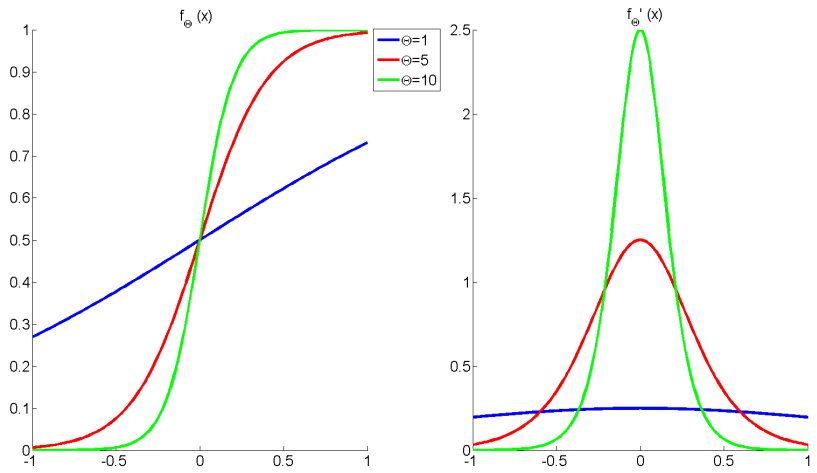


**Figure 5.1**: The shape of the sigmoid and sigmoid prime function depending on the parameter $\theta$.

ends up class typical. On the other side there are prototypes learned with the P-GLVQ and these prototypes are class-border representative.

Yet, an additional control of the weighting parameter $\gamma$ is required, which is not easy to handle (see Application 5.1). Therefore, an approach which is more GLVQ-inherent would be desirable.

### 5.1.2 Border Sensitivity by Parameterized Transfer Function

In Section 2.2 we mentioned that the cost function GLVQ approximate the classification error. The classifier function $\mu_W(\mathbf{v}) < 0$ (2.34), iff the data point is correct classified. Particularly, when using the transfer function from (2.38):

$$f_\theta(x) = \frac{1}{1 + e^{-\theta \cdot x}} \ ,$$

with the parameter $\theta > 0$ determines the shape of increase especially of the interval $\mu_W(\mathbf{v}) \in [-1, 1]$ (see Fig. 5.1). If $\theta = 1$, the sigmoid function is nearly linear in the interval $[-1, 1]$ and thus the training in GLVQ has the same behavior, than the identical function is chosen for the transfer function. Moreover, for $\theta \nearrow \infty$, the $f_\theta(x)$ converges to the Heaviside function. Thus, the following remark is valid:

**Remark 5.1.** If $\theta \nearrow \infty$, the cost function of the GLVQ converters to the classification error.

This perception yields to further applications of the GLVQ-principle (see Sec. 5.2).

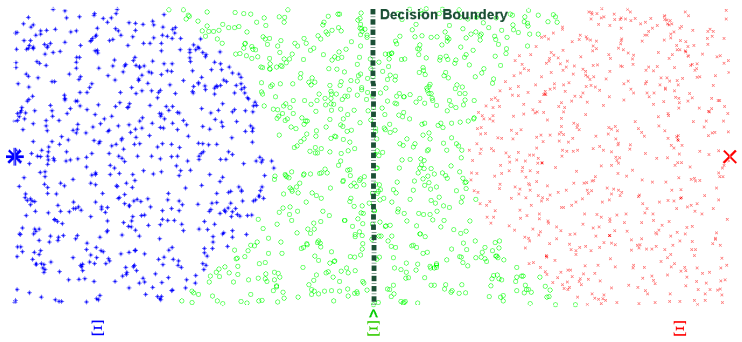


**Figure 5.2**: Illustration of the active set $\hat{\Xi}(V, W)$ and the *passive set* $\Xi(V, W)$ for the prototypes $\mathbf{w}_1$ and $\mathbf{w}_2$.

The derivative

$$f'_\theta(x) = \theta \cdot f_\theta(x) \cdot (1 - f_\theta(x)) \tag{5.5}$$

of the sigmoid function appears as scaling factor in the update of the prototypes (2.39). In Figure 5.1 is also apparent: the higher $\theta$, the smaller the range of the classifier value $\mu_W(\mathbf{v})$ with no vanishing values for $f'_\theta(\mu_W(\mathbf{v}))$. Thus, a significant prototype update, i. e. $|\xi^{\pm}| \gg 0$ (2.41), holds only for $|\mu_W(\mathbf{v})| \ll 1$, which implies $d^+(\mathbf{v}) \approx d^-(\mathbf{v})$. To describe the set of data points influencing the prototype update at least moderately, we define the *active set*

$$\hat{\Xi}(V, W) = \left\{ \mathbf{v} \in V | \mu_W(\mathbf{v}) \in \left[ -\frac{1-\mu_\theta}{1+\mu_\theta}, \frac{1-\mu_\theta}{1+\mu_\theta} \right] \right\} \tag{5.6}$$

with $|f'_\theta(\mu_\theta)| < \epsilon$ given a small $\epsilon > 0$. In Figure 5.2 an example of an active set with the counter part the *passive set* $\Xi(V, W) = V \backslash \hat{\Xi}(V, W)$ is visualized. Thus, the data points belonging to the active set contribute to a prototype shift, and on the other side, all data points belonging to the passive set have no or only a very small impact to the learning.

As can easily be seen, data points along the decision boundary belonging into the active set. The range around the boundary depends on the parameter $\theta$, i. e. as smaller $\theta$, as smaller the active set and therefore, as smaller the range around the boundary. Further, the active set can be interpreted as a generalized window rule known from the LVQ 2 (2.33) with $\omega = \theta$ [Witoelar et al., 2010]. Obviously, the active sets idea holds also for the metric adaption like in GRLVQ, GMLVQ (Sec. 2.4) or DK-GLVQ (Sec. 2.4.1).

A good choice of the parameter $\theta$ is task dependent. As higher the value for $\theta$, as better the approximation of the classification error and additional the prototypes become more border sensitive. It has to be taken into account: a high value for $\theta$ also implies a contribution of only a small set of data points to the learning, i. e. the range of the active

set around the decision border become very small. This might be cause a worse generalization, because only the data around the decision border is focused and sensitive for outliers in this region.

Another aspect is the numerical instability if $\theta$ is too high. This is based on the fact that $f'_\theta(\mu_W(\mathbf{v}))$ gets infinity if $\theta$ does. Therefore, in [Strickert, 2011] is suggested to replace the SGD by a 2nd order optimization scheme. Another possibility is to increase slowly the parameter $\theta$ during the learning process. Thereby, we start with $\theta = 1$, where all data points has nearly the same influence in the update concerning the factor $f'_\theta(\mu_W(\mathbf{v}))$ and then slowly increase the value for $\theta$. This strategy is applied in Application 5.1.

In summary, the border sensitivity can be controlled by the parameter $\theta$ when using $f_\theta$ as transfer function in GLVQ. Yet, it has to pointed out, that border sensitive prototypes does not imply that the prototypes are next to the decision border in this approach. We abbreviate this approach by BS-GLVQ. Thus BS-GLVQ differs from P-GLVQ.

If the BS-GLVQ is combined with DK-GLVQ, we achieve a model which generates border sensitive prototypes equipped with differentiable kernels $d_\kappa$. In Subsection 2.4.1 it is shown that the spaces $(V, d_\kappa)$ (DK-GLVQ) and $(\mathcal{I}_\kappa, d_\kappa)$ where $d_\kappa$ is an universal kernels provides the same topological richness. Hence, together with the additional requirement of border sensitivity, we obtain a counterpart model to the SVM.

In Application 5.1, experiments on artificial data sets show the principles of BS-GLVQ and P-GLVQ. Further, the BS-GLVQ is compared to the SVM in an example in Application 5.2.

**Application 5.1** (Possibilities for Border-Sensitive Prototypes in GLVQ)**.**

**data set** Czech-flag

**methods** GLVQ, P-GLVQ, BS-GLVQ

**parameter settings**

| | |
|---|---|
| number of prototypes | 3 \| 15 (equal class distributed) |
| distance function | $d_E^2$ |
| learning rates | $\alpha_W = 0.01$ |
| transfer function | GLVQ, P-GLVQ: $\theta = 1$ |
| | BS-GLVQ: $\theta(\iota_{start}) = 1$,$\theta(\iota_{end}) = 20$ |
| balancing parameter | $\gamma = 0.5$ for P-GLVQ |

**results in numbers**

| $\|W\|$ | GLVQ | P-GLVQ | BS-GLVQ |
|---|---|---|---|
| 3 | 89.7% | 91.1% | 97.2% |
| 15 | 97.4% | 91.2% | 99.4% |

**results in figures**

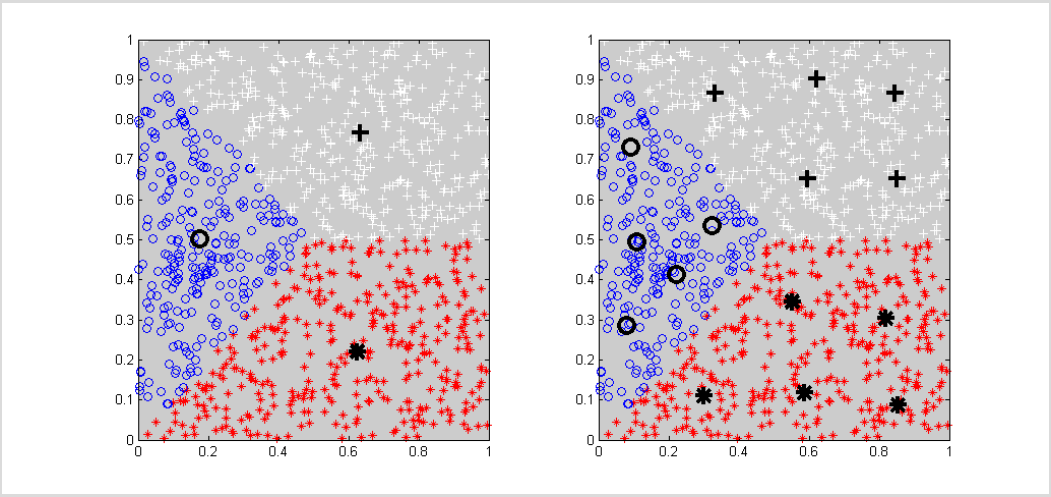

**Figure 5.3**: The resulting prototypes of the standard GLVQ with linear transfer function ends up with class typical prototypes.

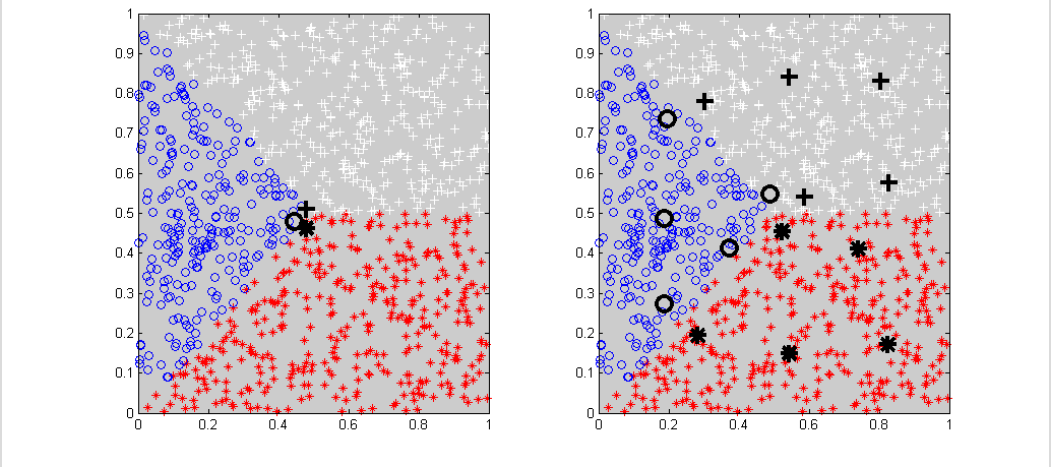

**Figure 5.4**: Left: Applying only three prototypes, the P-GLVQ results in class border sensitive prototypes. Right: In contrast, the P-GLVQ with five prototypes per class ends up only with a subset of prototypes which are next to the class border. A few prototypes a still remains class representative.

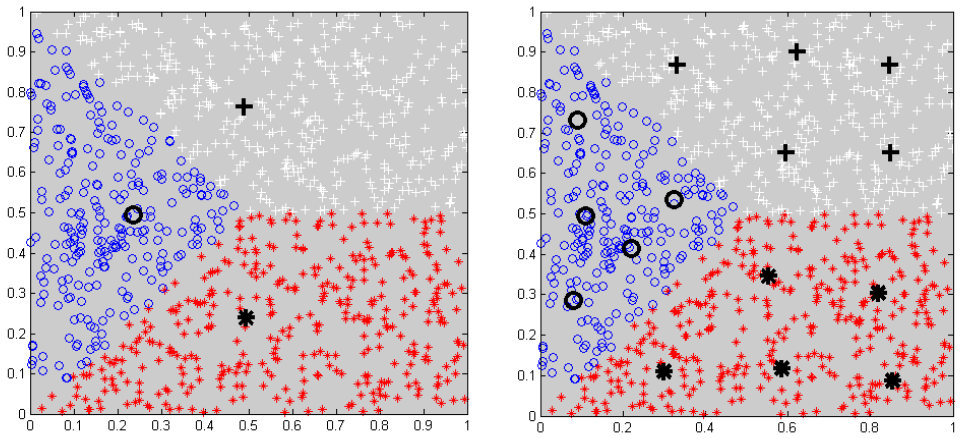

**Figure 5.5**: The position of the resulting three prototypes of BS-GLVQ seems neither class nor class border representative. Yet, they are border-sensitive due to the theory of the active sets.

**resume** This toy example shows the diversity of class representative versus class border representative as well as representative versus discriminative prototypes obtained by P-GLVQ and BS-GLVQ, respectively.
The GLVQ ends up with class representative prototypes (see Fig. 5.3). Yet, the GLVQ model can not achieve the same performance like the BS-GLVQ even for the sparse model ($N_W = 3$) like the P-GLVQ.
The three prototypes learned with the P-GLVQ (see Fig. 5.4) laying on the decision border and class border representative according to the three classes. If more prototypes are applied, only a subset of prototypes ends up near the decision border whereas the others are still representative (see Fig. 5.4).
The best accuracy is achieved by BS-GLVQ. The reason for this results is that the approximation of the goal (minimizing the classification error) gets better with higher $\theta$. Hence, the resulting prototypes are neither class nor class border representative (see Fig. 5.5), but class border sensitive. The active set for $\theta = 20$ consist of 313 data points (31% of whole data set).

**Application 5.2** (DK-GLVQ vs SVM)**.**

**data set** Morbus Wilson (2cl)

**methods** DK-GLVQ, SVM

**parameter settings**

| | DK-GLVQ |
|---|---|
| number of prototypes | 12 (6 per class) |
| initialization | prototypes assigned to data points |
| distance function | $d_{\kappa_{RBF}}$ with initialized $\sigma_j = 2.5$ (local, adapted) |
| learning rates | $\alpha_W = 0.1$ |
| | $\alpha_\sigma = 0.0005$ |
| | SVM |
| kernel | $\kappa_{lin}$ |
| | $\kappa_{RBF}$, $\sigma = 0.015$ (systematic search) |
| penalty value | $C = 200$ |

**experimental settings**

| | |
|---|---|
| number of CV-folds | 10 |
| epochs | 2500 |

**descriptions** In the DK-GLVQ, we learn beside the prototypes also the parameters of the RBF-kernel for each prototype $\sigma_{\mathbf{w}_k}$. In the SVM we determine the parameter $\sigma$ and $C$ by systematic search.

**results in numbers** accuracy ($\pm$ standard deviation) in $\%$

| | DK-GLVQ | BS-DK-GLVQ | | SVM | |
|---|---|---|---|---|---|
| | $\theta = 1$ | $\theta = 2.5$ | $\theta = 5$ | lin | RBF |
| training | 87.8 ($\pm$0.013) | 91.9 ($\pm$0.015) | 90.4 ($\pm$0.014) | 86.1 ($\pm$0.012) | 87.5 ($\pm$0.015) |
| test | 81.9 ($\pm$0.086) | 82.6 ($\pm$0.086) | 87.4 ($\pm$0.090) | 85.5 ($\pm$0.013) | 87.4 ($\pm$0.137) |
| | $N_P = 12$ | $N_P = 12$ | | $\|SV\| = 43.3$ | $\|SV\| = 45.5$ |

**resume** The Morbus Wilson data set with the two classes *neurological* and *non-neurological* is not easy to distinguish by the electrophysiological impairment profile (see Sec. 1.2). Yet, the given classifier achieve an averaged accuracy for the test data sets up to $87.4\%$. Thus, a distinction of the two classes is possible. The results of the BS-DK-GLVQ and the SVM each with the RBF-kernel equipped ends up with the same results in the test accuracy. However, on the one side the BS-DK-GLVQ has an smaller standard deviation, which suggest an more stable behavoir. On the other side the model complexity in the BS-DK-GLVQ is smaller compared to the SVM, i. e. the number of prototypes is smaller as the averaged number of Support Vectors.

**Parts of the section are based on**

*M. Kaden, W. Hermann, T. Villmann:* Optimization of General Statistical Accuracy Measures for Classification Based on Learning Vector Quantization. *Proc. of ESANN, 2014.*

*M. Kaden, W. Hermann, T. Villmann:* Attention Based Classification Learning in GLVQ and Asymmetric Misclassification Assessment. *Proc. of WSOM, 2014.*

## 5.2 Optimizing Different Validation Measures by the GLVQ

In Section 3 it is already mentioned that an optimal accuracy is not always the goal of a classification task. Depending on the classification problem with the underlying data and class properties other evaluation measures might be more appropriate. Examples are the weighted accuracy or statistical measures based on the confusion matrix like F-measure, G-measure or the Matthews correlation coefficient (see Sec. 3.3). In the Remark 5.1, we pointed out that the GLVQ approximated the classification accuracy or the error, respectively, i. e. if $\theta \nearrow \infty$, then $E_{GLVQ} \approx err(V, W)$. This insight enables the extension of the GLVQ for optimizing statistical classification evaluation measures. In Section 5.2.2 an approach is introduced which optimizes any differentiable statistical measure based on the confusion matrix by keeping the GLVQ principle.
Unfortunately, the most statistical measures are only for two-class problem. Generalizations are often only heuristics and based on the one-versus-all principle. However in multi-class problems, measures beside the accuracy might become important if the class distribution is very unbalanced or even single classes are more important due to the application. Thus, a generalization of the accuracy is the weighted accuracy, which consider different class priors. Further, not only the classes per se are important, but several misclassifications might be avoided or have different costs, respectively. This is for example known from medicine: It is given a group of persons. To one subgroup belong persons, suffering from a medical condition whereas the others do not. Now, if an healthy person is misclassified, unnecessary costs of drugs occur and the person might be suffer from undesirable side effects. The other way around, if an affected person is misclassified to be healthy, he does not get an appropriate treatment. This example show, that different misclassifications lead to diverse aftermaths or costs, respectively. These costs depend on the application and should be given by an expert. A general ansatz to consider auxiliary knowledge about the classes or asymmetric misclassifications cost is given in the next subsection.

### 5.2.1 Attention Based Learning Strategy

In the cost function of the GLVQ (2.37) the term $f_\theta(\mu_W(\mathbf{v}))$ is approximated 1 if $\mathbf{v}$ is misclassified and for a large $\theta$. Thus, to weight the importance of single classes or to

penalty different misclassifications, the cost function of the GLVQ can be extended by a multiplicative function $\varphi$:

$$E_{\varphi-GLVQ} = \sum_{\mathbf{v}\in V} \varphi(\Phi_V(\mathbf{v}, c(\mathbf{v})), \Psi_W(W, Y(W))) \cdot f_\theta(\mu_W(\mathbf{v})) \tag{5.7}$$

The formal function $\varphi$ should reflects the auxiliary assumptions or restriction about the vector space including the labeling and the prototypes coded in the free parameters $\Phi_V(\mathbf{v}, c(\mathbf{v}))$ and $\Psi_W(W, Y(W))$, respectively. The specific structure of $\varphi$ reflect the external conditions on the classification problem and, hence, is task dependent. In the following, we describe two possibilities for $\varphi$ closer.

**Class Priors**

Assume, in a given application the number of example of a class is very unbalanced, e. g. in the Indian Pine data set (see Sec. 1.2). Nevertheless, the comparatively underrepresented classes might be not less important. Thus, the standard accuracy might be not appropriated to evaluate unbalanced data set regarding the class distribution. A simple alternative is to use the weighted accuracy

$$wacc_\beta(V, W) = \frac{1}{N_V} \sum_{\mathbf{v}\in V} \beta(c(\mathbf{v}))\ \delta_{c(\mathbf{v}),\hat{c}(\mathbf{v})} \tag{5.8}$$

with $\delta_{i,j}$ the Kronecker delta (2.29) and the weighting vector $\beta \in \mathbb{R}^{N_C}$ with components $\beta(k) > 0\ \forall k$. The vector $\beta$, given by an expert, might include the class priors or relative importances of the classes.
Beside the modified validation measure, the classifier should take these class weights into account, too. The integration of $\beta$ in GLVQ cost function is very simple: The formal weighting function $\varphi$ from (5.7) becomes

$$\varphi(\Phi_V(\mathbf{v}, c(\mathbf{v})), \Psi_W(W, Y(W))) = \beta(c(\mathbf{v})). \tag{5.9}$$

The respective update of the prototypes is obtained as

$$\mathbf{w}^\pm \leftarrow \mathbf{w}^\pm - \alpha_W\ \beta(c(\mathbf{v}))\ \xi_\theta^\pm(\mathbf{v}) \cdot \frac{\partial d^\pm(\mathbf{v})}{\partial \mathbf{w}^\pm} \tag{5.10}$$

with the *local learning rate* $\alpha_W\ \beta(c(\mathbf{v}))$ and $\xi_\theta^\pm(\mathbf{v})$ given in (2.41). Thus, the update can also be interpreted as a kind of *attention based learning* [Herrmann et al., 1994]. In this case, the attention based learning is concentrated on the classes. A finer resolution can be obtained, if the weighting vector $\beta$ depends on $\mathbf{v}$. Then, each data point is equipped with a separate data point dependent learning rate.

**Asymmetric Misclassification Assessment**

In several applications not the single classes have different priors, but the misclassifications from one class to another might have different costs. An example can be found if a classification problem consists of main classes, which are divided into sub classes. Thereby it might be less costly if we misclassify data points within the main classes, but more costly if a data point is failed the main classes.
In other words, the non-diagonal elements in a confusion matrix have different consequences in the performance of a classifier model. To take these misclassification costs into account a weighted error can be designed:

$$werr_{\mathbf{\Gamma}}(V,W) = \sum_{\mathbf{v}\in V} \Gamma(c(\mathbf{v}),\hat{c}(\mathbf{v}))\ (1-\delta_{c(\mathbf{v}),\hat{c}(\mathbf{v})}) \tag{5.11}$$

with the *asymmetric misclassification assessment matrix* (AMAM) $\Gamma \in \mathbb{R}^{N_C\times N_C}$ and the single entries $\gamma(i,j) \geq 0$. Thereby, $\gamma(i,j) \in \Gamma$ defines the costs of a misclassification of a point of class $i$ to class $j\ \forall i \neq j$ and $\gamma(i,i)$ can be set arbitrary, because theses values do not contribute in (5.11), i. e. $\delta_{i,i} = 1$. Further, we assume a non-symmetric matrix $\Gamma$ to assure asymmetric misclassification costs. The AMAM $\Gamma$ has to be given by an expert in advanced.
Obviously, the AMAM can be integrated into the GLVQ cost function to regulate the penalizing for misclassifications of class $c(\mathbf{v})$ to $\hat{c}(\mathbf{v})$. Thereto, the function $\varphi$ of $E_{\varphi-GLVQ}$ (5.7) is set to:

$$\varphi(\Phi_V(\mathbf{v},c(\mathbf{v})),\Psi_W(W,Y(W))) = \gamma(\hat{c}(\mathbf{v}),c(\mathbf{v})) \tag{5.12}$$

Again, for the update we obtain:

$$\mathbf{w}^{\pm} \leftarrow \mathbf{w}^{\pm} - \alpha_W\ \cdot\ \gamma(c(\mathbf{v}),\hat{c}(\mathbf{v}))\ \cdot\ \xi_\theta^{\pm}(\mathbf{v})\ \cdot\ \frac{\partial d^{\pm}(\mathbf{v})}{\partial \mathbf{w}^{\pm}} \tag{5.13}$$

which can be also interpret as *attention based learning*, but in this case concerning the explicit penalizing of specific misclassifications.
Evidently, the weighting of a class and also the costs of misclassification can be merged. Thereby, the main diagonal elements of the AMAM are set to the class weights, i. e. $\gamma(i,i) = \beta(i)$ with $\beta(i)$ from (5.8).
Thus, with these weighted variants of the GLVQ it is possible to integrate auxiliary knowledge about the classes or misclassifications, respectively. This can be a ranking according the importance of the classes, the class priors coded in the weighting vector $\beta$, or the specific costs of misclassifications coded in the penalty matrix $\Gamma$. However, $\beta$ or $\Gamma$ have to be given in advanced by an expert.
The respective $\beta-$GLVQ and $\Gamma-$GLVQ are applied on a toy example to demonstrate the behavior of the positions of the prototype with respect to $\beta$ or $\Gamma$, respectively (see

Appl. 5.3). Furthermore, we applied the attention based learning GLVQ on a real world application to give an example of choosing $\beta$ or $\Gamma$, respectively (see Appl. 5.4). Other variants can easily be creative analogously.

**Application 5.3** (Attention Based Classification Learning in GLVQ - Artificial Data Set)**.**

**data set** $\triangle$-data

**methods** $\beta$-GLVQ, $\Gamma$-GLVQ

**parameter settings**

| | |
|---|---|
| number of prototypes | 6 (2 per class) |
| initialization | GLVQ: randomly chosen data |
| | $\beta$-GLVQ, $\Gamma$-GLVQ:GLVQ model |
| distance function | $d_E^2$ |
| learning rates | $\alpha_W = 0.0001$ |
| transfer function parameter | $\theta(\iota_{start}) = 1$,$\theta(\iota_{end}) = 10$ |

**experimental settings**

| | |
|---|---|
| training | all data points |
| epochs | 1000 |

**descriptions** For the toy example the weighting vector $\beta$, the AMAM $\Gamma_1$, and the combination of both $\Gamma_2$ are chosen arbitrary to demonstrate the effect on the localization of the learned prototypes and the effect of the integration of the weights/costs to the confusion matrix

$$\beta(c(\mathbf{v})) = \begin{pmatrix} 5 \\ 1 \\ 1 \end{pmatrix} \quad \Gamma_1(c(\mathbf{v}), \hat{c}(\mathbf{v})) = \begin{pmatrix} 1 & 5 & 1 \\ 1 & 1 & 5 \\ 5 & 1 & 1 \end{pmatrix}$$

$$\Gamma_2(c(\mathbf{v}), \hat{c}(\mathbf{v})) = \begin{pmatrix} 5 & 5 & 1 \\ 1 & 1 & 5 \\ 5 & 1 & 1 \end{pmatrix}$$

**results in numbers** confusion matrices

| GLVQ | | real | | |
|---|---|---|---|---|
| | | ○ | + | * |
| predicted | ○ | 88 % | 6 % | 10 % |
| | + | 7 % | 86 % | 4 % |
| | * | 5 % | 8 % | 86 % |

| $\beta$-GLVQ | | real | | |
|---|---|---|---|---|
| | | ○ | + | * |
| predicted | ○ | 99 % | 22 % | 29 % |
| | + | 0 % | 70 % | 4 % |
| | * | 1 % | 8 % | 67 % |

| $\Gamma_1$-GLVQ | | real | | |
|---|---|---|---|---|
| | | ○ | + | * |
| predicted | ○ | 83 % | 1 % | 16 % |
| | + | 15 % | 90 % | 4 % |
| | * | 2 % | 9 % | 80 % |

| $\Gamma_2$-GLVQ | | real | | |
|---|---|---|---|---|
| | | ○ | + | * |
| predicted | ○ | 91 % | 5 % | 15 % |
| | + | 7 % | 87 % | 4 % |
| | * | 2 % | 8 % | 81 % |

**results in figures**

The *Triangle* data set with the data points of the three classes labeled with ○, + and *, respectively. The prototypes are drawn as ◇ with the color according to their classes. The crosses × mark the misclassified points.

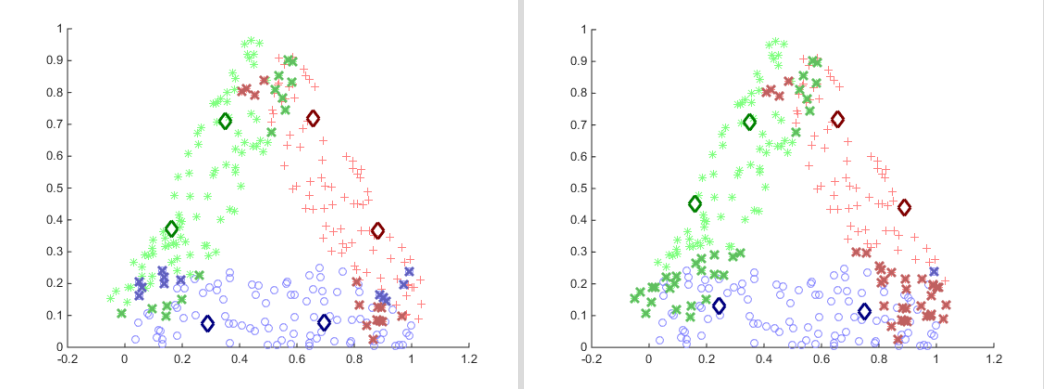

**Figure 5.6**: Position of the learned prototypes with standard GLVQ (left) and weighted $\beta$-GLVQ

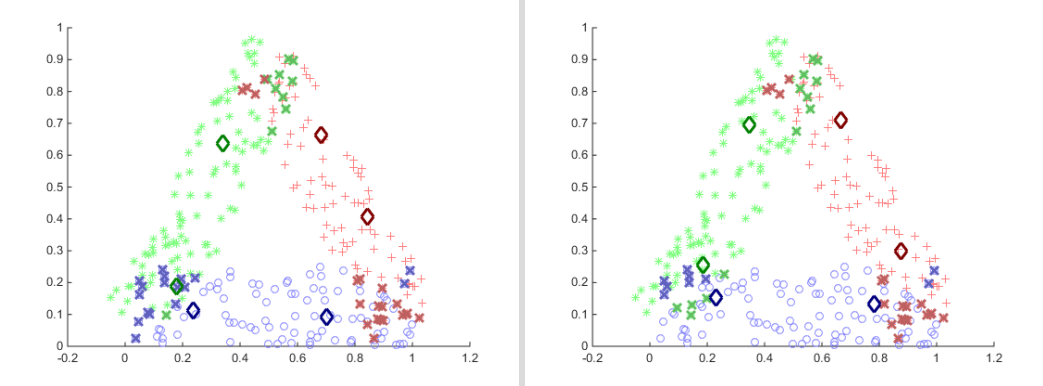

**Figure 5.7:** Position of the learned prototypes with $\Gamma_1$-GLVQ (left) and $\Gamma_2$-GLVQ (right)

**resume** In the triangle data set the 3 classes overlap each other in the corner. The overlapping is nearly equal distributed (see Fig. 5.6). This data behavior is reflected in the nearly symmetric confusion matrix of the GLVQ. The $\beta$-GLVQ weighted the first class $\circ$ twice as much as the second $+$ and third $*$ class. This effects a nearly perfect correct classification of class $\circ$, of course to the detriment of accuracy of the other two classes. The weighted accuracy is $wacc_\beta = 139$ (compare for standard GLVQ $wacc_\beta = 100$)
Applying the AMAM $\Gamma_1$ effects an asymmetric confusion matrix whereby the rates of misclassifications with high costs are drastically dropped down. Yet, the misclassification with low costs arises more frequently. The weighted error decrease from $werr_{\Gamma_1}(V, W) = 100$ of GLVQ to $werr_{\Gamma_1}(V, W) = 75$ learned with $\Gamma_1$-GLVQ. The position of the prototypes reflect this behavior, too (see Fig. 5.7): The two prototypes of class $\circ$ moves to left. Thus the number misclassification of $\circ \rightarrow *$ reduces and on the other side the number of misclassification $\circ \rightarrow +$ increase.
The combination of both, i.e. the weighting of the classes and weighted penalizing of misclassification causes a compromise between the two. The weighted error is $werr_{\Gamma_2}(V, W) = 85$ and the weighted accuracy $wacc_\beta = 607$.

**Application 5.4** (Attention Based Classification Learning in GLVQ - Real World Application)**.**

**data set** Morbus Wilson (EIP4)

**methods** Γ-GLVQ

**parameter settings**

| | |
|---|---|
| number of prototypes | 8 (2 per class) |
| initialization | GLVQ: randomly chosen data |
| | Γ-GLVQ: GLVQ model |
| distance function | $d_E^2$ |
| learning rates | $\alpha_W = 0.01$ |
| transfer function parameter | $\theta(\iota_{start}) = 1, \theta(\iota_{end}) = 15$ |

**experimental settings**

| | |
|---|---|
| number of CV folds | 8 |
| epochs | 1000 |

**descriptions** To differentiate the single classes (disease stages) of Wilson Disease (WD) is very difficult (see Sec. 1.2). Even to detect WD if the patient is in the non-neurological phase without detailed and costly medical investigation is challenging. The electro-physiological investigation is comparable cheap. Yet, an early detection of WD is mandatory to prevent the fast degeneration by pharmaceutical treatment. Further, the medical treatment depends on the phase (NN, N) and clear distinction between the classes is desired. These medical expert knowledge can be roughly modeled by the AMAM

$$\Gamma = \begin{pmatrix} 1 & 1 & 5 & 5 \\ 1 & 1 & 5 & 5 \\ 5 & 5 & 1 & 1 \\ 10 & 10 & 10 & 1 \end{pmatrix}.$$

**results in numbers**

| | | real | | | |
|---|---|---|---|---|---|
| | GLVQ | PS | PP+MT | NN | V |
| predicted | PS | 53.10 % | 12.50 % | 0 % | 0 % |
| | PP+MT | 34.40 % | 31.25 % | 6.25 % | 6.25 % |
| | NN | 6.25 % | 12.50 % | 37.50 % | 2.08 % |
| | V | 6.25 % | 43.8 % | 56.25 % | 91.67 % |

| | | real | | | |
|---|---|---|---|---|---|
| | Γ-GLVQ | PS | PP+MT | NN | V |
| predicted | PS | 50.00 % | 12.50 % | 0 % | 0 % |
| | PP+MT | 31.25 % | 25.00 % | 12.50 % | 6.25 % |
| | NN | 15.63 % | 12.50 % | 37.50 % | 2.08 % |
| | V | 3.13 % | 25.00 % | 31.25 % | 64.58 % |

**resume** Considering the confusion matrix, it becomes clear that with the $\Gamma-$GLVQ we get an improvement of the class distinction according to the desired diagnostic behavior. Particularly, the false-classification of NN-persons to the class V is drastically reduced although false-classifications still remain. This result is also reflected in the averaged weighted error: for the GLVQ we obtain $werr = 38.3$ whereas for the $\Gamma-$GLVQ it is dropped down to $werr = 33.3$. Thus, applying the asymmetric GLVQ we are able to reflect medical expert knowledge seriously.

### 5.2.2 Optimizing Statistical Validation Measurements for Binary Class Problems in the GLVQ

In the last section we have shown up possibilities to integrate auxiliary information about the importance of classes or costs of misclassifications. Thereby, we used the fact of approximation of the classification error by the GLVQ cost function applying the sigmoid transfer function $f_\theta$ (2.38) (see Sec. 5.1). Further, in chapter 3 we pointed out that beside the classification accuracy/error other statistical evaluation measures exist. Especially for 2-class problems with the positive and negative classes ($\mathcal{C} = \{\oplus, \ominus\}$), a lot of such measures based on the confusion matrix are used for evaluation. Thus, it seems reasonable to modify the GLVQ optimizing such evaluation measures.

For this propose, an expression fitting to the GLVQ-principle of the single entries of the confusion matrix is needed. The transfer function of the classifier function is assumed to approximate the Heaviside function, i. e. $f_\theta(\mu_W(\mathbf{v})) \approx 1$ if $\mathbf{v}$ would be misclassified by the classifier, i. e.$d^-(\mathbf{v}) < d^+(\mathbf{v})$. Analogously, $\hat{f}_\theta(\mu_W(\mathbf{v})) = f_\theta(-\mu_W(\mathbf{v})) \approx$

1 applies if $\mathbf{v}$ is correct classified. Obviously, the cost function of the GLVQ utilizes $\hat{f}_\theta(\mu_W(\mathbf{v}))$ to count the number of correct classified data points for high $\theta$.

Thus, we can approximately express all quantities of the confusion matrix in term of $\hat{f}_\theta(\mu_W(\mathbf{v}))$:

$$\text{true positives:} \quad tp_W(V) \approx \sum_{v \in V} \delta_{\oplus,c(\mathbf{v})} \cdot \hat{f}_\theta(\mu_W(\mathbf{v})) \tag{5.14}$$

$$\text{false positives:} \quad fp_W(V) \approx \sum_{v \in V} \delta_{\ominus,c(\mathbf{v})} \cdot \left(1 - \hat{f}_\theta(\mu_W(\mathbf{v}))\right) \tag{5.15}$$

$$\text{true negative:} \quad tn_W(V) \approx \sum_{v \in V} \delta_{\oplus,c(\mathbf{v})} \cdot \left(1 - \hat{f}_\theta(\mu_W(\mathbf{v}))\right) \tag{5.16}$$

$$\text{false negatives:} \quad fn_W(V) \approx \sum_{v \in V} \delta_{\ominus,c(\mathbf{v})} \cdot \hat{f}_\theta(\mu_W(\mathbf{v})) \tag{5.17}$$

with the Kronecker function $\delta_{\oplus,c(\mathbf{v})}$ and $\delta_{\ominus,c(\mathbf{v})}$. Exemplary, the true positive value $tp_W(V)$ approximates the number of correct classified data points $\mathbf{v}$ of the positive class $c(\mathbf{v}) = \oplus$. Apparently, the functions (5.14)-(5.17) are differentiable with respect to $\mu_W(\mathbf{v})$ and, therefore, with respect to the prototypes $\mathbf{w}$:

$$\frac{\partial tp_W(\mathbf{v})}{\partial \mathbf{w}} = \delta_{\oplus,c(\mathbf{v})} \cdot \frac{\partial \hat{f}_\theta(\mu_W(\mathbf{v}))}{\partial \mathbf{w}} \tag{5.18}$$

$$\frac{\partial fp_W(\mathbf{v})}{\partial \mathbf{w}} = -(1 - \delta_{\oplus,c(\mathbf{v})}) \cdot \frac{\partial \hat{f}_\theta(\mu_W(\mathbf{v}))}{\partial \mathbf{w}} \tag{5.19}$$

$$\frac{\partial fn_W(\mathbf{v})}{\partial \mathbf{w}} = -\delta_{\oplus,c(\mathbf{v})} \cdot \frac{\partial \hat{f}_\theta(\mu_W(\mathbf{v}))}{\partial \mathbf{w}} \tag{5.20}$$

$$\frac{\partial tn_W(\mathbf{v})}{\partial \mathbf{w}} = (1 - \delta_{\oplus,c(\mathbf{v})}) \cdot \frac{\partial \hat{f}_\theta(\mu_W(\mathbf{v}))}{\partial \mathbf{w}} \tag{5.21}$$

In the following, we suppose a general statistical evaluation measure $S\left(tp_W(V), fp_W(V), tn_W(V), fp_W(V)\right)$ based on the confusion matrix, which has to be optimized. The only conditions on the measure $S$ is the continuity and differentiability with respect to $tp_W(V), fp_W(V), tn_W(V)$ and $fp_W(V)$.

The basic precision

$$\pi(tp_W(V), fp_W(V)) = \frac{tp_W(V)}{tp_W(V) + fp_W(V)} \tag{5.22}$$

or recall

$$\rho(tp_W(V), fn_W(V)) = \frac{tp_W(V)}{tp_W(V) + fn_W(V)} \tag{5.23}$$

as well as the established F-measure (3.2)

$$F_{\beta^2}\left(tp_W(V), fn_W(V), fp_W(V)\right) = \frac{(1+\beta^2)\cdot tp_W(V)}{(1+\beta^2)\cdot tp_W(V) + \beta^2 \cdot fn_W(V) + fp_W(V)}$$

are examples for such a measure function $S$. The respective derivatives are:

**precsion**

$$\frac{\partial \pi}{\partial \hat{f}_\theta\left(\mu_W(\mathbf{v})\right)} = \frac{\delta_{\oplus,c(\mathbf{v})} fp_W(\mathbf{v}) + (1-\delta_{\oplus,c(\mathbf{v})}) tp_W(\mathbf{v})}{\left(tp_W(\mathbf{v}) + fp_W(\mathbf{v})\right)^2}$$

**recall**

$$\frac{\partial \rho}{\partial \hat{f}_\theta\left(c(\mathbf{v})\right)} = \frac{\delta_{\oplus,c(\mathbf{v})} fp_W(\mathbf{v}) + \delta_{\oplus,c(\mathbf{v})} tn_W(\mathbf{v})}{\left(tp_W(\mathbf{v}) + fn_W(\mathbf{v})\right)^2}$$

**F-measure**

$$\frac{\partial F_{\beta^2}}{\partial \hat{f}_\theta\left(\mu_W(\mathbf{v})\right)} = \frac{(1+\beta^2)\left(\delta_{\oplus,c(\mathbf{v})}\left(\beta^2 fn_W + fp_W + (\beta^2-1)tp_W\right) + tp_W\right)}{\left((1+\beta^2)tp_W + \beta^2 fn_W + fp_W\right)^2}$$

Hence, the principle of the learning methodology of standard GLVQ stay the same. Only the underlaying cost function is replaced by an arbitrary differentiable stochastic evaluation measure $S\left(tp_W(V), fp_W(V), tn_W(V), fp_W(V)\right)$. This offers new fields of application for the GLVQ, particularly, in medicine.

Other proper measures would be the modified accuracy or G-measure (see Section 3.3). Yet, it has to be emphasized that the GLVQ only approximates the confusion matrix entries. Thus, if the statistical measurement is complex like the Matthews correlation coefficient (3.1), the modified GLVQ may become instable.

Of course, the idea of metric adaption or replacing of the Euclidean distance by an other differential dissimilarity is straight forward. Exemplary, we optimize different $F_{\beta^2}$-measures for the medicine data set Morbus Wilson in Application 5.5.

**Application 5.5** (Optimizing Statistical Measures Using the GLVQ Principle)**.**

**data set** Morbus Wilson (PET2)

**methods** $F_{\beta^2}$-GLVQ

**parameter settings**

| | |
|---|---|
| number of prototypes | 2 (1 per class) |
| distance function | $d_E^2$ |
| learning rates | $\alpha_W = 0.01$ |
| transfer function parameter | $\theta(\iota_{start}) = 1$,$\theta(\iota_{end}) = 15$ |

**experimental settings**

| | |
|---|---|
| number of CV-folds | 8 |
| epochs | 1000 |

**results in numbers** confusion matrices for the test data

| | GLVQ | | $F_{0.5}$-GLVQ | | $F_1$-GLVQ | | $F_2$-GLVQ | |
|---|---|---|---|---|---|---|---|---|
| | true | | true | | true | | true | |
| conf mat. | N | NN | N | NN | N | NN | N | NN |
| N | 90.9 % | 28.0 % | 88.5 % | 7.2 % | 93.6 % | 11.1 % | 96.7 % | 15.4 % |
| NN | 9.1 % | 72.0 % | 11.5 % | 92.8 % | 6.4 % | 88.9 % | 3.2 % | 84.6 % |
| $F_{0.5}$ | 0.790 | | **0.907** | | 90.1 | | 88.7 | |
| $F_1$ | 0.816 | | 0.902 | | **0.910** | | 0.906 | |
| $F_2$ | 0.845 | | 0.896 | | 0.918 | | **0.926** | |
| precision | 0.741 | | **0.918** | | 0.885 | | 0.852 | |
| recall | 0.909 | | 0.885 | | 0.936 | | **0.968** | |

**resume** Looking at the confusion matrices, we observe the best $F_{\beta^2}$ value for the GLVQ-model optimizing the according $F_{\beta^2}$-measure. Small values for $\beta$ give the precision an higher influence on the $F_{\beta^2}$-measure, such the best precision are obtained by the $F_{0.5}$-GLVQ model. Further, the best recall value is achieved for the $F_2$-GLVQ model. Otherwise, the standard GLVQ is never yielded the best $F_{\beta^2}$-measure, as expected. Thus, this exemplary experiment show the ability of the modified GLVQ to optimizes statistical evaluations measurements based on the confusion matrix.

**This section is based on**

*M. Kästner, W. Hermann, and T. Villmann: Integration of Structural Expert Knowledge about Classes for Classification Using the Fuzzy Supervised Neural Gas. Proc. of ESANN, 2012.*

## 5.3 Integration of Structural Knowledge about the Labeling in Fuzzy Supervised Neural Gas

In several applications of classification problems the labeling of each data points can be very costly or sometimes it is subjected to uncertainty, i. e. the labeling need a lot of expertise. Let us assume an example: Given is a medical classification problem. The patients are divided into five stages in breast cancer: healthy or without symptoms, pre-cancerous/ non-invasive (level 0), invasive with different sizes and state of spreading to axillary lymph nodes (levels 1-3), and invasive breast cancer that has spread beyond the breast and nearby lymph nodes to other organs of the body (level 4) [Edge et al., 2010]. Especially, the assignment of a patient to level 1-3 is very challenging and not always unambiguous. Otherwise, the distinction between level 0 and the other ones is very clear. In general, such uncertainty in the labeling should be integrated in the training of a classifier model to represent the given problem in a better way. In this section, we figured out a possibility of a respective expert knowledge integration in the FSNG model (see Sec. 4.2.2), which is transferable to the FSSOM (see Sec. 4.2.1).

As known from Section 2.3, the FSNG is a semi-supervised method which can take into a account fuzzy labels. The above described uncertainty in label can be reflect by the help of fuzzy assignments. In the FSNG model, beside the data point distance a dissimilarity between the labels are taken into account during the learning. Thus, the integration of expert knowledge by modifying the dissimilarities between the labeling $\delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}))$ is obviously.

One simple strategy is to define $\delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}))$ as a quadratic form

$$\delta_{\mathbf{K}}(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w})) = (\mathbf{c}(\mathbf{v}) - \mathbf{y}(\mathbf{w}))^T \ \mathbf{K} \ (\mathbf{c}(\mathbf{v}) - \mathbf{y}(\mathbf{w}))^T \tag{5.24}$$

with a positive definite expert knowledge matrix $\mathbf{K}$. Thus, $\delta_{\mathbf{K}}(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}))$ is a positive definite belinear form. Another choice is a diagonal matrix $\mathbf{K}$, where the $K_{ii} > 0$ imply the class weighting related to their importance. This is in analogy to the weighted Euclidean distance for the class weighting vector $\beta$ in the GLVQ (see page 137).

To integrate the uncertainty of the labeling between specific classes, we consider the conditional probability $p(l|k) \geq 0$ that a data vector belonging to class $k$ could be assigned to class $l$. In the following we denote such a misclassification as failure event (fe). These information are collected in an uncertainty matrix $U \in [0; 1]^{N_C \times N_C}$ with the

entries $U_{k,l} = p(l|k)$. It is essential that $\sum_{l=1}^{N_C} U_{k,l} = \sum_{l=1}^{N_C} p(l|k) = 1$. Further, for probabilistic interpretation we suppose that $p(l|l) > p(l|k)\ \forall k \neq l$. Yet, $p(l,l)$ can be less than 1 which reflects the uncertainty in assignments of class $l$. Under this assumption the matrix U has non-vanishing off-diagonal elements, a failure event should be less contribute to an error criterion. For this purpose, we introduce the *unification dissimilarity* for data and prototype label vectors $\mathbf{c}(\mathbf{v})$ and $\mathbf{y}(\mathbf{w})$

$$D_{k,l} = \left( \frac{[\mathbf{c}(\mathbf{v})]_k + [\mathbf{c}(\mathbf{v})]_l}{2} - \frac{[\mathbf{y}(\mathbf{w})]_k + [\mathbf{y}(\mathbf{w})]_l}{2} \right)^2 \tag{5.25}$$

with respect to the classes $k, l \in \mathcal{C}$. Therefore, we obtain the *unification dissimilarity matrix* $\mathbf{D}$, where the entries $D_{k,l}$ describes the deviation of class vector entries if the classes $k$ and $l$ would be merged.

Both aspects, uncertainty and unification distance are combined in the class dissimilarity measure

$$\delta_{\mathbf{U}}(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w})) = Fr(\mathbf{U} \circ \mathbf{D}) \tag{5.26}$$

where $\mathbf{A} \circ \mathbf{B}$ is the Hadamard product between matrix $A$ and $B$ and $Fr(\mathbf{A})$ denotes the Frobenius norm of matrix $\mathbf{A}$. Generally, $\delta_{\mathbf{U}}(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}))$ is not a metric but still a dissimilarity measure (see Note 2.1).

We emphasize at this point that the uncertainty matrix does not explicitly code the classification goal like in the section before. An application on the proposed approach is presented in Application 5.6.

In this section, we gave a possibility how to integrate auxiliary knowledge about the labeling into the FSNG by use of an uncertainty matrix. Obviously, other possibility might be considered and has to be verified for specific problems.

**Application 5.6** (Integrate Uncertainty of Labeling in FSNG)**.**

**data set** Morbus Wilson (6cl)

**methods** $U$-FSNG

**parameter settings**

| | |
|---|---|
| number of prototypes | 11 |
| distance function | $d_E^2(\mathbf{v}, \mathbf{w})$, |
| | $\delta_\beta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}))$, |
| | $\delta_{\mathbf{U}}(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}))$ |
| learning rates | $\alpha_W = 0.01$ |

**experimental settings**

| | |
|---|---|
| training/test | training and test on whole data set |
| epochs | 2000 |
| initialization | random data point |

**descriptions**

The main task for the Wilson disease (WD) classification is to distinguish between neurological and non-neurological case (see Sec. 1.2). A subdivision between these main class is very difficult and it is not clear whether a precise classification is based on the EIP data set is possible [Hermann et al., 2003].

A first possibility is to weight the single WD-subtypes, such that a better detection of the volunteer group is desirable to prevent a gratuitous treatment. According to medical expertise, we used a class weighting vector $\beta$ (5.27).

The structural medical expert knowledge is available (as common sense) for failure events in clinical classification of patients. It is fed into the uncertainty matrix **U** (5.27) in this way that the non-diagonal elements in each row (diagnosis) of **U** describe the probability for detecting the respective diagnosis (column) instead. Thus, the entries $u_{ij}$ of the uncertainty matrix **U** reflect the common conditional probability of a $fe$ assigning class $i$ to the (true) class $j$. For example: the assignment of a health people is relatively reliable in contrast of the MT class.

$$\beta = \begin{pmatrix} 2 \\ 2 \\ 2 \\ 1 \\ 1 \\ 10 \end{pmatrix} \tag{5.27}$$

$$\mathbf{U} = \begin{pmatrix} 0.6 & 0.15 & 0.25 & 0 & 0 & 0 \\ 0.1 & 0.6 & 0.3 & 0 & 0 & 0 \\ 0.25 & 0.25 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.8 & 0.15 & 0.05 \\ 0 & 0 & 0 & 0.15 & 0.60 & 0.25 \\ 0 & 0 & 0 & 0 & 0.05 & 0.95 \end{pmatrix} \tag{5.28}$$

**results in numbers** confusion matrix (absolute values)

| FSNG | | N | | | NN | | |
|---|---|---|---|---|---|---|---|
| | | PS | PP | MT | HT | AT | V |
| N | PS | **16** | 8 | 6 | 2 | 2 | 0 |
| | PP | 0 | **10** | 1 | 0 | 2 | 1 |
| | MT | 2 | 1 | **4** | 0 | 0 | 1 |
| NN | HT | 0 | 6 | 0 | **1** | 0 | 3 |
| | AT | 0 | 0 | 0 | 0 | **4** | 4 |
| | V | 0 | **15** | 1 | 0 | 10 | **22** |

| $\beta$-FSNG | | N | | | NN | | |
|---|---|---|---|---|---|---|---|
| | | PS | PP | MT | HT | AT | V |
| N | PS | **17** | 2 | 12 | 0 | 2 | 1 |
| | PP | 0 | **11** | 1 | 0 | 2 | 1 |
| | MT | 0 | 1 | **6** | 0 | 0 | 1 |
| NN | HT | 0 | 6 | 0 | **1** | 0 | 3 |
| | AT | 0 | 0 | 0 | 0 | **4** | 4 |
| | V | 0 | **14** | 1 | 0 | 8 | **25** |

| **U**-FSNG | | N | | | NN | | |
|---|---|---|---|---|---|---|---|
| | | PS | PP | MT | HT | AT | V |
| N | PS | **16** | 4 | 11 | 1 | 2 | 0 |
| | PP | 0 | **1** | 1 | 5 | 2 | 5 |
| | MT | 0 | 1 | **6** | 1 | 0 | 1 |
| NN | HT | 0 | 0 | 0 | **4** | 0 | 5 |
| | AT | 1 | 0 | 0 | 0 | **4** | 4 |
| | V | 0 | **0** | 1 | 1 | 10 | **36** |

**resume** In the standard FSNG, a high misclassification number of the volunteers (V) to the neurological group especially to the pseudo-parkinsonian (PP) class occur. The integration of expert knowledge reduces these effects: In application the weighting $\beta$-FSNG achieved only a small improvement. Yet, the integration of extended expert knowledge about the uncertainty of medical doctors classification leads to a substantial improvement. In particular, the volunteer group is clearly separated from the neurological class. A remaining violation is due to the asymptomatic subtype. This effect is in agreement with clinical findings, because the AT-group usually show very weak symptoms [Hermann et al., 2002]. Furthermore, the separation between the neurological and the non-neurological types is also improved. Yet, the confusions within these groups are not solved reflecting the struc-

tural expert knowledge.
To sum up, a classification of Wilson's disease types based on neurophysiology measurements is possible if structural expert knowledge is additional used for the task specific classification scheme. Without this expert information a precise classification is difficult at least.

# Chapter 6

# Conclusion and Future Work

In this thesis, we had a closer look at the problem of classification in context of prototype based vector quantization and the respective model learning. The motivation of this work was to consider problems around evaluation of classifiers beyond the simple accuracy and their incorporation into the model description. Thereby, one of the key aspects was the integration of expert knowledge to obtain problem specific models. In this view, classification can be regarded as an *ill-posed* problem as pointed out in Section 3.4.

The aspects dealt with in this thesis, can be subdivided into the following main topics:

1. misclassification costs beyond accuracy
2. integration of structural information about data into model learning exemplified by processing functional data
3. semi-supervised learning and integration of uncertainty in labeling

Of course, these topics are neither independent nor complete. Usually they interact and are additionally influenced by further aspects which, however, may be unknown for the user. In this sense this thesis is a contribution for those classification tasks, where explicit knowledge about the data, classification costs or structural information is available.

One of the most powerful and also intuitive prototype based classifiers is the LVQ as introduced by Kohonen. We restricted us to the cost variant Generalized Learning Vector Quantization. In particular we focused on GLVQ to consider other evaluation statistical classification measures beside accuracy and integrated knowledge about classification weights or misclassification costs. One essential outcome of the thesis is the interpretation of the GLVQ cost function as an approximation of counting misclassifications: the border sensitive GLVQ (see Sec. 5.1). This perception allows us to modify the GLVQ such that the direct optimization of statistical evaluation measures based on the confusion matrix by keeping the principle of GLVQ becomes possible. This idea was demonstrated with the $F_\beta$-GLVQ, where the cost function approximated the F-measure (see Sec. 5.2.2). Another option is the modification of the GLVQ cost function to regard asymmetric misclassification costs like they often occur in medical environments (see Sec. 5.2.1).

Another topic of this thesis was the integration of structural information about the data into the relevance learning scheme of GLVQ. We exemplified these ideas for functional data leading to the functional relevance learning (GFR/MLVQ, see Sec. 4.1.1) and the enhanced GR/MLVQ (eGR/MLVQ, see Sec. 4.1.2). Both algorithms integrate the lateral dependencies of the neighboring dimensions of functional vector data into the learning scheme. In the GFR/MLVQ instead of learning each dimension independently, the relevance profile is composed by a linear combination of non-linear and smooth basis functions and the number of parameters to adapt was decreased significantly. Yet, in the experiments the GFR/MLVQ shows difficulties learning these parameters. In contrast to GFR/MLVQ, in eGR/MLVQ the number of free parameters was indirectly reduced by adding a neighborhood function for relevance learning. The eGR/MLVQ leads to more stable results compared to the GR/MLVQ and experiments show a speed up in learning. Further, the eGR/MLVQ can be applied also to other kinds of data with structural information about the dependencies in the data.

The third topic was the integration of label information in unsupervised vector quantization principles to obtain semi-supervised methods. Therefore, the distance measures in Neural Gas and Self Organizing Maps are extended to incorporate label information in a multiplicative manner (FSSOM/FSNG, see Sec. 4.2.1 and 4.2.2). The resulting FSNG/FSSOM can handle labeled as well as unlabeled data where the label can be either crisp or fuzzy. The obtained models including of prototypes with fuzzy labels provide a broad range of applications. The FSNG/FSSOM model can be extened to handle also uncertainty in data labeling as explained in Section 5.3.

Summarizing all topics, as mentioned above, the considered model extensions and modifications of vector quantizers are neither comprehensive nor covering all aspects of classification learning. Although many real classification tasks and applications of vector quantization can be tackled in this manner, there are still opportunities for further improvements and perspectives. For example, we could think about other evaluation measures like the AUROC-measure as optimization objective for GLVQ. Another possibility would be to identify outliers or doubtful data samples in classification and to sensitize the GLVQ model for those data.

Further, so far only binary classification evaluation measures were covered in this thesis, when alternative misclassification costs were considered. There also exist generalizations of statistical measures to evaluate models for multi-class problems. An extension of the GLVQ for such problems might be feasible and desired.

Although this thesis concentrates on the standard GLVQ, a transfer to Relational or Median variants can be considered.

# My Publications

List of my publications which are related to my thesis.

## Journals

1. T. Villmann, S. Haase, M. Kaden: Kernelized Vector Quantization in Gradient-Descent Learning. Neurocomputing 147, p. 83-95, 2015

2. M. Kaden, M. Riedel, W. Hermann, T. Villmann: Border-Sensitive Learning in Generalized Learning Vector Quantization: an Alternative to Support Vector Machines. Soft Computing, 19 (9), p. 2423–2434, 2015

3. M. Kaden, M. Lange, D. Nebel, M. Riedel, T. Geweniger and T. Villmann: Aspects in Classification Learning - Review of Recent Developments in Learning Vector Quantization. Foundation of Computing and Decisions Sciences, 39 (2), 2014.

4. T. Villmann, M. Kaden, D. Nebel, M. Riedel: Lateral Enhancement in Adaptive Metric Learning for Functional Data. Neurocomputing 131, p. 23-31, 2014.

5. T. Geweniger, L. Fischer, M. Kaden, M. Lange, T. Villmann: Clustering by Fuzzy Neural Gas and Evaluation of Fuzzy Clusters. Computational Intelligence and Neuroscience, 2013.

6. M. Kästner, B. Hammer, M. Biehl, T. Villmann: Functional Relevance Learning in Generalized Learning Vector Quantization. Neurocomputing, 90, p. 85–95, 2012.

## Conferences

1. M. Kaden, W. Hermann, T. Villmann: Attention Based Classification Learning in GLVQ and Asymmetric Misclassification Assessment. Advances in Self-

Organizing Maps: 9th International Workshop, Springer-Verlag, p. 77-87, 2014.

2. M. Kaden, W. Hermann, T. Villmann: Optimization of General Statistical Accuracy Measures for Classification Based on Learning Vector Quantization. in M. Verleysen (Ed.), Proc. European Symposium on Artificial Neural Networks, 2014.

3. T. Geweniger, M. Kästner, T. Villmann: Border sensitive fuzzy vector quantization in semi-supervised learning. in M. Verleysen (Ed.), Proc. European Symposium on Artificial Neural Networks, p. 509-514, 2013.

4. M. Kästner, M. Strickert, T. Villmann: A sparse kernelized matrix learning vector quantization model for human activity recognition. in M. Verleysen (Ed.), Proc. European Symposium on Artificial Neural Networks, p. 449-454, 2013.

5. M. Riedel, F. Rossi, M. Kästner, T. Villmann: Regularization in relevance learning vector quantization using l1-norms. in M. Verleysen (Ed.), Proc. European Symposium on Artificial Neural Networks, p. 17-22, 2013.

6. T. Villmann, M. Kästner, A. Backhaus, U. Seiffert: in M. Verleysen (Ed.), Proc. European Symposium on Artificial Neural Networks, 2013.

7. M. Kästner, M. Riedel, M. Stickert, W. Hermann and T. Villmann: Border-Sensitive Learning in Kernelized Learning Vector Quantization. International Work Conference on Artificial Neural Networks, 2013.

8. T. Villmann, S. Haase, M. Kästner: Gradient Based Learning in Vector Quantization Using Differentiable Kernels. in P.A. Estévez, J.C. Príncipe, P. Zegers (Eds.), Advances in Self-Organizing Maps: 9th International Workshop, Springer-Verlag, p. 193–204 , 2013.

9. M. Kästner, M. Lange, T. Villmann: Fuzzy Supervised Self-Organizing Map for Semi-Supervised Vector Quantization. in L. Rutkowski, M. Korytkowski, R. Scherer, R. Tadeusiewicz, L. Zadeh, J. Zurada (Eds.), Artificial Intelligence and Soft Computing - Proc. of the International Conference ICAISC, Springer, p. 256–265, 2012.

10. M. Kästner, W. Hermann, T. Villmann: Integration of Structural Expert Knowledge about Classes for Classification Using the Fuzzy Supervised Neural Gas. in M. Verleysen (Ed.), Proc. European Symposium on Artificial Neural Networks (ESANN), 209–214, 2012.

11. M. Kästner, D. Nebel, M. Riedel, M. Biehl, T. Villmann: Differentiable Kernels in Generalized Matrix Learning Vector Quantization. Proc. of the 11th Annual

International Conference on Machine Learning and Applications. IEEE Press, 14–19, 2012.

12. T. Villmannn, M. Kästner, D. Nebel, M. Riedel: ICMLA Face Recognition Challenge – Results of the Team Computational Intelligence Mittweida. Proc. of 2012 11th Annual International Conference on Machine Learng and Applications. IEEE Press, 40–46, 2012.

13. M. Kästner, A. Backhaus, T. Geweniger, S. Haase, U. Seiffert, T. Villmann: Relevance Learning in Unsupervised Vector Quantization based on Divergences. in J. Laaksonen and T. Honkela (eds.), Advances in Self-Organizing Maps - Proceeding of the Workshop on Self-Organizing Maps (WSOM'2011). Springer LNCS, 90–100, 2011.

14. T. Villmann, M. Kästner: Sparse Functional Relevance Learning in Generalized Learning Vector Quantization. in J. Laaksonen and T. Honkela (eds.), Advances in Self-Organizing Maps - Proceeding of the Workshop on Self-Organizing Maps. Springer LNCS, 79–89, 2011.

15. T. Geweniger, M. Kästner, T.Villmann: Optimization of Parametrized Divergences in Fuzzy-c-means. Proc. European Symposium on Artificial Neural Networks, M. Verleysen (ed), Brügge, 11–16, 2011.

16. M. Kästner, B. Hammer, M. Biehl, T. Villmann: Generalized functional relevance learning. Proc. European Symposium on Artificial Neural Networks, M. Verleysen (ed), Brügge, 93–98, 2011.

17. M. Kästner, T. Villmann: Functional relevance learning in learning vector quantization for hyperspectral data. Proc. 3rd Workshop on Hyperspectral Image and Signal Processing: Evolution Remote Sensing, 2011.

## Further Articles and Technical Reports

1. T. Villmann, M. Kästner, D. Nebel, M. Riedel: Enhancement Learning in Functional Relevance Learning Vector Quantization. *Machine Learning Reports* 6 (MLR-3-2012), p. 46–57, 2012.

2. M. Kästner, M. Riedel, M. Strickert, T. Villmann: Class Border Sensitive Generalized Learning Vector Quantization – An Alternative to Support Vector Machines. *Machine Learning Reports* 6 (MLR-6-2012),p. 40–56, 2012.

3. M. Kästner, M. Strickert, D. Labudde, M. Lange, S. Haase, T. Villmann: Utilization of Correlation Measures in Vector Quantization for Analysis of Gene Expression

Data – A Review of Recent Developments. *Machine Learning Reports* 6 (MLR-6-2012), p. 5–22, 2012.

4. T. Villmann, T. Geweniger, M. Kästner: Border Sensitive Fuzzy Classification Learning in Fuzzy Vector Quantization. *Machine Learning Reports* 6 (MLR-6-2012), p. 23–39, 2012.

5. M. Kästner, T. Villmann: Functional relevance learning in generalized learning vector quantization. *Machine Learning Reports* 5 (MLR-1-2011), p. 81–89, 2011.

6. M. Kästner, T. Villmann: Fuzzy Supervised Neural Gas for Semi-supervised Vector Quantization – Theoretical Aspects. *Machine Learning Reports* 5 (MLR-2-2011), p. 1–16, 2011.

7. M. Kästner, T. Villmann, M. Biehl: About Sparsity in Functional Relevance Learning in Generalized Learning Vector Quantization. *Machine Learning Reports* 5 (MLR-3-20101), p. 1–12, 2011.

8. T. Villmann, M. Kästner: Fuzzy Supervised Neural Gas with Sparsity Constraints. *Machine Learning Reports* 5 (MLR-5-2011), p. 17–20, 2011.

9. T. Villmann, S. Haase, S. Simmuteit, M. Kästner, F.-M. Schleif: Functional Vector Quantization Based on Divergence Learning. *Ulmer Informatik-Berichte* , p. 8–11, 2010.

# Appendix A

# Appendix

## A.1 Stochastic Gradient Descent (SGD)

The stochastic gradient descent (SGD) is a suitable method for minimizing non-linear and non-convex optimization problems with the real valued cost function $E : \mathbb{R}^D \rightarrow \mathbb{R}$. In this thesis, we concentrate on minimizing of vector quantization cost functions $E(V, W, \Phi)$ based on single example presentation $\mathbf{v} \in V$ with the prototypes $W$ and further parameters $\Phi$. The principle sequence is specified in Algorithm 1.

This SGD-principle is easy to implement and flexible for every kind of cost function. The only assumption is the differentiability with respect to the parameters. Further, several parameters can be optimized simultaneously like the prototypes $W$ and the distance based parameters $\Phi$ (see Section 2.4).

Yet, this great flexibility causes challenges, too. The convergence of the SGD to the global optima are guaranteed under some requirements. However, two requirements are the adiabatic decrease of the learning rates and the infinity time of learning [Kushner and Clark, 1978]. Unfortunately, this is not practicable. Thus, in application the SGD ends up with a local optima and the handling of the learning of the parameters is challenging especially if more than one parameter is adapted simultaneously.

Nevertheless, the SGD is an established optimization tool in machine learning particularly in prototype based vector quantization.

**Algorithm 1** Stochastic Gradient Descent principle

1: **procedure** SGD($V$)
2: Initialize the parameters $W$ and $\Phi$
3: Choose a data point $\mathbf{v} \in V$ randomly
4: Determine all prototypes $\mathbf{w}_k \in W$ and further parameters $\phi_l \in \Phi$ which are responsible for $\mathbf{v}$
5: Calculate the partial derivatives according to the parameter $\mathbf{w}_k,\ \phi_l$:

$$\Delta\mathbf{w}_k = \frac{\partial E(\mathbf{v}, W, \Phi)}{\partial \mathbf{w}_k} \quad \Delta\phi_l = \frac{\partial E(\mathbf{v}, W, \Phi)}{\partial \phi_l} \quad \forall k, l$$

6: Update the parameters with a suitable step size/learning rate $\alpha_{\mathbf{w}}(\iota),\ \alpha_{\Phi}(\iota)$:

$$\mathbf{w}_k \leftarrow \mathbf{w}_k - \alpha_{\mathbf{w}}(\iota) \cdot \Delta\mathbf{w}_k$$
$$\phi_l \leftarrow \phi_l - \alpha_{\Phi}(\iota) \cdot \Delta\phi_l$$

7: Decrease the learning rates with a decreasing factor $\epsilon > 0$:

$$\alpha_{\mathbf{w}}(\iota + 1) = (1 - \epsilon)\alpha_{\mathbf{w}}(\iota) \qquad \alpha_{\Phi}(\iota + 1) = (1 - \epsilon)\alpha_{\Phi}(\iota)$$

8: Repeat steps 1-5 until convergence or manual stop
9: **end procedure**

## A.2 Support Vector Machine

This section gives a short review of basic ideas of the Support Vector Machine (SVM,[Schölkopf and Smola, 2002]). The SVM developed by `Vapnik` based on the statistical learning theory and the Vapnik-Chervonenkis dimension (VC-dimension) [Vapnik, 1995]. One goal in classification is to obtain a good generalization of a learned model, i. e. simply stated a model should not only perform well on training data but also on new data, the test data. The statistical learning theory deals among others with such a generalization ability of a model.

The idea of the SVM is to optimizes the separation margin between two classes. If a binary and linear separable classification problem with $\mathcal{C} = \{-1, +1\}$ is given the following convex optimization problem with constraints has to been solved:

$$\min_{\mathbf{n}\in\mathbb{R}^{\mathbf{D}}, b\in\mathbb{R}} \frac{1}{2} \langle \mathbf{n}, \mathbf{n} \rangle_E \tag{A.1}$$

$$\text{subject to} \quad c(\mathbf{v}) \cdot (\langle \mathbf{n}, \mathbf{v} \rangle_E + b) - 1 \geq 0 \qquad \forall \mathbf{v} \in V, \tag{A.2}$$

whereby $\mathbf{n} \in \mathbb{R}^D$, $b \in \mathbb{R}$ called weight vector and bias, respectively. The term $\langle \cdot, \cdot \rangle_E$ is the inner product in $\mathbb{R}^D$. Thereby $(\langle \mathbf{n}, \mathbf{v} \rangle_E + b)$ describes the hyperplane and can interpret as classifier function whereby

$$\text{sgn}(\langle \mathbf{n}, \mathbf{v} \rangle + b) = \begin{cases} > 0, \mathbf{v} \text{ belongs to } -1 \\ < 0, \mathbf{v} \text{ belongs to } +1 \\ = 0, \mathbf{v} \text{ layes on the decision hyperplane} \end{cases} \tag{A.3}$$

and $\text{sgn}(x)$ returns the sign of $x$.

The convex optimization problem can be transfered into the *Wolfe* dual problem which ends up in:

$$\max_{\alpha} \sum_{i=1}^{N_V} \alpha_i - \sum_{i=1}^{N_V} \sum_{j=1}^{N_V} c(\mathbf{v}_i) c(\mathbf{v}_j) \alpha_i \alpha_j \langle \mathbf{v}_i, \mathbf{v}_j \rangle_E \tag{A.4}$$

$$\text{subject to} \quad \sum_{i=1}^{N_V} c(\mathbf{v}_i) \alpha_i = 0 \tag{A.5}$$

$$\alpha_i \geq 0 \qquad \forall i = 1, \ldots, N_V \tag{A.6}$$

This optimization problem depends only on the Lagrange vector $\alpha$. However the hy-

perplane parameters can be derived from $\alpha$:

$$\mathbf{n} = \sum_{i=1}^{N_V} \alpha_i \cdot c(\mathbf{v}_i) \cdot \mathbf{v}_i \tag{A.7}$$

$$b = \frac{1}{N_V} \sum_{j=1}^{N_V} \left( c(\mathbf{v}_j) - \sum_{i=1}^{N_V} \alpha_i \cdot c(\mathbf{v}_i) \cdot \langle \mathbf{v}_j, \mathbf{v}_i \rangle_E \right) \tag{A.8}$$

Therefore, the decision function is:

$$\hat{c}(\mathbf{v}) = \text{sgn} \left( \sum_{i=1}^{N_V} \alpha_i \cdot c(\mathbf{v}_i) \cdot \langle \mathbf{v}, \mathbf{v}_i \rangle_E + b \right) \tag{A.9}$$

Further, out of the Karush-Kuhn-Tucker condition it ensues that $\alpha_i > 0$ iff $\mathbf{v}_i$ is *Support Vector (SV)*, i.e. only the SV are needed for the calculation of the decision function. Thus, the model complexity in the test phase depends directly on the number SV. A detailed mathematical derivation can be found in [Schölkopf and Smola, 2002].

Until now, the optimization problem (A.1) or (A.4), respectively, can only solve linear separable problems. A first solution for overlapping classes is to introduce slack variables

$$\xi(\mathbf{v}) \geq 0 \ (\forall \mathbf{v} \in V) \tag{A.10}$$

which relax the constraints in (A.1) to

$$c(\mathbf{v}) \cdot (\langle \mathbf{n}, \mathbf{v} \rangle_E + b) - 1 + \xi(\mathbf{v}) \geq 0 \qquad \forall \mathbf{v} \in V \tag{A.11}$$

It ends up on a *Soft Margin Classifier* with the cost function:

$$\min_{\mathbf{n} \in \mathbb{R}^{\mathbf{D}}, b \in \mathbb{R}} \frac{1}{2} \langle \mathbf{n}, \mathbf{n} \rangle_E + C \sum_{i=1}^{N_V} \xi(\mathbf{v}_i) \tag{A.12}$$

and the constraints (A.10) and (A.11). The parameter $C$ determines the trade off between maximization of the margin and minimization of the training error. This parameter has to be defined in advanced carefully. Another possibility to realize a Soft Margin Classifier is the $\nu$-SVM described in [Schölkopf and Smola, 2002].

However, the optimization problem (A.12) still ends up with decision hyperplane. This is may be not an adequate solution for several applications. One of the most powerful concepts realized in SVM remains the idea of kernel mapping or also called *kernel trick*, which is described on page 2.4ff.

The SVM is a binary classifier. For multi-class problems exists only heuristic, e. g.

one-versus-all or one-versus-one [Bishop, 2006]. However, these heuristics are very effective and therefore the SVM can be applied also for multi-class problems. Due to the great success of the SVM in classification, a lot of variations and extensions are developed e. g. Relevance SVM [Tipping, 2001], $\nu$-SVM, or the SVM for regression [Schölkopf and Smola, 2002].

**Table A.1**: List of common kernels with their related distance measures

| kernel type | $\kappa(\mathbf{v}, \mathbf{w})$ | $d^2_\kappa(\mathbf{v}, \mathbf{w})$ | $\frac{\partial d^2_\kappa(\mathbf{v},\mathbf{w})}{\partial \mathbf{w}}$ |
|---|---|---|---|
| Radial Basis Function (RBF) | $e^{\frac{-d_E(\mathbf{v},\mathbf{w})}{2\varsigma^2}}$ | $2 - 2\ \kappa_{RBF}(\mathbf{v}, \mathbf{w})\ (\varsigma > 0)$ | $-\frac{2(\mathbf{v}-\mathbf{w})}{\varsigma^2}\kappa_{RBF}$ |
| Radial (RD) | $g(d(\mathbf{v}, \mathbf{w}))$ ($g : \mathbb{R}^+ \to \mathbb{R}^+$, $d$- distance) | $2 \cdot g(0) - 2\kappa_(RD)$ | $-2g'(d(\mathbf{v}, \mathbf{w}))\frac{\partial d(\mathbf{v},\mathbf{w})}{\partial \mathbf{w}}$ |
| Linear (LIN) | $\mathbf{v}^T\mathbf{w}$ | $\mathbf{v}^T\mathbf{v} - \mathbf{2v^Tw} + \mathbf{w^Tw}$ | $2\mathbf{w} - 2\mathbf{v}$ |
| Polynomial (POL) | $(\mathbf{v}^T\mathbf{w})^p\ (p > 0)$ | $(\mathbf{v}^T\mathbf{v})^d - 2(\mathbf{v}^T\mathbf{w})^p + (\mathbf{w}^T\mathbf{w})^p$ | $2p \cdot \mathbf{v}(\mathbf{v}^T\mathbf{w})^{p-1} + 2p \cdot \mathbf{w}(\mathbf{w}^T\mathbf{w})^{p-1}$ |
| Exponential (EXP) | $e^{\mathbf{v}^T\mathbf{w}}$ | $e^{\mathbf{v}^T\mathbf{v}} - 2e^{\mathbf{v}^T\mathbf{w}} + e^{\mathbf{w}^T\mathbf{w}}$ | $-2\mathbf{v}e^{\mathbf{v}^T\mathbf{w}} + 2\mathbf{w}e^{\mathbf{w}^T\mathbf{w}}$ |
| Sobolev $S$ | $\mathbf{v}^T\mathbf{w} + \sum_{l=1}^k \alpha_l {\mathbf{v}^{(l)}}^T\mathbf{w}^{(l)}$ ($\mathbf{v}^{(l)}, \mathbf{w}^{(l)}$ − derivatives of $l^{th}$ order) | $\kappa_{LIN} + \sum_{l=1}^k \alpha_l({\mathbf{v}^{(l)}}^T\mathbf{v}^{(l)} - 2{\mathbf{v}^{(\mathbf{l})}}^\mathbf{T}\mathbf{w}^{(l)} + {\mathbf{w}^{(l)}}^T\mathbf{w}^{(l)})$ | $2\mathbf{w} - 2\mathbf{v}$ |
| Student-type Gaussian (STG) | $\left(\beta + \frac{d^2_E(\mathbf{v},\mathbf{w})}{\varsigma^2}\right)^{-\alpha}\ (\alpha, \beta, \varsigma > 0)$ | $2\beta^{-\alpha} - 2 \cdot \kappa_{ST}(\mathbf{v}, \mathbf{w})$ | $\frac{-4\cdot\alpha}{\varsigma^2}\ (\mathbf{v} - \mathbf{w})\left(\beta + \frac{d^2_E(\mathbf{v},\mathbf{w})}{\varsigma^2}\right)^{-(\alpha+1)}$ |

**Parts of the section are based on**

- *M. Kästner, T. Villmann: Fuzzy Supervised Neural Gas for Semi-supervised Vector Quantization – Theoretical Aspects.* Machine Learning Reports **5** *(MLR-2-2011) 1–16, 2011.*

## A.3 Fuzzy Supervised Neural Gas Algorithm Solved by Stochastic Gradient Descent

In the following it is shown that the adaptation dynamic of prototypes (4.42) and labels (4.42) of Fuzzy Semi-Supervised Neural Gas algorithm (FSNG, see section 4.2.2) follows a stochastic gradient descent on the cost function given in (4.40). Thus it overcomes the difficulties in the convergence proof of Fuzzy Labeld Neural Gas (FLNG, see section 2.3.3), where we have to distinguish discrete and continuous data distributions.

To proof the convergence, we have a closer look to the derivatives of the cost function according with respect to the parameters [Martinetz et al., 1993]. On the one side the derivatives $\frac{\partial E_{\mathrm{FSNG}}}{\partial \mathbf{w}_j}$ according to the prototypes and on the other side the derivatives $\frac{\partial E_{\mathrm{FSNG}}}{\partial \mathbf{y}(\mathbf{w}_j)}$ according to the label assignment has do be considered.

First, we analyzes the prototype dynamic:

$$\frac{\partial E_{\mathrm{FSNG}}}{\partial \mathbf{w}_j} = R_j + \int P(\mathbf{v})\, h_\sigma^{NG}\left(rk_j(\mathbf{v}, \mathbf{w}_j)\right) \frac{\partial D_\varepsilon(\nu, \omega_j, \gamma)}{\partial \mathbf{w}_j} d\mathbf{v} \tag{A.13}$$

with the derivative

$$\frac{\partial D_\varepsilon(\nu, \omega_j, \gamma)}{\partial \mathbf{w}_j} = (\gamma \cdot \delta(\mathbf{c}(\mathbf{v}), \mathbf{y}(\mathbf{w}_j)) + \varepsilon_\delta) \cdot (1-\gamma) \cdot \frac{\partial d(\mathbf{v}, \mathbf{w}_j)}{\partial \mathbf{w}_j}.$$

The term $R_j$ is obtained as

$$R_j = \sum_j \int P(\mathbf{v}) \frac{\partial h_\sigma\left(rk_j(\nu, \mathcal{W}, \gamma)\right)}{\partial \mathbf{w}_j} D_\varepsilon(\nu, \omega_j, \gamma)\, d\mathbf{v} \tag{A.14}$$

with

$$\frac{\partial h_\sigma\left(rk_j(\nu, \mathcal{W}, \gamma)\right)}{\partial \mathbf{w}_j} = \left[h_\sigma\right]'\left(rk_j(\nu, \mathcal{W}, \gamma)\right) \cdot \frac{\partial rk_j(\nu, \mathcal{W}, \gamma)}{\partial \mathbf{w}_j}$$

and $\left[h_\sigma\right]'(\bullet)$ denotes the derivative of $h_\sigma^{NG}(\bullet)$. If $R_j$ is vanishing, then the derivative (A.13) yields the prototype learning rule (4.42) of FSNG. We decompose $R_j$ into $R_j = R_{j,1} + R_{j,2}$ such that

$$R_{j,1} = \int P(\mathbf{v}) \left[h_\sigma\right]'\left(rk_j(\nu, \mathcal{W}, \gamma)\right) \cdot D_\varepsilon(\nu, \omega_j, \gamma) \frac{\partial d(\mathbf{v}, \mathbf{w}_j)}{\partial \mathbf{w}_j} \sum_l \mathtt{h}(\triangle_{jl})\, d\mathbf{v}$$

and

$$-R_{j,2} = \sum_j \int P(\mathbf{v}) \left[h_\sigma\right]' (rk_j(\nu, \mathcal{W}, \gamma)) \cdot D_\varepsilon(\nu, \omega_j, \gamma) \cdot \frac{\partial d(\mathbf{v}, \mathbf{w}_j)}{\partial \mathbf{w}_j} \cdot \mathtt{h}(\triangle_{tl})\, d\mathbf{v}$$

with $\triangle_{mk} = d(\mathbf{v}, \mathbf{w}_m) - d(\mathbf{v}, \mathbf{w}_k)$. Thereby we used the fact that the derivative of the Heaviside function $\mathtt{H}(x)$ (2.14) is the Dirac distribution $\mathtt{h}(x)$, which is zero iff $x \neq 0$ and $\int \mathtt{h}(x)\, dx = 1$. It has to be mentioned that the rank function $rk_j(\nu, \mathcal{W}, \gamma)$ depends only on the dissimilarities $d(\mathbf{v}, \mathbf{w}_j)$ whereas the cost function depends on the extended dissimilarity $D_\varepsilon(\nu, \omega_j, \gamma)$, which incorporates also the class dissimilarities $\mathtt{h}(\mathbf{c}_\mathbf{v}, \mathbf{y}_j)$.

For $R_{j,2}$ we can interchange integration and summation. Further, $R_{j,2}$ is non-vanishing only for $d(\mathbf{v}, \mathbf{w}_t) = d(\mathbf{v}, \mathbf{w}_j)$ according to the Dirac distribution $\mathtt{h}(x)$. For those data points obviously the equation

$$\sum_k \mathtt{h}(\triangle_{jk}) = \sum_k \mathtt{h}(\triangle_{tk})$$

holds implying immediately the equivalence $rk_j(\nu, \mathcal{W}, \gamma) = rk_t(\nu, \mathcal{W}, \gamma)$. At this end, we obtain

$$-R_{j,2} = \int P(\mathbf{v}) \left[h_\sigma\right]' (rk_j(\nu, \mathcal{W}, \gamma)) \cdot D_\varepsilon(\nu, \omega_j, \gamma) \cdot \frac{\partial d(\mathbf{v}, \mathbf{w}_j)}{\partial \mathbf{w}_t} \cdot \sum_l \mathtt{h}(\triangle_{tl})\, d\mathbf{v}$$

which leads to $R_{j,1} = -R_{j,2}$ paying attention to the fact that $\mathtt{h}(x)$ is symmetric: $\mathtt{h}(x) = \mathtt{h}(-x)$.

Further, using the derivative $\frac{\partial D_\varepsilon(\nu, \omega_j, \gamma)}{\partial \mathbf{w}_j}$, the gradient $\frac{\partial E_{\mathrm{FSNG}}}{\partial \mathbf{w}_j}$ in (A.13) finally reduces to

$$\frac{\partial E_{\mathrm{FSNG}}}{\partial \mathbf{w}_j} = (1-\gamma) \int P(\mathbf{v})\, h_\sigma(rk_j(\nu, \mathcal{W}, \gamma)) (\gamma \cdot \delta(\mathbf{c}_\mathbf{v}, \mathbf{y}_j) + \varepsilon_\delta) \cdot \frac{\partial d(\mathbf{v}, \mathbf{w}_j)}{\partial \mathbf{w}_j} d\mathbf{v},$$

which is exactly the averaged prototype dynamic (4.42). This completes the proof for the prototype dynamic.

It remains to investigate the dynamic for the class labels $\mathbf{y}_j$. We have

$$\frac{\partial E_{\mathrm{FSNG}}}{\partial \mathbf{y}_j} = \tilde{R}_j + \gamma \int P(\mathbf{v})\, h_\sigma(rk_j(\mathbf{v}, \mathbf{w}_j)) \frac{\partial D_\varepsilon(\nu, \omega_j, \gamma)}{\partial \mathbf{y}_j} d\mathbf{v}$$

with

$$\tilde{R}_j = \sum_j \int P(\mathbf{v}) \frac{\partial h_\sigma(rk_j(\mathbf{v}, \mathbf{w}_j))}{\partial \mathbf{y}_j} D_\varepsilon(\nu, \omega_j, \gamma)\, d\mathbf{v}.$$

Yet, $h_\sigma(rk_j(\mathbf{v}, \mathbf{w}_j))$ is independent from the labels $\mathbf{y}_j$ and, therefore, the respective derivative $\frac{\partial h_\sigma(rk_j(\mathbf{v}, \mathbf{w}_j))}{\partial \mathbf{y}_j}$ is zero such that we get $\tilde{R}_j = 0$ for this case, immediately.

Hence, using the result for $\frac{\partial D_\varepsilon(\nu,\omega_j,\gamma)}{\partial \mathbf{y}_j}$ from (4.42), we finally obtain for the averaged label dynamic

$$\frac{\partial E_{\mathrm{FSNG}}}{\partial \mathbf{y}_j} = \gamma \int P(\mathbf{v})\, h_\sigma\left(rk_j(\mathbf{v}, \mathbf{w}_j)\right)\left((1-\gamma)\cdot d(\mathbf{v}, \mathbf{w}_j) + \varepsilon_d\right) \cdot \frac{\partial \delta(\mathbf{c}_\mathbf{v}, \mathbf{y}_j)}{\partial \mathbf{y}_j} d\mathbf{v}$$

corresponding to (4.42). This completes the proof for the desired FSNG dynamic.

# Bibliography

[Aas and Haff, 2006] Aas, K. and Haff, I. (2006). The generalized hyperbolic skew student t-distribution. *Journal Financial Econom*, 4(2):275–309.

[Aronszajn, 1950] Aronszajn, N. (1950). Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404.

[Augusteijn et al., 1995] Augusteijn, M., Clemens, L., and Shaw, K. (1995). Performance evaluation of texture measures for ground cover identification in satellite images by means of a neural network classifier. *IEEE Transaction on Geoscience and Remote Sensing*, 33(3).

[Bach et al., 2011] Bach, F., Jenatton, R., Mairal, J., and Obozinski, G. (2011). *Convex optimization with sparsity-inducing norms*. Optimization for Machine Learning, MIT Press.

[Backhaus et al., 2011] Backhaus, A., Bollenbeck, F., and Seiffert, U. (2011). High-throughput quality control of coffee varieties and blends by artificial neural networks and hyperspectral imaging. In *Proceedings of the 1st International Congress on Cocoa, Coffee and Tea, CoCoTea*.

[Backhaus and Seiffert, 2014] Backhaus, A. and Seiffert, U. (2014). Classification in high-dimensional spectral data: Accuracy vs. interpretability vs. model size. *Neurocomputing*, 131:15–22.

[Ball and Hall, 1967] Ball, G. H. and Hall, D. (1967). A clustering technique for summarizing multivariate data. *Behavioral Science*, 12(2):153–155.

[Bartlett and Mendelson, 2002] Bartlett, P. L. and Mendelson, S. (2002). Rademacher and gaussian complexicities: risk bounds ans structural results. *Journal of Machine Learning and Research*, 3:463–482.

[Bashyal and Venayagamoorthy, 2008] Bashyal, S. and Venayagamoorthy, G. K. (2008). Recognition of facial expressions using gabor wavelets and learning vector quantization. *Engineering Applications of Artificial Intelligence*, 21(7):1056 – 1064.

[Baum et al., 1979] Baum, L. E., Petrie, T., Soules, G., and Weiss, N. (1979). A maximization technique occuring in the statistical analysis of probablistic functions of markov chains. In *Annals of Mathematical Statistics*, volume 41, pages 164–171.

[Bezdek, 1974] Bezdek, J. (1974). Clustering validity with fuzzy sets. *Journal of Cybernetics*, 3:58 – 72.

[Bezdek, 1980] Bezdek, J. (1980). A convergence theorem for the fuzzy isodata clustering algorithm. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 2 (1):1 – 8.

[Bezdek and Hathaway, 2003] Bezdek, J. C. and Hathaway, R. J. (2003). Convergence of alternating optimization. *Neural, Parallel Scintific Computing*, 11(4):351 – 368.

[Biehl et al., 2013] Biehl, M., Bunte, K., and Schneider, P. (2013). Analysis of flow cytometry data by matrix relevance learning vector quantization. *PLoS ONE*, 8(3):e59401.

[Biehl et al., 2009] Biehl, M., Hammer, B., Schleif, F.-M., Schneider, P., and Villmann, T. (2009). Stationarity of matrix relevance learning vector quantization. Technical report, Computational Intelligence Group in Uni Leipzig.

[Biehl et al., 2012] Biehl, M., Schneider, P., Schmith, D., Stiekema, H. S., Taylor, A., Hughes, B., Shackleton, C., Stewart, P., and Arl, W. (2012). Matrix relevance lvq in steroid metabolomics based classification of adrenal tumors. In *Proc. of European Symposium on Artificial Neural Networks (ESANN'2012)*.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Regognition and Machine Learning*. Springer.

[Bishop et al., 1998] Bishop, M., Svensé, M., and Williams, C. (1998). Gtm: The generative topographic mapping. *Neural Computation*, 10(1):215–234.

[Bordes et al., 2005] Bordes, A., Ertekin, S., Weston, J., and Bottou, L. (2005). Fast kernel classifiers with online and active learning. *Journal on Machine Learning Research*, 6:1579 – 1619.

[Bottou and Bengio, 1994] Bottou, L. and Bengio, Y. (1994). Convergence proberties of the k-means algorithm. In Tesauro, G., Touretzky, D., and Leen, T. K., editors, *Proceedings of the NIPS 1994*, pages 585–592. MIT Press.

[Bunte, 2011] Bunte, K. (2011). *Adaptive dissimilarity measures, dimension reduction and visualization*. PhD thesis, University of Groningen.

[Bunte et al., 2012] Bunte, K., Schneider, P., Hammer, B., Schleif, F., Villmann, T., and Biehl, M. (2012). Limited rank matrix learning, discriminative dimension reduction and visualization. *Neural Networks*, 26:159–173.

[Buttoi and Bengio, 1994] Buttoi, L. and Bengio, Y. (1994). Convergence properties of k-means algorithm. In G. Tesauro, D. S. T. and Leen, T. K., editors, *Conference on Neural Information Processing Systems*, MIT, pages 585 – 592.

[Chapelle et al., 2006] Chapelle, O., Schölkopf, B., and Zien, A., editors (2006). *Semi-Supervised Learning*. MIT Press, Cambridge, Massachusetts, London, England.

[Chen and Mangasarian, 1995] Chen, C. and Mangasarian, O. (1995). Smooth methods for convex inequalities and linear complementarity problems. *Mathematical Programming*, 71:51–69.

[Cheng, 1997] Cheng, Y. (1997). Convergence and ordering of kohonen's batch map. *Neural Computation*, 9:1667–1676.

[Christmann and Steinwart, 2010] Christmann, A. and Steinwart, I. (2010). Universal kernels on non-standard input spaces. In J. Lafferty, C. K. I. Williams, J. S.-T. R. Z. and Culotta, A., editors, *in Advances in Neural Information Processing Systems*, volume 23, pages 406–414.

[Cichocki and Amari, 2010] Cichocki, A. and Amari, S. (2010). Families of alpha- beta- and gamma-divergences: Flexible and robust measures of similarities. *Entropy*, 12:1532–1568.

[Cohen, 1960] Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychologocal Measurement*, 20:37–46.

[Cottrell et al., 2006] Cottrell, M., Hammer, B., Hasenfuß, A., and Villmann, T. (2006). Batch and median neural gas. *Neural Networks*, 19:762–771.

[Crammer et al., 2003] Crammer, K., Gilad-Bachrach, R., A.Navot, and A.Tishby (2003). Margin analysis of the LVQ algorithm. In Becker, S., Thrun, S., and Obermayer, K., editors, *Advances in Neural Information Processing (Proc. NIPS 2002)*, volume 15, pages 462–469, Cambridge, MA. MIT Press.

[Donoho, 2004] Donoho, D. L. (2004). For most large underdetermined systems of linear equations the minimal ?1-norm solution is also the sparsest solution. *Comm. Pure Appl. Math*, 59:797–829.

[Dou et al., 2007] Dou, W., Ren, Y., Wu, Q., Ruan, S., Chen, Y., Bloyet, D., and Constans, J. (2007). Fuzzy kappa for the agreement measure of fuzzy classifications. *Neurocomputing*, 70(4-6):726–734.

[Dunn, 1973] Dunn, J. (1973). A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3(3):32–57.

[Edge et al., 2010] Edge, S., byrd, D., Compton, C., Fritz, A., Greene, and Trotty, A. (2010). *S. Edge and D. byrd and C. Compton and A. Fritz and Greene and A. TrottyAJCCC Cancer staging manual*. Spinger.

[Erwin et al., 1992] Erwin, E., Obermayer, K., and Schulten, K. (1992). Self-organizing maps: ordering, convergence properties and energy functions. *Biological Cybernetics*, 67(1):47–55.

[Fawcett, 2006] Fawcett, T. (2006). An introduction to roc analysis. *Pattern Recogn. Lett.*, 27(8):861–874.

[Fleiss, 2003] Fleiss, J. L. (2003). *Statistical Methods for Rates and Proportions*. Wiley.

[Fort et al., 2001] Fort, J.-C., Cottrell, M., and Letremy, P. (2001). Stochastic on-line algorithm versus batch algorithm for quantization and self organizing maps. In *Neural Networks for Signal Processing*, volume 6. IEEE Signal Processing Society Workshop.

[Frey and Dueck, 2007] Frey, B. J. and Dueck, D. (2007). Clustering by passing messages between data points. *Science*, 315:972–976.

[Geweniger, 2012] Geweniger, T. (2012). *Fuzzy Variants of Prototype Based Clustering and Classification Algorithms*. PhD thesis, University of Groningen.

[Geweniger et al., 2013] Geweniger, T., Fischer, L., Kaden, M., Lange, M., and Villmann, T. (2013). Clustering by fuzzy neural gas and evaluation of fuzzy clusters. *Computational Intelligence and Neuroscience*, 2013.

[Goutte and Gaussier, 2005] Goutte, C. and Gaussier, E. (2005). A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. *In Proceedings of 27th European conference on IR research (ECIR)*, pages 345–359.

[Haase, 2014] Haase, S. (2014). *The Application of Divergences in Prototype Based Vector Qu*. PhD thesis, University of Groningen.

[Hadamard, 1902] Hadamard, J. (1902). Sur les problèmes aux dérivées partielles et leur signification physique. *Princeton University Bulletin*, pages 49–52.

[Hammer et al., 2014a] Hammer, B., Hofmann, D., Schleif, F., and Zhu, X. (2014a). Learning vector quantization for (dis-)similarities. *Neurocomputing*, 131:43 – 51.

[Hammer et al., 2014b] Hammer, B., Nebel, D., Riedel, M., and Villmann, T. (2014b). Generative versus discriminative prototype based classification. In *Advances in Self-Organizing Maps and Learning Vector Quantization - Proceedings of the 10th International Workshop (WSOM2014), Mittweida, Germany 2014*, pages 123– 132.

[Hammer et al., 2011] Hammer, B., Schleif, F., and Zhu, X. (2011). Relational extensions of learning vector quantization. In *Neural Information Processing - 18th International Conference, ICONIP 2011, Shanghai, China, November 13-17, 2011, Proceedings, Part II*, pages 481–489.

[Hammer et al., 2001] Hammer, B., Strickert, M., and Villmann, T. (2001). On the generalization ability of grlvq networks. In G.Dorffner, H.Bischof, K., editor, *Artificial Neural Networks - ICANN'2001*, pages 731–736. Springer.

[Hammer et al., 2005] Hammer, B., Strickert, M., and Villmann, T. (2005). Prototype based recognition of splice sites. In Seiffert, U., Jain, L., and Schweitzer, P., editors, *Bioinformatic using Computational Intelligence Paradigms*, pages 25–56. Springer-Verlag.

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

[Hammer and Villmann, 2005] Hammer, B. and Villmann, T. (2005). Classification using non-standard metrics. In Verleysen, M., editor, *Proc. Of European Symposium on Artificial Neural Networks (ESANN'2005)*, pages 303–316, Brussels, Belgium. d-side publications.

[Harth, 2012] Harth, C. (2012). Erweiterung von generalized [releane | matrix]learnign vector quantization zur anwendung auf functionale daten. Master's thesis, Univ. of Appl. Sc. Mittweida.

[Hasenfuss, 2009] Hasenfuss, A. (2009). *Topographic Mapping of Dissimilarity Datasets*. PhD thesis, Univ. of Clausthal.

[Hasenfuss and Hammer, 2007] Hasenfuss, A. and Hammer, B. (2007). Relational topographic maps. In *Advances in Intelligent Data Analysis VII, 7th International Symposium on Intelligent Data Analysis, IDA 2007, Ljubljana, Slovenia, September 6-8, 2007, Proceedings*, pages 93–105.

[Hashimoto et al., 2009] Hashimoto, W., Nakamura, T., and Miyamoto, S. (2009). Comparison and evaluation of different cluster validity measures including their kernelization. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 13:204–209.

[Haykin, 1994] Haykin, S. (1994). *Neural Networks - A Comprehensive Foundation*. IEEE Press, New York.

[Hermann et al., 2002] Hermann, W., Barthel, H., Hesse, S., Grahmann, F., Kühn, H.-J., Wagner, A., and Villmann, T. (2002). Comparison of clinical types of wilson's disease and glucose metabolism in extrapyramidal motor brain regions. *Journal of Neurology*, 249 (7):896–901.

[Hermann et al., 2003] Hermann, W., Villmann, T., and Wagner, A. (2003). Elektrophysiologisches schädigungsprofil von patienten mit einem morbus wilson. *Der Nervenarzt*, 74(10):881–887.

[Herrmann et al., 1994] Herrmann, M., Bauer, H.-U., and Der, R. (1994). The 'perceptual magnet' effect: A model based on self-organizing feature maps. In Smith, L. S. and Hancock, P. J. B., editors, *Neural Computation and Psychology*, pages 107–116, Stirling. Springer-Verlag.

[Heskes, 1999] Heskes, T. (1999). Energy functions for self-organizing maps. In Oja, E. and Kaski, S., editors, *Kohonen Maps*, pages 303–315. Elsevier, Amsterdam.

[Holt et al., 2010] Holt, R. S., Mastromarino, P. A., Kao, E. K., and Hurley, M. B. (2010). Information theoretic approach for performance of multi-class assingments systems. *MIT Open Access Articles*, pages 1–12.

[Hoyer, 2002] Hoyer, P. O. (2002). Non-negative sparse coding. *Neural Networks for Signal Processing*, 12:557 – 565.

[I.Kabanikhin, 2011] I.Kabanikhin, S. (2011). *Inverse and Ill-Posed problems*. Series 55. Chapter 2 p. 24.

[Kaden et al., 2014] Kaden, M., Lange, M., Nebel, D., Riedel, M., Geweniger, T., and Villmann, T. (2014). Aspects in classification learning - review of recent developments in learning vector quantization. *Foundations of Computing and Decision Sciences*, 39(2):79–105.

[Kantorowitsch and Akilow, 1978] Kantorowitsch, I. and Akilow, G. (1978). *Funktionalanalyis in nomierten Räumen*, volume 2nd ed. Akademie-Verlag, Berlin.

[Karayiannis and Randolph-Gips, 2003] Karayiannis, N. B. and Randolph-Gips, M. M. (2003). Non-euclidean c-means clustering algorithms. *Intelligent Data Analysis*, 7:405–425.

[Kaski et al., 1998] Kaski, S., Kangas, J., and Kohonen, T. (1998). Bibliography of self-organizing map (SOM) papers: 1981–1997. *Neural Computing Surveys*, 1(3&4):1–176.

Available in electronic form at http://www.icsi.berkeley.edu/~jagota/NCS/: Vol 1, pp. 102–350.

[Kästner et al., 2011] Kästner, M., Backhaus, A., Geweniger, T., Haase, S., Seiffert, U., and Villmann, T. (2011). Relevance learning in unsupervised vector quantization based on divergences. In Laaksonen, J. and Honkela, T., editors, *Proc. of Workshop on Self-Organizing Maps (WSOM'2011)*, volume 6731 of *LNCS*, pages 90–100, Helsinki, Finland. Springer.

[Kästner and Villmann, 2012] Kästner, M. and Villmann, T. (2012). Fuzzy supervised self-organizing map for semi-supervised vector quantization. In L. Rutkowski, M. Korytkowski, R. S. R. T.-L. Z. J. Z., editor, *11th International Conference on Artificial Intelligence and Soft Computing ICAISC 2012*, pages 256–265, Zakopane. Springer.

[Kaufman and Rousseeuw, 1990] Kaufman, L. and Rousseeuw, P. J. (1990). *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley.

[Kohonen, 1982] Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59–69.

[Kohonen, 1986] Kohonen, T. (1986). Learning vector quantization for pattern recognition. *Technical Report*, TKK-F-A601. Helsinski Universtiy of Technology.

[Kohonen, 1998] Kohonen, T. (1998). The self-organizing map. *Neurocomputing*, 21:1–6.

[Kohonen, 2013] Kohonen, T. (2013). Essentials of the self-organizing map. *Neural Networks*, 37:52–65.

[Kohonen and Somervuo, 2002] Kohonen, T. and Somervuo, P. (2002). How to make large self-organizing maps for nonvectorial data. *Neural Networks*, 15:945–952.

[Krier et al., 2008] Krier, C., Rossi, F., Francois, D., and Verleysen, M. (2008). A data-driven functional projection approaches for the selection of feature ranges in spectra with ica or cluster analysis. *Chemom. Intell. Lab Syst.*, 91:43–53.

[Krier et al., 2009] Krier, C., Verleysen, M., Rossi, F., and Francois, D. (2009). Supervised variable clustering for classification of nir spectra. In Verleysen, M., editor, *Proceedings of teh 16 European Symposium on Artificial Neural Networks (ESANN)*, pages 263–268, Bruges, Belgium.

[Kästner et al., 2012] Kästner, M., Nebel, D., Riedel, M., Biehl, M., and Villmann, T. (2012). Differentiable kernels in generalized matrix learning vector quantization. In *Proc. of International Conference in Machine Learning and Application (ICMLA)*, volume 11 of *ISBN 978-1-4673-2326-0*, pages 14–19, Boca Raton. IEEE Press.

[Kästner et al., 2013] Kästner, M., Strickert, M., and Villmann, T. (2013). A sparse kernelized matrix learning vector quantization model for human activity recognition. In Verleysen, M., editor, *Proc. of European Symposium on Artificial Neural Networks (ESANN'2013)*, pages 449–454, Louvain-La-Neuve, Belgium.

[Kushner and Clark, 1978] Kushner, H. and Clark, D. (1978). *Stochastic Appproximation Methods for Constrained and Unconstrained Systems*. Springer-Verlag, New York.

[Lange et al., 2013] Lange, M., Kästner, M., and Villmann, T. (2013). About analysis and robust classification of searchlight fmri-data using machine learning classifiers. In *IJCNN*, pages 1–8.

[Lange and Villmann, 2013] Lange, M. and Villmann, T. (2013). Deriatives of $l_p$-norm and their approximations. Technical report, Machine Learning Reports, MLR-03-2013.

[Lange et al., 2014] Lange, M., Zühlke, D., Holz, O., and Villmann, T. (2014). Applications of lp-norms and their smooth approximations for gradient based learning vector quantization. In *22th European Symposium on Artificial Neural Networks, ESANN 2014, Bruges, Belgium, April 23-25, 2014*.

[Langrebe, 2003] Langrebe, D. (2003). *Signal Theory Methods in Multispectral Remote Sensing*. Wiley, Hoboken, New Jersey.

[Lee and Verleysen, 2005] Lee, L. and Verleysen, M. (2005). Generalization of the $l_p$ norm for times series and its application to self-organizing maps. In Corrtell, M., editor, *Proc. of Workshop on Self-Organizing Maps (WSOM)*, pages 733–740, Paris, Sorbonne.

[MacQueen, 1967] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In Cam, L. M. L. and J.Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–294. University of California Press.

[Martinetz et al., 1993] Martinetz, T., Berkovich, S., and Schulten, K. (1993). ¨Neural-gas¨ Network for Vector Quantization and its Application to Time-Series Prediction. *IEEE-Transactions on Neural Networks*, 4(4):558–569.

[Martinetz and Schulten, 1994] Martinetz, T. and Schulten, K. (1994). Topology representing networks. *Neural Networks*, 7 (3):507–522.

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

[Mercer, 1909] Mercer, J. (1909). Functions of positive and negative type and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society, London*, 209:415–446. London.

[Micchelli et al., 2006] Micchelli, C., Xu, Y., and Zhang, H. (2006). Universal kernels. *Journal of Machine Learning Research*, 7:2651–2667.

[Midenet and Grumbach, 1994] Midenet, S. and Grumbach, A. (1994). Learning {ASsociations} by self-organization: The {LASSO} model. *Neurocomputing*, 6(3):343 – 361.

[Nebel et al., 2013] Nebel, D., Hammer, B., and Villmann, T. (2013). A median variant of generalized learning vector quantization. In *Neural Information Processing - 20th International Conference, ICONIP 2013, Daegu, Korea, November 3-7, 2013. Proceedings, Part II*, pages 19–26.

[Nebel et al., 2014] Nebel, D., Hammer, B., and Villmann, T. (2014). Supervised generative models for learning dissimilarity data. In *22th European Symposium on Artificial Neural Networks, ESANN 2014, Bruges, Belgium, April 23-25, 2014*.

[Nebel and Villmann, 2013] Nebel, D. and Villmann, T. (2013). About the equivalence of robust soft learning vector quantization and soft nearest prototype classification. Technical report, Machine Learning Reports (MLR-02-2013). ISSN:1865-3960.

[Olshausen and Field, 1997] Olshausen, B. A. and Field, D. J. (1997). Sparse coding with an overcomplete basis set: a strategy employed by v1? *Vision Research*, 37:3311–3325.

[Pekalska and Duin, 2005] Pekalska, E. and Duin, R. P. (2005). *The Dissimilarity Representation for Pattern Recognition. Foundations and Applications.* World Scientific, Singapore. ISBN: 981-256-530-2.

[Platt, 1998] Platt, J. C. (1998). Sequential minimal optimization: A fast algorithm for training support vector machines. Technical report, ADVANCES IN KERNEL METHODS - SUPPORT VECTOR LEARNING.

[Powers, 2011] Powers, D. M. (2011). Evaluation: From precision, recall and f-factor to roc, informedness, markedness & correlation. *Journal of Machine Learning Technologies*, 2:37–63. ISSN 2229-2981.

[Principe, 2010] Principe, J. C. (2010). *Information Theoretic Learning*. Springer.

[Qin et al., 2004] Qin, A. K., , and Suganthan, P. N. (2004). A novel kernel prototype-based learning algorithm. *Pattern Recognition, International Conference on*, 4:621–624.

[Ramsay and Silverman, 2006] Ramsay, J. and Silverman, B. (2006). *Functional Data Analysis*, volume 2nd ed. Springer Science + Media, New York.

[Riedel and Nebel, 2012] Riedel, M. and Nebel, D. (2012). Generalized functional matrix learning vector quantization (gfmlvq). Master's thesis, University of Applied Sciences Mittweida.

[Riedel et al., 2013] Riedel, M., Rossi, F., Kästner, M., and Villmann, T. (2013). Regularization on relevance learning vector quantization using l1-norm. In Verleysen, M., editor, *Proc. of European Symposium on Artificial Neural Networks (ESANN'2013)*, pages 17–21, Louvain-La-Neuve, Belgium.

[Rijsbergen, 1979] Rijsbergen, C. (1979). *Information Retrieval*. Butterworths, London, 2nd edition.

[Rokach and Maimon, 2008] Rokach, L. and Maimon, O. (2008). *Data mining with decision trees: theory and applications*. World Scientific Pub Co Inc.

[Rossi et al., 2005] Rossi, F., Delannay, N., Conan-guez, B., and Verleysen, M. (2005). Representation of functional data in neural networks. *Neurocomputing*, 64:183–210.

[Sachs, 1992] Sachs, L. (1992). *Angewandte Statistik*. Springer Verlag, 7-th edition.

[Sato and Yamada, 1996] Sato, A. and Yamada, K. (1996). Generalized learning vector quantization. In Touretzky DS, Mozer MC, H. M., editor, *Advances in neural information processing systems*, volume 8, pages 423–429. MIT Press, Cambridge.

[Schleif, 2014] Schleif, F. (2014). Discriminative fast soft competitive learning. In *Artificial Neural Networks and Machine Learning - ICANN 2014 - 24th International Conference on Artificial Neural Networks, Hamburg, Germany, September 15-19, 2014. Proceedings*, pages 81–88.

[Schleif et al., 2007a] Schleif, F., Villmann, T., and Hammer, B. (2007a). Analysis of proteomic spectral data by multi resolution analysis and self-organizing maps. In *Applications of Fuzzy Sets Theory, 7th International Workshop on Fuzzy Logic and Applications, WILF 2007, Camogli, Italy, July 7-10, 2007, Proceedings*, pages 563–570.

[Schleif et al., 2007b] Schleif, F., Villmann, T., and Hammer, B. (2007b). Supervised neural gas for classification of functional data and its application to the analysis of clinical proteom spectra. In *Computational and Ambient Intelligence, 9th International Work-Conference on Artificial Neural Networks, IWANN 2007, San Sebastián, Spain, June 20-22, 2007, Proceedings*, pages 1036–1044.

[Schleif et al., 2011] Schleif, F.-M., Villmann, T., Hammer, B., and Schneider, P. (2011). Efficient kernelized prototype-based classification. *Journal of Neural Systems*, 21(6):443–457.

[Schölkopf and Smola, 2002] Schölkopf, B. and Smola, A. (2002). *Learning with Kernels*. MIT Press.

[Schmidt et al., 2007] Schmidt, M., Fung, G., and Rosales, R. (2007). Fast optimization methods for $l_1$-regularization: A comparative study and two new approaches. In Kok, J., Koronacki, J., Mantaras, R., Matwin, S., Mladenic, D., and Skowron, A., editors, *Machien Learning: ECML*, volume 4701 of *Lecture Notes in Computer Science*, pages 286–297. Springer Berlin Heidelberg.

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

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

[Seo et al., 2003] Seo, S., Bode, M., and Obermayer, K. (2003). Soft nearest prototype classification. *IEEE Transaction on Neural Networks*, 14:390–398.

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

[Sokolova and Lapalme, 2009a] Sokolova, M. and Lapalme, G. (2009a). A systematic analysis of performance measures for classification task. *Information Processing and Management*, 45:427–437.

[Sokolova and Lapalme, 2009b] Sokolova, M. and Lapalme, G. (2009b). A systemtiac analysis of performance measures for classification tasks. *Information Processing and Management*, 45:427–437.

[Sriperumbudur et al., 2011] Sriperumbudur, B. K., Fukumizu, K., and Lanckriet, G. R. (2011). Universality, characteristic kernels and rkhs embedding of measures. *Journal of Machine Learning Research*, 12:2389–2410.

[Steinwart, 2001a] Steinwart, I. (2001a). On the influence of the kernel on the consistency of support vector machines. *Journal on Machine Learning Research*, 2:67–93.

[Steinwart, 2001b] Steinwart, I. (2001b). On the influence of the kernel on the consistency of support vector machines. *Journal Machine Learning Research*, 2:67–93.

[Steinwart and Christmann, 2008] Steinwart, I. and Christmann, A. (2008). *Support Vector Machines*. Springer Science and Business Meda.

[Strickert, 2011] Strickert, M. (2011). Enhancing m | grlvq by quasi step discriminatory function training $2^{nd}$ order training. In Villmann, T. and Schleif, F.-M., editors, *Mittweida Workshop on Computational Intelligence (MIWOCI)*, volume 6 of *Machine Learning Reports*.

[Sutton and Barto, 1988] Sutton, R. S. and Barto, A. G. (1988). *Reinforcement Learning: An Introduction*. MIT Press Cambridge, Massachusetts London, England 1.

[Suzuki and Isozaki, 2005] Suzuki, J. and Isozaki, H. (2005). Sequence and tree kernels with statistical feature mining. In *NIPS*. Citeseer.

[T. Geweniger and Villmann, 2012] T. Geweniger, M. Kästner, M. L. and Villmann, T. (2012). Modified conn-index for the evaluation of fuzzy clusterings. In Verleysen, M., editor, *Proc. of European Symposium on Artificial Neural Networks (ESANN'2012)*, pages 465–470, Louvain-La-Neuve, Belgium.

[T. Landgrebe and Bradley, 2006] T. Landgrebe, P. Paclìk, R. D. and Bradley, A. (2006). Precision-recall operating characteristic (p-roc) curves in imprecise environments. In *Proceedings of ICPR*.

[T. Villmann and Riedel, 2012] T. Villmann, M. Kästner, D. N. and Riedel, M. (2012). Icmla face recognition challenge - results of the team computational intelligence mittweida. In *Proc. of 11th Annual International Conference in Machine Learning and Application*, ISBN 978-1-4673-2326-0, pages 40–46, Boca Raton. IEEE Press.

[Taşdemir and Merényi, 2011] Taşdemir, K. and Merényi, E. (2011). A validity index for prototype-based clustering of datasets with complex structures. *IEEE Transactions on Systems, Man, and Cybernetics*, 41(4):1039–1053.

[Tibshirani, 1996] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Society*, 58(1):267–288.

[Tipping, 2001] Tipping, M. E. (2001). Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244.

[Tsao et al., 1994] Tsao, E. C.-K., Bezdek, J. C., and Pal, N. R. (1994). Fuzzy kohonen clustering networks. *Pattern Recognition*, 27(5):757 – 764.

[Ultsch and Semon, 1990] Ultsch, A. and Semon, H. P. (1990). Kohonen's self-organizing feature maps for exploratory data analysis. *International Neural Network Conference*, pages 305–308.

[Umer and Khiyal, 2007] Umer, M. F. and Khiyal, M. S. H. (2007). Classification of textual documents using learning vector quantization. *Information Technology Journal*, 6 (1):154–159.

[v. d. Maaten et al., 2009] v. d. Maaten, L., Postma, E., and v. d. Herik, J. (2009). Dimensionality reduction: A comparative review. Technical report, Tilburg centre for Creative Computing, Tilburg Univers.

[v. Luxburg, 2007] v. Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17 (4).

[Vapnik, 1995] Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

[Vesanto, 1999] Vesanto, J. (1999). Som-based data visualization methods. *Intell. Data Anal.*, 3(2):111–126.

[Villa and Rossi, 2005] Villa, N. and Rossi, F. (2005). Support vector machine for functional data classification. In *In Proceedings of ESANN 2005*, pages 467–472. Miscellaneous.

[Villmann, 2006] Villmann, T. (2006). *Neural Maps and Learning Vector Quantization for Data Mining - Theory and Application* -. PhD thesis, University of Leipzig.

[Villmann, 2007] Villmann, T. (2007). Sobolev metrics for learning of functional data - mathematical and theroretical aspects. Technical Report MLR-03-2007, Machine Learning Reports,.

[Villmann et al., 2015] Villmann, T., Haase, S., and Kaden, M. (2015). Kernelized vector quantization in gradient-descent learning. *Neurocomputing*, 147(0):83 – 95. Advances in Self-Organizing Maps Subtitle of the special issue: Selected Papers from the Workshop on Self-Organizing Maps 2012 (WSOM 2012).

[Villmann and Hammer, 2009] Villmann, T. and Hammer, B. (2009). Functional principal component learning using oja's method and sobolev norms. In Principe, J. C. and Miikkulainen, R., editors, *Workshop on Self Organizing Maps*, pages 325–333. Srpinger-Verlag Berlin Heidelberg. LNCS 5629.

[Villmann et al., 2006a] Villmann, T., Hammer, B., Schleif, F., Geweniger, T., and Hermann, W. (2006a). Fuzzy classification by fuzzy labeled neural gas. *Neural Networks*, 19(6-7):772–779.

[Villmann et al., 2008] Villmann, T., Hammer, B., Schleif, F.-M., Hermann, W., and Cottrell, M. (2008). Fuzzy classification using information theoretic learning vector quantization. *Neurocomputing*, 71:3070–3076.

[Villmann and Lange, 2015] Villmann, T. and Lange, M. (2015). A comment on the functional $l_p^{TS}$- measure regarding the norm propertie. *Machine Learning Reports*, 2:1–13.

[Villmann et al., 2007] Villmann, T., Schleif, F., Merényi, E., and Hammer, B. (2007). Fuzzy labeled self-organizing map for classification of spectra. In *Computational and Ambient Intelligence, 9th International Work-Conference on Artificial Neural Networks, IWANN 2007, San Sebastián, Spain, June 20-22, 2007, Proceedings*, pages 556–563.

[Villmann et al., 2003] Villmann, T., Schleif, F. M., and Hammer, B. (2003). Supervised neural gas and relevance learning in learning vector quantization. In *Proc. International Workshop on Self-Organizing Maps (WSOM'2003)*, pages 47–52, Kitakyushu.

[Villmann et al., 2006b] Villmann, T., Seiffert, U., Schleif, F.-M., Brüß, C., Geweniger, T., and Hammer, B. (2006b). Fuzzy labeled self-organizing map with label-adjusted prototypes. In Schwenker, F. and Marinai, S., editors, *Artificial Neural Networks in Pattern Recognition*, volume 4087 of *Lecture Notes in Computer Science*, pages 46–56. Springer Berlin Heidelberg.

[Witoelar et al., 2010] Witoelar, A., Ghosh, A., de Vries, J. J. G., Hammer, B., and Biehl, M. (2010). Window-based example selection in learning vector quantization. *Neural Computation*, 22(11):2924–2961.

[Zalik and Zalik, 2011] Zalik, K. R. and Zalik, B. (2011). Validity index for clusters of different sizes and densities. *Pattern Recognition Letters*, 32(2):221 – 234.

[Zhang et al., 2009] Zhang, H., Y.Xu, and J.Zhang (2009). Reproducing kernel banach spaces for machine learning. *Journal Machine Learning Research*, 10:2741–2775.

[Zühlke et al., 2009] Zühlke, D., Geweniger, T., Heimann, U., and Villmann, T. (2009). Fuzzy fleiss-kappa for comparison of fuzzy classifiers. In *ESANN 2009, 17th European Symposium on Artificial Neural Networks, Bruges, Belgium, April 22-24, 2009, Proceedings*.

## Selbstädigkeitserklärung

Hiermit erkläre ich, die vorliegende Dissertation selbständig und ohne unzulässige fremde Hilfe angefertigt zu haben. Ich habe keine anderen als die angeführten Quellen und Hilfsmittel benutzt und sämtliche Textstellen, die wörtlich oder sinngemäßaus veröffentlichten oder unveröffentlichten Schriften entnommen wurden, und alle Angaben, die auf mündlichen Auskünften beruhen, als solche kenntlich gemacht. Ebenfalls sind alle von anderen Personen bereitgestellten Materialen oder erbrachten Dienstleistungen als solche gekennzeichnet.

## Declaration

I hereby testify that I have written the whole of this thesis myself and used no sources or aids (including electronic media and online sources) other than those named. All passages taken from a source, whether verbatim or in substance, have been indicated as such.

Name:

Mittweida, the