Neurocomputing 141 (2014) 84–96

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Learning interpretable kernelized prototype-based models

Daniela Hofmann*, Frank-Michael Schleif, Benjamin Paaßen, Barbara Hammer

CITEC Center of Excellence, Bielefeld University, Germany

ARTICLE INFO

Article history:
Received 8 July 2013
Received in revised form
19 December 2013
Accepted 28 March 2014
Available online 13 April 2014

Keywords:
Kernel learning vector quantization
Proximity data
Sparsity
Interpretable models

ABSTRACT

Since they represent a model in terms of few typical representatives, prototype based learning such as learning vector quantization (LVQ) constitutes a directly interpretable machine learning technique. Recently, several LVQ schemes have been extended towards a kernelized or dissimilarity based version which can be applied if data are represented by pairwise similarities or dissimilarities only. This opens the way towards its application in domains where data are typically not represented in vectorial form. Albeit kernel LVQ still represents models by typical prototypes, interpretability is usually lost this way: since no vector space model is available, prototypes are represented indirectly in terms of combinations of data. In this contribution, we extend a recent kernel LVQ scheme by sparse approximations to overcome this problem: instead of the full coefficient vectors, few exemplars which represent the prototypes can be directly inspected by practitioners in the same way as data in this case. For this purpose, we investigate different possibilities to approximate a prototype by a sparse counterpart during or after training relying on different heuristics or approximation algorithms, respectively, in particular sparsity constraints while training, geometric approaches, orthogonal matching pursuit, and core techniques for the minimum enclosing ball problem. We discuss the behavior of these methods in several benchmark problems as concerns quality, sparsity, and interpretability, and we propose different measures how to quantitatively evaluate the performance of the approaches.

1. Introduction

Due to their intuitive learning and classification rule based on a winner-takes-all scheme, prototype-based techniques such as learning vector quantization (LVQ) enjoy a great popularity in diverse application domains ranging from telecommunication and robotics up to bioinformatics and data mining [32,4,20]. Apart from an only linear training time and its suitability for online scenarios, as demonstrated e.g. in [31,15], one of its benefits is given by the fact that models are represented in terms of few prototypes which can be inspected by practitioners in the same way as data. Hence this inherent representation scheme lends itself as an intuitive interface to the model, unlike many black box alternatives in machine learning which offer state-of-the-art results but, usually, do not provide a justification why a certain classification takes place [1]. In complex settings where the overall task is not necessarily clear a priori or in settings where the human has to take responsibility for a subsequent action, interpretability becomes crucial: here, human insight is often the only way to further specify a priorly unclear training setting or to substantiate mere observations by causalities. Due to this reason, there is an increasing demand of interpretable models which provide a human understandable interface to their decisions besides excellent classification accuracy in areas such as biomedical data analysis or interactive data inspection [56].

Recently, quite a few approaches have addressed the interpretability of powerful machine learning algorithms, including, for example, intelligent approximation techniques and feature selection mechanisms for SVM, blind signal separation, enhanced score methods, or visualization techniques [44,54,8,53,23]. One prominent example, for which interpretability is guaranteed per the design of the model, is offered by prototype based techniques such as learning vector quantization (LVQ) or generalizations thereof as proposed in [48,50,32,7]. LVQ relies on prototypical class representatives as model parameters. Decisions are taken based on the distance of a data point and the prototypes by means of a winner-takes-all rule. Interestingly, some LVQ techniques can be easily enhanced such that they provide an inherent low dimensional visualization of their decisions [11], or an extension of the models by directly interpretable relevance terms is possible [49,48]. Further, very strong learning theoretical guarantees substantiate LVQ algorithms as classification models with excellent generalization behavior [3,5,49].

Classical LVQ methods are restricted to vectorial data such that they cannot be applied if data are non-vectorial and represented in terms of pairwise similarities or dissimilarities. Examples for such

* Corresponding author.
E-mail address: dhofmann@techfak.uni-bielefeld.de (D. Hofmann).

http://dx.doi.org/10.1016/j.neucom.2014.03.003
0925-2312/

settings include structured data such as graphs, trees, sequence data, XML, or the like [17,19,46]. Often, these data can be addressed by means of a dedicated similarity measure or kernel, including e.g. sequence alignment, the normalized compression distance, graph kernels, or similar [19,13,12,40,26,30,34,35]. As such, the similarity or dissimilarity measure can serve as a canonical interface of the model towards the given data set, as is the case e.g. in popular kernel approaches.

Several extensions of prototype methods to general distances or kernels have recently been proposed, see e.g. [33,14,24,9,42,29,18,27,38,41]. The key problem which is addressed in these approaches is the definition of a space where prototypes can be represented since no embedding vector space is explicitly available for this purpose. Some of these approaches restrict the prototype locations to exemplars, i.e. data points, and adapt prototypes within this discrete set. Alternatives rely on an implicit embedding of the data in a kernel space, or, more generally, pseudo-Euclidean space or Krein space, in which vector operations can be done [39]. Concrete learning algorithms usually provide means of how to perform this embedding implicitly by means of kernelization or relationalization. This technique results in methods which have squared complexity as opposed to cubic complexity for an explicit embedding result. Interestingly, approximation techniques as proposed in [21,24,47] can improve the complexity to linear time. While exemplar based techniques often suffer from the restricted numerical flexibility, relational or kernel approaches in particular have obtained results which are competitive to state-of-the-art alternatives such as SVM [29,25].

For kernel LVQ schemes, one important property of prototype-based techniques is lost: prototypes are no longer given as explicit points in the data space, rather, an indirect representation as a linear combination of an underlying (usually not explicitly given) feature space is used. Thus, interpretability of the models, one of the main benefits of LVQ techniques, is no longer given. In this contribution, we address the question how to get around this problem by means of sparse approximations of prototypes. In this case, prototypes are represented by one or few exemplars only, whereby the latter can be directly inspected by practitioners in the same way as data. At the same time, training benefits from the larger flexibility of a continuous adaptation space as provided by the full model.

The principle of sparsity constitutes a common paradigm in nature-inspired learning, as discussed e.g. in the seminal work [37]. Interestingly, apart from an improved complexity, sparsity can often serve as a catalyzer for the extraction of semantically meaningful entities from data. In our case, the basic entities are represented by the data itself, and the task is to approximate given prototypes by sparse counterparts, thereby minimizing the loss of accuracy. It is well known that the problem of finding smallest subsets of coefficients such that a set of linear equations can still be fulfilled constitutes an NP hard problem, being directly related to NP-complete subset selection. Because of this fact, approximation techniques have to be considered, one popular approach being e.g. a l_1-relaxation of the problem [16] as used in LASSO.

In this contribution, we propose a few possibilities to approximate prototypes in a classical LVQ scheme by sparse approximations, thereby partially relying on classical solutions, but also taking into account simple heuristics which are motivated by the underlying geometrical background. Thereby, we propose one technique which emphasizes sparsity already while training, comparing this to two mathematical approximation schemes of the representation, namely classical orthogonal matching pursuit [10] and core techniques to approximately solve the minimum enclosing ball problem for the receptive fields of prototypes. As an alternative, we investigate two simple heuristics: an approximation of the prototypes by their closest exemplars, and a simple numerical rounding of the coefficient vector obtained by full training. We investigate the performance of these different techniques as concerns their classification accuracy and degree of sparsity. As one quantitative measure which can be related to the model interpretability, we use Rissanen's description length principle in a supervised setting as well as the overall data entropy to judge the representativity of prototypes in an unsupervised perspective [43].

Now we first introduce robust soft learning vector quantization (RSLVQ) as a LVQ scheme based on a statistical model where training can be derived as likelihood ratio optimization [50], and its extension towards general kernels [25,29]. Afterwards, we introduce different sparse approximation schemes for the representation of prototypes. We test the approaches using different benchmarks from similarity based learning [12] and evaluate the degree of sparsity obtained in the diverse approaches as well as their accuracy. We conclude with an interpretation of the results in the light of the data signature.

2. Kernel robust soft learning vector quantization

LVQ as originally proposed by Kohonen constitutes a very intuitive classifier which bases its decision on a winner-takes-all scheme and its learning rule on variants of Hebbian learning. Original LVQ 1 is surprisingly good in typical model situations as investigated e.g. in [5], but its adaptation rule is based on heuristic grounds only and cannot be interpreted as direct optimization of a valid cost function [6]. One of the first proposals of an underlying cost function related to large margin maximization can be found in [45], see e.g. [28,49] for a corresponding proof. The alternative proposal presented in [50] takes the perspective of generative models by relying on a mixture of Gaussians. A learning rule similar to LVQ2.1 can be derived thereof as likelihood ratio maximization.

Formally, assume that data $\xi_i \in \mathbb{R}^n$ are labeled y_i. A trained RSLVQ network represents a mixture distribution characterized by m prototypes $w_j \in \mathbb{R}^n$. The labels of the prototypes $c(w_j)$ are fixed. σ_j denotes the bandwidth. Mixture component j induces $p(\xi|j) = \text{const}_j \cdot \exp(f(\xi, w_j, \sigma_j^2))$ with normalization constant const_j and function $f(\xi, w_j, \sigma_j^2) = -\|\xi - w_j\|^2/\sigma_j^2$. The probability of data point ξ is defined as mixture $p(\xi|W) = \sum_j P(j) \cdot p(\xi|j)$ with prior $P(j)$ and parameters W of the model. The probability of a data point ξ and a given label y is $p(\xi, y|W) = \sum_{c(w_j)=y} P(j) \cdot p(\xi|j)$. Learning aims at an optimization of the log likelihood ratio

$$L = \sum_i \log \frac{p(\xi_i, y_i|W)}{p(\xi_i|W)}.$$

For optimization, usually a stochastic gradient ascent is used which yields update rules similar to LVQ2.1 provided class priors are equal, see [50] for details.

Given a novel data point ξ, its class label is the most likely label y corresponding to a maximum value $p(y|\xi, W) \sim p(\xi, y|W)$. For typical settings, this rule can be approximated by the standard winner-takes-all rule. We refer to the data ξ_i which are closest to a given prototype w_j as the receptive field R_j of the prototype.

In this standard form, RSLVQ can be used to classify Euclidean vectors only. Often, data are presented in more general form, representing pairwise similarities or dissimilarities of the data. Depending on whether the underlying similarity corresponds to an Euclidean feature space, an implicit underlying vector space is present in the case of kernel variants of prototype based techniques (see e.g. [9,29,42,41,47,57]), or a more general Krein space is present in relational variants (see e.g. [24,38,25]). Here we consider a recent kernelized version of RSLVQ model [50,29,25]. We assume a fixed kernel k corresponding to a feature map Φ. We set

$k_{il}:=k(\xi_i,\xi_l)=\Phi(\xi_i)^t\Phi(\xi_l)$. Usually, pairwise similarities are given and the feature map Φ is not known. A matrix $K=(k_{il})_{i,l}$ corresponds to a valid kernel matrix if and only if it is positive-semidefinite.

Since the feature space is usually not known, prototypes have to be represented implicitly in feature space. Usually, one restricts to linear combinations

$$w_j=\sum_i \gamma_{ji}\Phi(\xi_i).$$

In this setting, the cost function of RSLVQ becomes

$$L=\sum_i \log \frac{\sum_{c(w_j)=y_i} P(j)p(\Phi(\xi_i)|j)}{\sum_j P(j)p(\Phi(\xi_i)|j)}.$$

Here, we assume equal bandwidth $\sigma^2=\sigma_j^2$, for simplicity. Further, we assume constant prior $P(j)$ and mixture components induced by normalized Gaussians. These can be computed in the data space based on the Gram matrix because of the identity

$$\|\Phi(\xi_i)-w_j\|^2=\left\|\Phi(\xi_i)-\sum_m \gamma_{jm}\Phi(\xi_m)\right\|^2=k_{ii}-2\cdot\sum_m \gamma_{jm}k_{im}+\sum_{s,t}\gamma_{js}\gamma_{jt}k_{st}$$

where the distance in the feature space is referred to by $\|\cdot\|^2$.

There are two ways to optimize the cost function of kernel RSLVQ as explained in [25] in terms of a general framework: it L can be optimized directly with respect to the model parameters γ_{jm} by means of a gradient ascent technique. As an alternative, the cost function can be optimized with respect to the prototypes w_j, and the resulting update rules can be decomposed into contributions of the coefficient vectors γ_{jm}, resulting in update rules for the latter. Note that there is no guarantee that the gradient commutes with linear combinations of parameters such that the two update rules yield numerically different behavior, albeit the same local and global minima are present. Further, it is not clear a priori whether a decomposition of the update rule of w_j in terms of coefficients is possible; this is indeed not the case if adaptation takes place in the pseudo-Euclidean space, while Euclideanity allows such a decomposition, see [25].

Here, we rely on an optimization of the costs by implicit updates of the prototypes, which exactly mimics the numerical behavior of the vectorial setting. It has been shown in [29,25] that a stochastic gradient ascent of the cost function with respect to the prototypes can be expressed in terms of the coefficients only, leading to the following adaptation rules:

$$\Delta\gamma_{jm}\sim\begin{cases}-(P_y(j|\Phi(\xi_i))-P(j|\Phi(\xi_i)))\gamma_{jm} & \text{if } \xi_m\neq\xi_i, c(w_j)=y_i\\ (P_y(j|\Phi(\xi_i))-P(j|\Phi(\xi_i)))(1-\gamma_{jm}) & \text{if } \xi_m=\xi_i, c(w_j)=y_i\\ P(j|\Phi(\xi_i))\gamma_{jm} & \text{if } \xi_m\neq\xi_i, c(w_j)\neq y_i\\ -P(j|\Phi(\xi_i))(1-\gamma_{jm}) & \text{if } \xi_m=\xi_i, c(w_j)\neq y_i\end{cases}$$

It performs exactly the same updates as RSLVQ in the feature space if prototypes are in the linear span of the data. Often, a further restriction of the parameters to the convex hull takes place to ensure a representative location of the prototypes within the convex hull of the data. We will follow this principle to already boost the interpretability of the prototype coefficients while training.

Note that, unlike vectorial RSLVQ, prototypes are represented implicitly in terms of linear combinations. The inspection of a prototype thus requires to inspect the coefficients representing the prototype γ_j and all data, the latter usually being characterized in terms of pairwise similarities only. Thus, the method does no longer give interpretable results, and sparsity of the model is lost.

3. Approximation of the prototypes

Kernel RSLVQ yields prototypes which are implicitly represented as linear combinations of data points. Since the training algorithm and classification depends on pairwise distances only, simple linear algebra allows us to compute the distance of a data point and a prototype based on the pairwise similarity of the data point and all training data only. However, sparseness of the prototype is lost this way.

Here we propose different ways to arrive at sparse prototype representations. If only a few coefficients γ_{jm} are non-vanishing, a direct inspection of the corresponding exemplars ξ_m allows practitioners to judge the characteristics of the correlated prototype and its receptive field by a direct inspection of the exemplars. A sparse representation of a prototype $w_j=\sum_m\gamma_{jm}\Phi(\xi_m)$ refers to a set of one or more prototypes $\{\tilde{w}_j^i|i\}$ of the form

$$\tilde{w}_j^i=\sum\tilde{\gamma}_{jm}^i\Phi(\xi_m)$$

such that

- the size of this set is small, ideally, only one approximating prototype $\tilde{w}_j^1$ for w_j is necessary,
- these vectors are sparse, i.e. $|\tilde{\gamma}_j^i|_0$ is as small as possible,
- the set approximates w_j in the sense that the receptive field of w_j as compared to the union of the receptive fields of its approximations $\tilde{w}_j^i$ contains approximately the same set of data points.

One possibility to ensure that the last condition holds is to enforce $\tilde{w}_j^i\approx w_j$ as measured by the distance in the feature space.

This formulation includes as a subproblem the task to find a vector $\tilde{w}_j=\sum\tilde{\gamma}_{jm}\Phi(\xi_m)=w_j$ such that $|\tilde{\gamma}_j|_0$ is minimum. This problem is NP-hard, such that we have to rely on approximations. In the following, we introduce a variety of possible schemes.

3.1. Sparse training

A classical way to enforce sparsity constraints consists in the addition of a regularization term while training. This technique has been proposed, among others, in the pioneering work of Olshausen and Field based on a probabilistic model, for example [37]. Thus, we substitute the cost function L by the sum

$$L-Const\cdot S(\gamma)$$

where $S(\gamma)$ constitutes a constraint which emphasizes sparse solutions such as

$$S(\gamma)=\sum_{ji}|\gamma_j^i|_1$$

and $Const>0$ is a priorly chosen constant which weighs the two objectives of the combination. Optimization of these costs can be done by a subgradient method [51], which reduces to a standard gradient ascent for most of the regions. For $\gamma_j^i=0$, the subgradient is set to the constant 0 to emphasize sparse solutions. Note that, this way, sparse prototypes are chosen already while training, which has usually the effect that the final location of the resulting prototypes can be very different from the prototypes obtained by standard kernel RSLVQ without sparsity constraint. This technique is the only one among the proposed ones which changes the shape of the prototypes already while training, all other techniques start from a trained set of prototypes and try to exchange the linear combinations by a sparse variant. Therefore, we refer to this method as *sparse training* in the following.

3.2. Simple heuristic approximation of the prototypes

Geometric heuristic: As a simple alternative, we propose two very simple approximation schemes which substitute trained prototypes by sparse approximations. The first approach relies on the geometry of LVQ. For kernels, the distance of prototypes to points in their receptive field is changed to a small amount only, if

we approximate the prototype by the closest exemplar. As a generalization thereof, in particular to meet settings where the feature space is not densely populated, we can use the K_{approx} closest exemplars for some fixed K_{approx}. Note that this method, which we refer to as K_{approx}-*approximation* in the following, represents a prototype by a set of K_{approx} new sparse ones with l_0 norm equal to one.

Numerical heuristic: As an alternative, we can consider the coefficient vector γ_j and take the size of the coefficients as an indicator for the importance of the underlying exemplar. For the K_{hull}-*convex hull*, we select the K_{hull} largest coefficients γ_{jm} and we delete all but these coefficients in the vector γ_j. This is then normalized to 1: $\sum_m \tilde{\gamma}_{jm} = 1$; thereby, we neglect the upper index since only one prototype is used for the approximation.

3.3. Approximate representation of the prototypes

As an alternative to these simple heuristic approximation schemes, we can use more fundamental optimization techniques which try to represent a given prototype as accurately as possible regarding some explicit mathematical objective. Based on such an explicit objective, an optimization can be performed.

3.3.1. Numeric approximation

We can formalize the task to approximate a given prototype as the mathematical objective to approximate a prototype by a sparse linear combination of data such that the residual error of this approximation and the original prototype is as small as possible. This corresponds to the following mathematical problem (again, we use only one prototype for the approximation and, in consequence, neglect the corresponding index):

$$\begin{array}{ll} \min & |\tilde{\gamma}_j|_0 \\ \text{such that} & \left|\sum_m \tilde{\gamma}_{jm}\Phi(\xi_m) - w_j\right| \le \epsilon \end{array}$$

for a given $\epsilon > 0$. It is well-known that this problem is NP hard. Hence a variety of approximate solution strategies exist in the literature. Here, we rely on a popular and very efficient approximation offered by *orthogonal matching pursuit* (OMP) [10]. Given an acceptable error $\epsilon > 0$ of the approximation, a greedy approach is taken: the algorithm iteratively determines the most relevant direction and the optimum coefficient for this axis to minimize the remaining residual error. The algorithm can be easily kernelized, such that it can directly be used in our setting, where we assume a normalized kernel $k_{mm} = 1$ corresponding to a fixed length $\Phi(\xi_i)$ (alternatively, the normalization could be added to the greedy selection step):

Kernelized OMP:
$I := \emptyset$;
$\tilde{\gamma}_j := 0$;
while $(\gamma_j - \tilde{\gamma}_j)^t K(\gamma_j - \tilde{\gamma}_j) > \epsilon^2$ **do**
  $r := \gamma_j - \tilde{\gamma}_j$;
  $l_0 := \text{argmax}_l |[Kr]_l|$;
  $I := I \cup \{l_0\}$
  $\tilde{\gamma}_{jm} := (K_{II})^{-1} K_{Im}$ with $K_{II} :=$ restriction of K to index set I;
end while
return $\tilde{\gamma}_j$;

3.3.2. Geometric approximation

An alternative mathematical approximation can be derived based on a geometric view. The prototype represented by γ_j is located at a central position of its receptive field, since it represents the center of the corresponding Gaussian mode. We denote the latter receptive field of w_j by R_j. Under the assumption of spherical classes, we can characterize a prototype as the center of a ball which encloses all data assigned to it. To achieve uniqueness, we choose the smallest ball. The following geometric optimization problem referred to as *minimum enclosing ball* (*MEB*) results:

$$\begin{array}{ll} \min_{R^2, C} & R^2 \\ \text{such that} & \|C - \Phi(\xi_i)\|^2 \le R^2, \quad \forall \xi_i \in R_j \end{array}$$

here C is the center and R the radius of the MEB. We expect that $C \approx w_j$. The key observation of a sparse approximation technique starting from this characterization consists in the fact that the MEB can be approximately solved with a sparse vector C where the degree of sparsity is independent of the size of R_j. Further, a linear time approximation algorithm is available, see [2]. We shortly outline the idea of this sparse approximation, typically referred to as core approximation.

First, the dual problem of MEB can be phrased as follows:

$$\begin{array}{ll} \min_{\alpha_i \ge 0} & \sum_{ij} \alpha_i \alpha_j k_{ij} - \sum_i \alpha_i k_{ii}^2 \\ \text{where} & \sum_i \alpha_i = 1 \end{array}$$

Any solution of the dual problem gives rise to a primal solution in terms of $C = \sum \alpha_i \Phi(\xi_i)$. This dual is a convex problem with a unique solution, but worst case effort $\mathcal{O}(|R_j|^3)$ and no bound on the sparsity of the resulting solution. Therefore, this problem is not solved for the entire receptive field R_j, rather, starting from the empty set, a core set of points is built for which this dual problem is solved. A surprising fact proved e.g. in [2] is that a fixed finite number of such points are sufficient to form a core set which represents the entirety of R_j. The size is thereby independent of the size of R_j and the dimensionality of points.

This iterative algorithm to determine a core set uses the dual MEB as a subroutine. It terminates with a core set of limited size as a subset of R_j, for which the dual variables α_i induce a center of the MEB for the entirety of R_j. We refer to this sparse center as $\tilde{w}_j$:

MEB:
$S := \{\xi_i, \xi_m\}$ for a pair of largest distance $\|\Phi(\xi_i) - \Phi(\xi_m)\|^2$ in R_j
repeat
  solve MEB $(S) \to C, R$
  if exists $\xi_l \in R_j$ where $\|\Phi(\xi_l) - C\|^2 > R^2(1+\epsilon)^2$ **then**
    $S := S \cup \{\xi_l\}$
  end if
until all ξ_l are covered by the $R(1+\epsilon)$ ball in the feature space
return $\tilde{w}_j := C$

It has been proved in [2] that the number of loops of this algorithm is limited by a constant of order $\mathcal{O}(1/\epsilon^2)$ independent of R_j. In each loop, the dual MEB problem is solved for a small subset S of constant size, such that each loop has linear complexity only. An approximation of w_j as center of an approximate MEB is given by the dual variables of the found core set: $C_j = \sum_{i \in S} \alpha_i \Phi(\xi_i)$ hence a sparse approximation of w results by setting $\tilde{\gamma}_{ji}$ to α_i iff the coefficient i corresponds to a core point. We arrive at a sparse solution, whereby the quality of the approximation ϵ determines the resulting sparsity. Since data are used in the form of dot products only, all computations can be kernelized. Note that similar tricks have been used to speed up e.g. support vector machine training, see [52].

3.4. Characteristics of the techniques

Note that the proposed techniques differ in several characteristics, regarding

- their motivation being heuristics (for the K-approximation and K-convex hull) or grounded in an explicit mathematical objective to approximate the prototypes,
- their application during or after training: only the sparse approximation changes the representation of prototypes already while training,
- the way in which the degree of sparsity can be controlled,
- the way in which prototypes are represented in a sparse approximation; these correspond to one exemplar for a heuristic approximation using $K=1$, a set of exemplars for the K_{approx}-approximation, or a sparsely populated element of the kernel space for all other techniques. In consequence, classification takes place by computing the distance to the new exemplar, or the minimum distance to all exemplars in the set representing a prototype in the case of the K_{approx}-approximation,
- the sign and size of the coefficients; for RSLVQ, the coefficients are convex to increase interpretability, and we would like to maintain this fact also for the approximations. While OMP restricts to convex combinations, MEB does not allow this in an easy way, because of which it is dropped at this place;
- the location of the non-vanishing index set, which can be central as for the K_{approx}-approximation, induced by dimensionality characteristics like OMP, at boundaries as for MEB, which focuses on extremal points.

Table 1
Characteristics of different sparse approximations of prototype based models.

Method	Control of sparsity	Coefficients	Location of exemplars
RSLVQ	No sparsity	Convex	Prototype is central
Sparse training	Soft *Const*	Convex	Not clear
K_{approx}-approx.	Fixed K_{approx}	Set of exemplars	Central
K_{hull}-convex hull	Fixed K_{hull}	Convex	Not clear
OMP	Soft ϵ	Possibly negative	Determined by variance
MEB	Soft ϵ	Convex	Extremal

Table 2
Results of kernel RSLVQ in comparison to k_{NN}-NN, SVM and AP classifiers without preprocessing. The percentage of misclassifications is given. The standard deviation is given in parenthesis. The results for SVM and k_{NN}-NN are taken from [12].

	k_{NN}-NN	SVM	Kernel RSLVQ	AP
Amazon47	16.95(4.85)	75.98(7.33)	**15.37**(0.36)	24.10(0.90)
Aural Sonar	17.00(7.65)	14.25(7.46)	**11.50**(0.37)	31.50(4.00)
Patrol	**11.88**(4.42)	40.73(5.95)	17.50(0.25)	41.90(1.60)
Protein	29.88(9.96)	**2.67**(2.97)	26.98(0.37)	22.90(1.00)
Voting	5.80(1.83)	5.52(1.77)	**5.46**(0.04)	6.50(0.50)

We summarize the characteristics of the methods in Table 1.

We demonstrate the effect of these different characteristics exemplarily in Figs. 2 and 3. In Fig. 1, the result of sparse training is compared to the result of OMP. Obviously, the location of the prototypes is very different which can be attributed to the fact that sparse training influences the prototype locations already while training (Table 2).

In Fig. 2, the location of the exemplars underlying the MEB approximation versus the K_{approx}-approximation is shown in a benchmark. The K_{approx}-approximation tends to locate the exemplars closer to the class centers, while MEB also puts some of the exemplars on extremal positions.

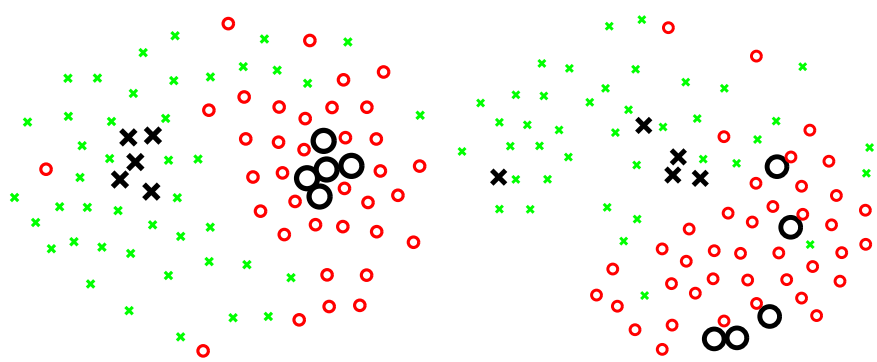


Fig. 1. Aural Sonar with spectrum flip visualized by t-stochastic neighbor embedding [55]. The left figure shows the results of sparse training and the right of OMP. In both settings, the location of the prototypes (not the corresponding exemplars) is shown. Obviously, very different prototype locations are obtained.

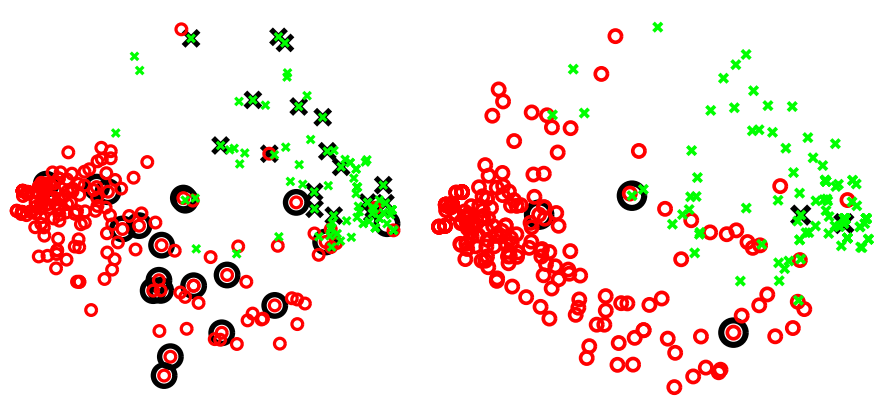


Fig. 2. Voting with spectrum clip visualized by MDS. The left figure shows the results of MEB and the right the results of the 1-approximation. In both cases, the exemplars corresponding to coefficients larger than zero are shown. Obviously, the 1-approximation puts exemplars close to the centers, while MEB also selects boundary positions due to its grounding in an MEB problem.

3.5. A remark on direct exemplar based approaches

When considering sparse prototype approximations, the question occurs whether it is possible to directly learn a sparse prototype model instead of a posterior approximation only. Techniques which represent solutions in terms of prototypical exemplars only, i.e. prototypes $\vec{w}_j = \vec{\xi}_i$ which equal exactly a given data point, have been proposed in prototype-based research under the umbrella of median techniques, see e.g. [14] and references therein. Essentially, this corresponds to the case of a sparse model where the number of exemplars used to represent prototypes is reduced to $K=1$. Recently, a median approach for supervised LVQ has also been proposed [36]. Essentially, median techniques try to devise efficient methods which optimize the given cost function but restricting prototypes to the discrete space formed by the given data.

One problem of such median approaches consists in the fact that their optimization is essentially discrete; hence optimization is either costly, when relying on meta-heuristics for cost function optimization such as simulated annealing or similar, or optimization is prone to local optima due to the very restricted representation abilities in the discrete data space. This effect has been observed in unsupervised median prototype-based methods such as median neural gas in comparison to its continuous relational counterparts, such as relational neural gas, see [24]. Albeit median approaches of this form have quadratic costs only comparable to kernel methods, their performance is often inferior as compared to kernel or relational approaches.

One notable exception of this observation is offered by affinity propagation [18] which rephrases an exemplar based prototype-based clustering scheme in terms of a factor graph representing the data likelihood, for which efficient continuous optimization is possible using message passing algorithms. Hence this technique combines the efficiency of kernel approaches with a direct interpretability of the result by restricting prototypes to exemplars. Still, it is restricted to an unsupervised optimization of the quantization error, such that the obtained classification accuracy is inferior to supervised kernel LVQ approaches, as we will see in experiments.

4. Experiments

We compare kernel RSLVQ and its sparse approximations on a variety of benchmarks as introduced in [12]. Additionally the two more illustrative data sets VBB Midi and Artificial data are investigated, which will be introduced in a later subsection. Thereby, we particularly want to check whether characteristics of the data allow us to infer which approximation is best suited for the given task. The data sets in [12] consist of similarity matrices which are, in general, non-Euclidean. The matrices are symmetrized and normalized before processing. Since the given similarity matrices do not constitute a valid kernel we apply standard preprocessing tools which transfer a given similarity matrix into a valid kernel, as presented e.g. in [12,39]. We test the two transformations *Spectrum clip:* set negative eigenvalues of the matrix to 0. *Spectrum flip:* negative eigenvalues are substituted by their positive values.

- *Amazon47*: This data set consists of 204 books written by 47 different authors. The similarity is determined as the percentage of customers who purchase book *j* after looking at book *i*.
- *Aural Sonar*: This data set consists of 100 wide band solar signals corresponding to two classes, observations of interest versus clutter. Similarities are determined based on human perception, averaging over two random probands for each signal pair.
- *Patrol*: 241 samples representing persons in seven different patrol units are contained in this data set. Similarities are based on responses of persons in the units about other members of their groups.
- *Protein*: 213 proteins are compared based on evolutionary distances comprising four different classes according to different globin families.
- *Voting*: Voting contains 435 samples with categorical data compared by means of the value difference metric. Class labeling into two classes is present.

The eigenvalue spectra of the data are shown in Fig. 3. Obviously, the data differ in the number and characteristic of dimensions which are different from zero. Voting and Protein possess a large

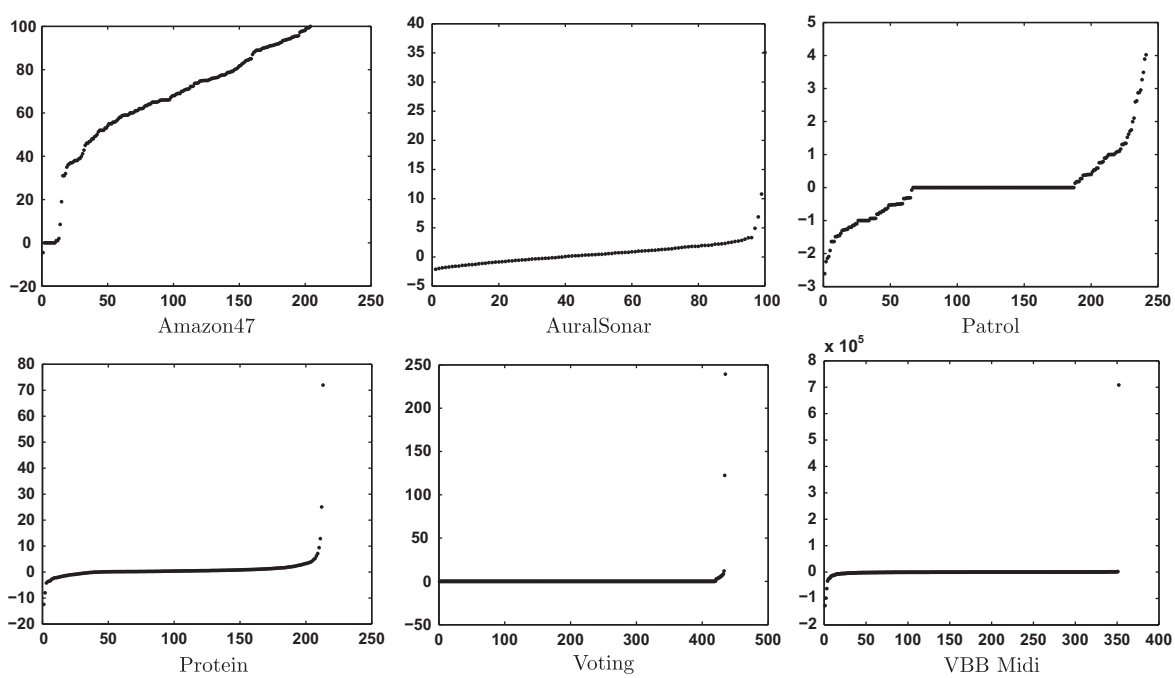


Fig. 3. Characteristic spectrum of the considered similarities. The data sets differ as concerns negative eigenvalues corresponding to non-Euclideanity, and the number of eigenvalues which is different from zero, corresponding to a high dimensional feature space.

number of eigenvalues close to zero, while Amazon47 and Patrol have a significant number of comparably large eigenvalues. Aural Sonar has relatively small but still non-vanishing eigenvalues. Only Amazon47 and Voting are almost Euclidean, for the other data, preprocessing by clip or flip significantly changes the data.

For training, we use the same setting as in [12], we report the results of a 20-fold cross-validation. To judge the overall quality of kernel RSLVQ, we also report the results as obtained with an SVM and a k_{NN}-NN classifier as reported in [12]. The settings for standard RSLVQ are taken from [29] as regards parameters. For comparison, we also report the results of a sparse exemplar-based unsupervised clustering technique equipped with posterior labeling, affinity propagation (AP), which optimizes the classical quantization error by means of a reformulation of this problem as a factor graph and optimization using the max-sum algorithm [18]. See Table 1 for the results. Obviously, kernel RSLVQ is capable of obtaining results which are comparable to SVM or k_{NN}-NN classifiers. Taking supervised information into account improves the result as compared to a fully unsupervised method such as AP in all but one case.

We approximate the solutions of kernel RSLVQ by sparse approximations using the methods as specified above. Thereby, we set the sparsity to $K_{approx}, K_{hull} \in \{1, 10\}$. If training with sparsity constraint is used, an appropriate weighting parameter *Const* is determined by binary search such that a desired sparsity is obtained. The parameter *Const* can be very sensitive depending on the data, leading to non-trivial results in a small range only. For the approximations using OMP and MEB, the quality ϵ of the approximation is determined such that a sparsity in the range of 1–10 is obtained.

4.1. Results as regards sparsity and accuracy

The classification accuracy is shown in Table 3. Interestingly, the obtained classification results when considering sparse approximations differ depending on the data set and the used technique. For the intrinsically low-dimensional data sets, Protein and Voting, different sparse approximations give results comparable to full prototypes, while the situation seems more difficult for the other data sets. For Amazon47, none of the sparse approximations reaches the accuracy of the full model, which can be attributed to a high dimensionality of the data with few data points and a large number of classes. This is a situation where we would possibly expect that the full information of the data set is necessary to obtain a good classification accuracy. For Aural Sonar and Patrol, some sparse techniques yield results comparable to the full models.

It seems that there exists no universally suited method to enforce sparsity. Sparse approximation already while training yields best results in three of the cases. However, the choice of the parameter *Const* is crucial and a high degree of sparsity is not easy to achieve for this setting, as can be seen from the variance of the sparsity as reported in Table 4. In many cases a simple K_{approx}-approximation yields surprisingly good results, indicating that the location of the prototypes can often be well preserved by a simple substitution with its closest exemplar. Besides these observations, one can also detect two cases where the mathematical approximations OMP and MEB yield best results with respect to alternative posterior regularizations, whereby the degree of sparsity is easier to handle as compared to sparse training.

4.2. Results as regards representativity

How can we evaluate the representativity of the obtained prototypes for the given data? Eventually, this question has to be answered by practitioners in the field who inspect the found exemplars. Naturally, the degree of sparsity as reported in Table 4 is a first indicator about the complexity of the resulting model. However, a sparse model does not necessarily correlate with a good classification accuracy, or the representativity of the found exemplars. Here, we investigate two principled ways to access the representativity of the models as a first try to quantitatively measure in how far models could be seen as interpretable.

As a first measure which takes supervised labeling into account, we evaluate Rissanen's minimum description length as introduced in [22]. The minimum description length estimates the

Table 3
Results of kernel RSLVQ and diverse sparse approximations on the investigated benchmark data. The best results (given as percentage misclassifications) of the approximation methods are shown in boldface.

	Kernel RSLVQ	K_{approx}-approx		K_{hull}-convex hull		OMP	MEB	Sparse tr.
		$K_{approx}=1$	$K_{approx}=10$	$K_{hull}=1$	$K_{hull}=10$			
Amazon47								
Clip	15.37	**32.26**	43.82	33.09	55.85	70.12	87.79	39.92
Flip	16.34	**32.32**	46.06	34.18	54.51	68.66	88.54	43.18
Aural Sonar								
Clip	11.25	25.75	14.50	58.50	23.25	15.00	13.50	**10.75**
Flip	11.75	22.75	15.12	61.50	19.75	26.00	**14.75**	15.50
Patrol								
Clip	17.40	39.84	**19.90**	39.17	24.58	29.79	25.42	40.00
Flip	19.48	38.91	**21.03**	40.16	25.52	33.33	24.17	41.56
Protein								
Clip	4.88	18.49	26.94	36.28	27.44	52.09	14.59	**13.84**
Flip	1.40	23.84	24.48	25.35	3.95	49.07	3.72	**2.21**
Voting								
Clip	5.34	8.82	11.39	86.44	82.76	**5.34**	17.70	**5.34**
Flip	5.34	7.99	9.91	86.95	82.53	**5.46**	17.18	5.80
VBB Midi								
Clip	0.00	22.73	21.45	43.75	**14.77**	15.62	17.33	18.18
Flip	0.00	29.55	20.45	38.35	18.47	21.31	17.05	**12.50**
Artificial data								
	0.00	6.67	0.00	33.33	0.00	0.00	0.00	3.33

Table 4
Sparsity (number of non-negative coefficients per prototype and label) of kernel RSLVQ and diverse sparse approximations on the investigated benchmark data. Due to exemplars becoming identical, a sparsity smaller than 1 is possible.

	Kernel RSLVQ	K_{approx}-approx		K_{hull}-convex hull		OMP	MEB	Sparse tr.
		$K_{approx}=1$	$K_{approx}=10$	$K_{hull}=1$	$K_{hull}=10$			
Amazon47								
Clip	3.67	**0.75**	5.28	1.00	3.51	1.96	1.61	1.00
Flip	3.67	**0.75**	5.31	1.00	3.51	1.95	1.60	1.00
Aural Sonar								
Clip	40.00	**0.53**	3.15	1.00	10.00	3.79	5.30	12.75
Flip	40.00	**0.47**	3.07	1.00	10.00	1.28	5.72	12.73
Patrol								
Clip	24.12	**0.68**	4.85	1.00	9.95	6.66	6.93	6.71
Flip	24.12	**0.68**	4.43	1.00	9.95	3.55	6.98	6.69
Protein								
Clip	42.50	**0.47**	3.25	1.00	10.00	1.84	4.89	13.37
Flip	42.50	**0.43**	2.75	1.00	10.00	8.43	4.97	13.52
Voting								
Clip	174.00	**0.29**	2.42	1.00	10.00	11.71	2.16	68.68
Flip	174.00	**0.30**	2.31	1.00	10.00	8.82	1.99	59.92
VBB Midi								
Clip	29.33	**1.00**	10.00	**1.00**	9.92	4.08	7.00	14.42
Flip	29.33	**1.00**	10.00	**1.00**	9.92	1.75	7.25	13.42
Artificial data								
	10.00	1.00	10.00	1.00	10.00	2.00	4.33	4.00

Table 5
Rissanen's minimum description length of kernel RSLVQ and diverse sparse approximations on the investigated benchmark data.

	Kernel RSLVQ	K_{approx}-approx		K_{hull}-convex hull		OMP	MEB	Sparse tr.
		$K_{approx}=1$	$K_{approx}=10$	$K_{hull}=1$	$K_{hull}=10$			
Amazon47								
Clip	151.82	**42.17**	43.39	43.59	44.44	246.80	367.90	252.33
Flip	147.47	**39.49**	43.26	42.66	45.69	416.98	389.68	253.24
Aural Sonar								
Clip	23.30	**4.74**	5.35	15.20	13.84	24.63	24.42	18.98
Flip	21.94	5.21	**4.52**	12.60	13.42	31.31	23.08	16.87
Patrol								
Clip	235.12	35.65	**33.41**	53.63	56.99	274.79	226.68	174.57
Flip	232.20	45.57	**36.71**	56.67	53.40	268.95	229.75	172.31
Protein								
Clip	74.59	**14.83**	16.86	23.25	33.41	208.16	75.35	60.40
Flip	51.42	20.34	20.41	22.91	**18.12**	339.72	49.56	38.34
Voting								
Clip	190.86	**12.25**	12.38	200.83	199.01	75.94	174.90	103.37
Flip	190.89	**15.84**	18.32	181.60	136.44	72.86	183.62	103.16
VBB Midi								
Clip	**18.22**	25.01	21.68	45.43	18.35	18.35	23.24	20.63
Flip	**18.21**	29.60	22.60	42.41	20.41	24.84	23.91	20.46
Artificial data								
	1.47	5.15	1.47	29.83	1.47	3.50	1.78	2.88

number of information it takes to represent the prototypes on the one hand and the errors induced by the prototypes on the data on the other hand. The resulting quantity is depicted in Table 5 for the different sparse approximations. In all cases sparsity clearly yields a more compact representation of the available information as shown by the results reported in Table 5. Further, this measure highlights that simple techniques such as the K_{approx}-approximation seem a good compromise of accuracy and sparsity of the models.

As an unsupervised evaluation measure, we evaluate the entropy of the probability distribution which assigns data to prototypes. To account for different numbers of prototypes, a normalization by its logarithm takes place. Results are depicted in Table 6. The intuition is that a small entropy allows for clearly separated clusters, i.e. representative exemplars, while a large entropy is an indicator for a more uniform distribution. Naturally, the result depends on the cluster structure of the underlying data, indicating e.g. that Voting does not seem to be easily separable into classes with gaps in between the classes. But also within data sets, differences of the different techniques can be found, indicating that the K_{approx}-approximation for $K_{approx}=1$, for example, surprisingly is not able to separate the clusters as well as alternatives.

Table 6
Entropy of kernel RSLVQ and diverse sparse approximations on the investigated benchmark data.

	Kernel RSLVQ	K_{approx}-approx		K_{hull}-convex hull		OMP	MEB	Sparse tr.
		$K_{approx}=1$	$K_{approx}=10$	$K_{hull}=1$	$K_{hull}=10$			
Amazon47								
Clip	3.18	3.99	0.81	4.37	3.16	3.25	3.93	3.98
Flip	2.90	3.66	0.74	4.23	2.90	3.09	3.71	3.86
Aural Sonar								
Clip	3.43	6.03	1.41	1.97	2.85	2.49	2.45	2.30
Flip	1.10	2.23	0.43	1.88	0.82	0.73	0.76	0.73
Patrol								
Clip	3.31	4.81	0.90	3.16	2.93	2.68	2.36	2.28
Flip	2.48	3.62	0.72	3.04	2.17	2.30	1.67	1.95
Protein								
Clip	8.05	13.53	3.20	3.22	6.28	1.94	5.78	7.08
Flip	6.58	11.36	2.98	3.14	5.39	4.71	4.80	5.47
Voting								
Clip	89.86	76.23	56.71	50.06	77.84	80.68	72.14	75.16
Flip	88.40	82.74	57.72	51.37	77.22	84.08	71.23	71.71
VBB Midi								
Clip	9.63	5.64	7.51	5.10	8.90	8.07	6.96	9.02
Flip	5.54	4.34	6.35	3.72	5.04	3.61	3.29	4.48
Artificial data								
	2.22	1.53	2.22	1.70	2.22	2.00	1.57	2.45

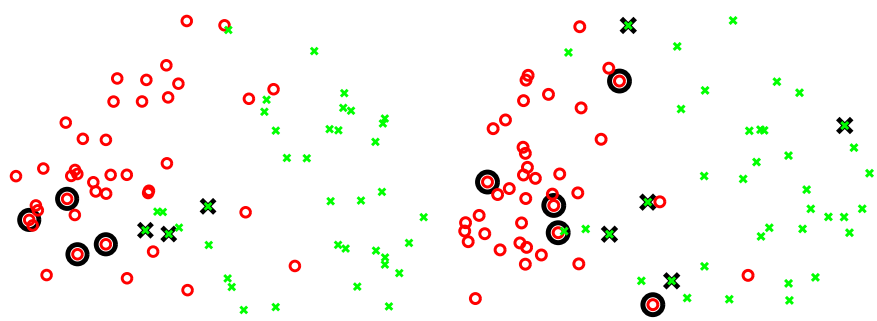


Fig. 4. Aural Sonar with spectrum clip visualized by MDS. The left figure shows the results of 1-approximation and the right of 1-convex hull.

Fig. 4 shows the approximations for extremal values of the entropy in an example data set. The smallest entropy is found in the K_{approx}-approximation setting, whereas most information can be found with K_{hull}-convex hull. Since data points at the border of the data set carry the most information about the location of the whole class it is not surprising that these points get a larger value in the linear combination and give indeed most information about the data set, since they define the borders well. On the other hand the approximated location of the prototypes give more interpretable results, but cannot specify the borders as well, ending in a lower entropy overall.

4.3. Two illustrative examples

The examples as introduced above allow already some insight into the behavior of the techniques, indicating that

- it is not always possible to find sparse solutions of the same quality in particular when data dimensionality is large, but it is possible in many cases,
- for sparse approximations, a simple K-nearest neighbor heuristics seems as appropriate as more fundamental approaches,
- the approximation methods differ in the final location of the exemplars, focusing partially on boundary points rather than central representatives,
- these effects are partially mirrored in measures such as the minimum description length or the entropy.

However, the experiments are in some way preliminary since the involved data are only implicitly given by their pairwise dissimilarities; a direct inspection of the underlying data and its interpretability is problematic. Because of this fact we investigate two further data sets which can directly be inspected: an artificial two-dimensional Euclidean set, and a data set stemming from a transportation system.

- *Artificial data*: Data are randomly generated in two dimensions with 10 data points for each of three classes, see Fig. 5. Since data are Euclidean, we can also directly inspect the prototypes, its approximations, and the exemplars used for the approximation.
- *VBB Midi*: This data set is based on openly accessible public transportation time-tables provided by the Verkehrsverbund Berlin Brandenburg (VBB).[1] As data points we used a subset of 352 train and metro stops in Berlin and defined the distance of two stops as the shortest possible trip between them using the Berlin public transportation system (including bus, train, or

[1] http://daten.berlin.de/datensaetze/vbb-fahrplan-2013

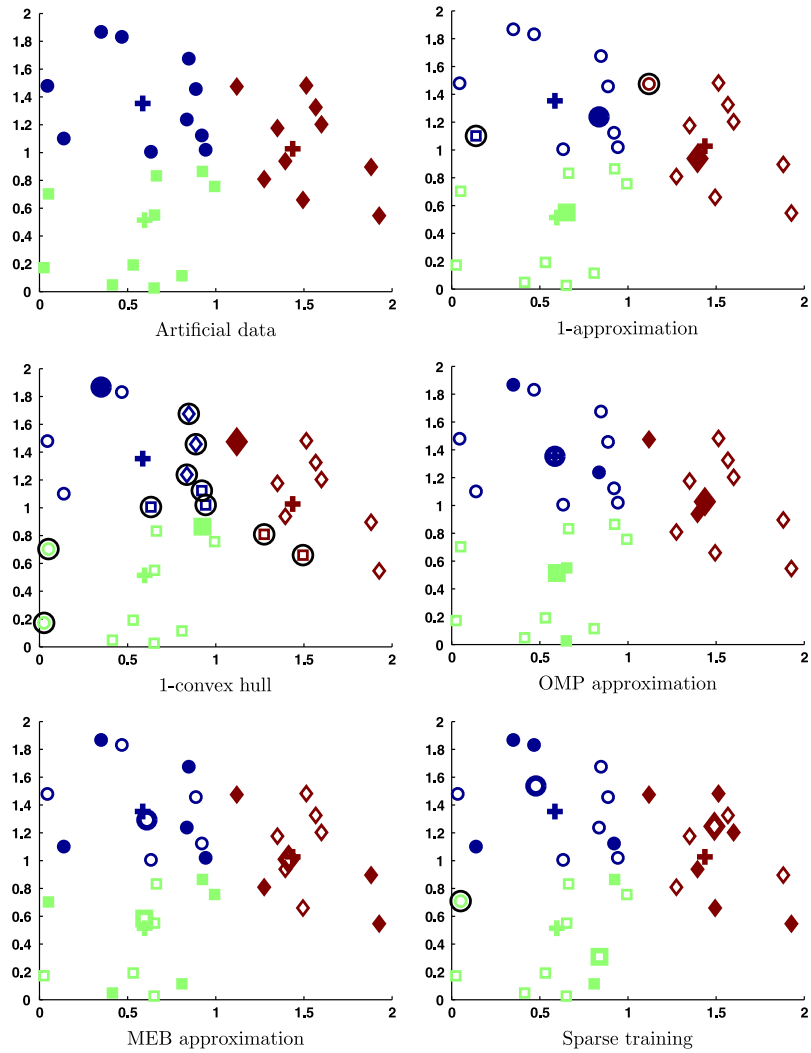


Fig. 5. Two-dimensional artificial data set with prototype locations (crosses) and the respective approximation (big symbols). Note that the approximation is identical to the prototypes for OMP due to the dimensionality of the data. For OMP, MEB, and sparse training, the exemplars used to represent the approximated prototypes are shown via filled symbols. In addition, some prototype approximations cause errors, highlighted by black circles around the misclassified points.

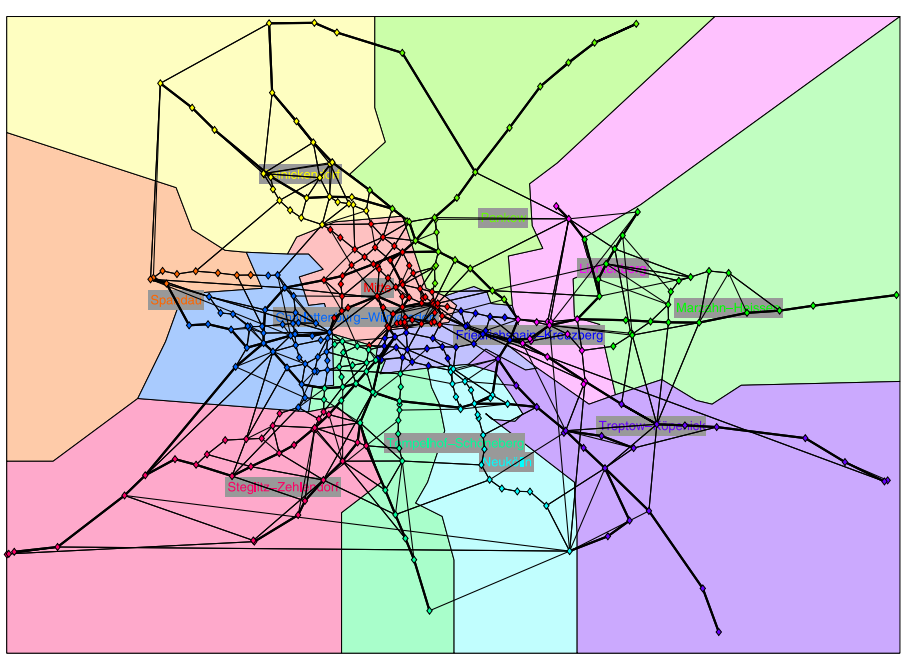


Fig. 6. VBB Midi data set with classes (i.e. districts) marked with different colors. The train, tram, and bus connections are shown and stations correspond to diamonds. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this paper.)

metro). The supervised learning task is generated by using the 12 administrative districts of Berlin as class labels. Data are non-Euclidean (see Fig. 3 for its spectrum) and the distances are preprocessed using clip. See Fig. 6 for the train, metro, and bus lines for the whole area.

Training takes place using one prototype per class and all data points in the training set. The classification results are displayed in Table 3. Interestingly, the classification accuracy is excellent for both data sets provided original kernel RSLVQ is used, while the accuracy deteriorates quite a lot for approximations for the VBB Midi data set due to its high intrinsic dimensionality. In contrast, the artificial data set allows a good approximation of the prototypes, with a drop in accuracy only for the two heuristic approximations. This indicates that more fundamental mathematical methods are better suited to find a close approximation of the prototypes, as can be expected due to the explicit mathematical modeling of the objective. Still, the k approximation gives reasonable results in both cases.

Interestingly, the exemplars which are used to represent the prototypes are qualitatively very dissimilar for the different approximation methods. For the artificial data set, only the 1-approximation searches exemplars from the class centers. All other approximations select exemplars which are located more at the class boundaries. Further, the number of exemplars which are necessary to obtain a good approximation is higher than for

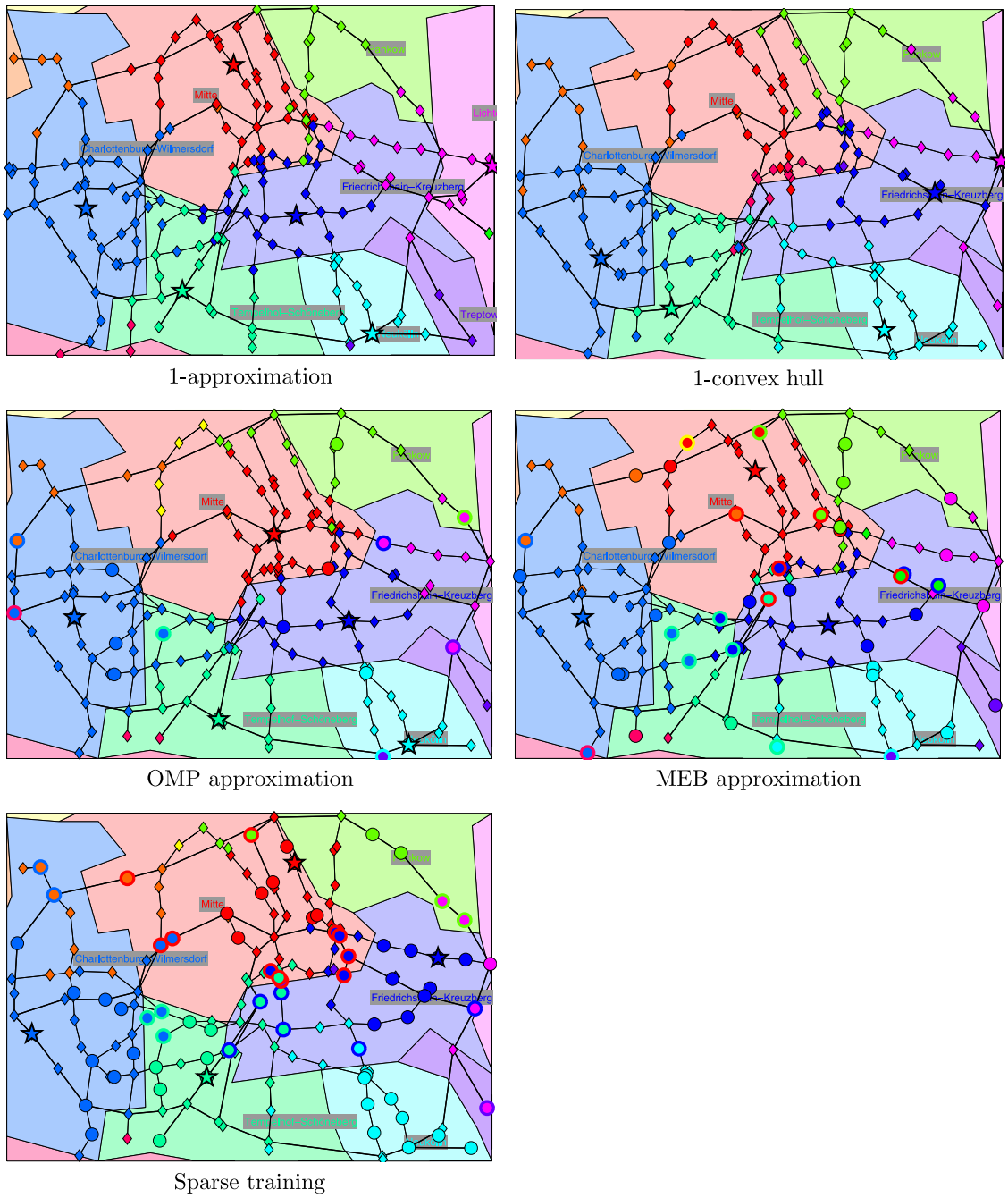


Fig. 7. Central part of the VBB Midi data set with classes (i.e. districts) marked with different colors. Prototypes are represented by their closest exemplar (the data being non-Euclidean), displayed as a star. Further, the exemplars which are used to represent the prototypes, are marked with big circles. Points correspond to diamonds; in addition, train and tram connections are shown, but no bus connections. Misclassifications are indicated by color codes of the stations. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this paper.)

the 1-approximation. A similar conclusion can be drawn in the VBB Midi data set, see Fig. 7, where the central part of the transportation map is displayed. For the 1-approximation, the prototypes and exemplars are located in the center, but distortions are observed for the other techniques. In particular the two techniques based on mathematical optimization, OMP and MEB, put exemplars at the boundaries of the receptive fields, as indicated by the encircled points. Interestingly, the prototypes itself (which are displayed as closest exemplar due to the non-Euclideanity of the data set) are often located at central positions of the traffic map, hence we would expect those to be representative as concerns centrality of the traffic stops. Note that bus lines are not displayed since these are too many. Nevertheless, bus lines often account for short distances of stations in particular at class boundaries, such that misclassifications can easily occur.

5. Discussion

We have proposed different ways to arrive at sparse solutions for kernel RSLVQ schemes, which open the way towards interpretable prototypes also for kernel LVQ. Interestingly, it is indeed possible to obtain sparse representations of high accuracy for all but one data set within a benchmark suite, however, the optimum method varies. Very simple techniques such as an approximation by the closest exemplars seem to work as well as formal optimization approaches as provided by OMP or MEB. The accuracy of MEB and OMP can be better due to their explicit mathematical minimization of the representation error, but they use exemplars located at class boundaries due to the used mathematical formalism. Hence it is not clear whether they are more interpretable: a higher number of exemplars are necessary to describe the class boundaries, while simple heuristics use exemplars at central positions of the classes. We have proposed first quantitative measures to evaluate the usefulness of the results as regards interpretability, relying on Rissanen's minimum description length and the entropy. Alternative principled evaluation techniques as well as their suitability for concrete applications will be the subject of future work.

Acknowledgement

Financial support by the DFG under Grant numbers HA2719/7-1 and HA2719/6-1 and by the CITEC center of excellence funded in the frame of the excellence initiative is gratefully acknowledged.

References

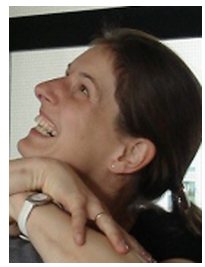
[1] A. Backhaus, U. Seiffert, Quantitative measurements of model interpretability for the analysis of spectral data, in: Proceedings of IEEE SSCI, 2013.
[2] M. Badoiu, K.L. Clarkson, Optimal core sets for balls, in: DIMACS Workshop on Computational Geometry, 2002.
[3] P.L. Bartlett, S. Mendelson, Rademacher and Gaussian complexities: risk bounds and structural results, J. Mach. Learn. Res. 3 (2002) 463–482.
[4] M. Biehl, K. Bunte, P. Schneider, Analysis of flow cytometry data by matrix relevance learning vector quantization, PLOS One 8 (3) (2013).
[5] M. Biehl, A. Ghosh, B. Hammer, Dynamics and generalization ability of LVQ algorithms, J. Mach. Learn. Res. 8 (2007) 323–360.
[6] M. Biehl, B. Hammer, P. Schneider, T. Villmann, Metric learning for prototype-based classification, in: M. Bianchini, M. Maggini, F. Scarselli (Eds.), Innovations in Neural Information Paradigms and Applications, Studies in Computational Intelligence, vol. 247, Springer, Heidelberg, 2009, pp. 183–199.
[7] M. Biehl, B. Hammer, M. Verleysen, T. Villmann (Eds.), Similarity Based Clustering, Springer Lecture Notes Artificial Intelligence, vol. 5400/2009, Springer, Heidelberg, 2009.
[8] C. Bottomley, V. Van Belle, E. Kirk, S. Van Huffel, D. Timmerman, T. Bourne, Accurate prediction of pregnancy viability by means of a simple scoring system, Hum. Reprod. 28 (1) (2013) 68–76.
[9] R. Boulet, B. Jouve, F. Rossi, N. Villa, Batch kernel SOM and related Laplacian methods for social network analysis, Neurocomputing 71 (7–9) (2008) 1257–1273.
[10] A.M. Bruckstein, D.L. Donoho, M. Elad, From sparse solutions of systems of equations to sparse modeling of signals and images, SIAM Rev. 51 (1) (2009) 34–81.
[11] K. Bunte, P. Schneider, B. Hammer, F.-M. Schleif, T. Villmann, M. Biehl, Limited Rank Matrix Learning, discriminative dimension reduction and visualization, Neural Netw. 26 (2012) 159–173.
[12] Y. Chen, E.K. Garcia, M.R. Gupta, A. Rahimi, L. Cazzanti, Similarity-based classification: concepts and algorithms, J. Mach. Learn. Res. 10 (June) (2009) 747–776.
[13] R. Cilibrasi, M.B. Vitanyi, Clustering by compression, IEEE Trans. Inf. Theory 51 (4) (2005) 1523–1545.
[14] M. Cottrell, B. Hammer, A. Hasenfuss, T. Villmann, Batch and median neural gas, Neural Netw. 19 (2006) 762–771.
[15] A. Denecke, H. Wersing, J.J. Steil, E. Korner, Online figure-ground segmentation with adaptive metrics in generalized LVQ, Neurocomputing 72 (7–9) (2009) 1470–1482.
[16] D.L. Donoho, For most large underdetermined systems of linear equations the minimal l1-norm solution is also the sparsest solution, Commun. Pure Appl. Math. 56 (6) (2006) 797–829.
[17] P. Frasconi, M. Gori, A. Sperduti, A general framework for adaptive processing of data structures, IEEE Trans. Neural Netw. 9 (5) (1998) 768–786.
[18] B.J. Frey, D. Dueck, Clustering by passing messages between data points, Science 315 (2007) 972–976.
[19] T. Gärtner, Kernels for structured data (Ph.D. thesis), Univ. Bonn, 2005.
[20] I. Giotis, K. Bunte, N. Petkov, M. Biehl, Adaptive matrices and filters for color texture classification, J. Math. Imaging Vision 20 (2012).
[21] A. Gisbrecht, B. Mokbel, F.-M. Schleif, X. Zhu, B. Hammer, Linear time relational prototype based learning, Int. J. Neural Syst. 22 (5) (2012).
[22] P. Grünwald, The Minimum Description Length Principle, MIT Press, Cambridge MA, 2007.
[23] B. Hammer, A. Gisbrecht, A. Schulz, Applications of discriminative dimensionality reduction, in: Proceedings of ICPRAM, 2013.
[24] B. Hammer, A. Hasenfuss, Topographic mapping of large dissimilarity datasets, Neural Comput. 22 (9) (2010) 2229–2284.
[25] B. Hammer, D. Hofmann, F.-M. Schleif, X. Zhu, Learning vector quantization for similarities, Neurocomputing 131 (2014) 43–51.
[26] B. Hammer, A. Micheli, A. Sperduti, Universal approximation capability of cascade correlation for structures, Neural Comput. 17 (2005) 1109–1159.
[27] B. Hammer, B. Mokbel, F.-M. Schleif, X. Zhu, Prototype Based Classification of Dissimilarity Data, IDA, 2011.
[28] B. Hammer, T. Villmann, Generalized relevance learning vector quantization, Neural Netw. 15 (8–9) (2002) 1059–1068.
[29] D. Hofmann, B. Hammer, Kernel robust soft learning vector quantization, in: ANNPR, 14–23, 2012.
[30] P.J. Ingram, M.P.H. Stumpf, J. Stark, Network motifs: structure does not determine function, BMC Genomics 7 (2006) 108.
[31] S. Kirstein, H. Wersing, H.-M. Gross, E. Korner, A life-long learning vector quantization approach for interactive learning of multiple categories, Neural Netw. 28 (2012) 90–205.
[32] T. Kohonen, Self-Oganizing Maps, 3rd ed., Springer, Heidelberg, 2000.
[33] T. Kohonen, P. Somervuo, How to make large self-organizing maps for nonvectorial data, Neural Netw. 15 (8–9) (2002) 945–952.
[34] C. Lundsteen, J. Phillip, E. Granum, Quantitative analysis of 6985 digitized trypsin g-banded human metaphase chromosomes, Clin. Genet. 18 (5) (1980) 355–370.
[35] B. Mokbel, A. Hasenfuss, B. Hammer, Graph-based representation of symbolic musical data, in: GbRPR, vol. 42–51, 2009.
[36] David Nebel, Barbara Hammer, Thomas villmann: a median variant of generalized learning vector quantization, in: ICONIP, vol. 2, 2013, pp. 19–26.
[37] B.A. Olshausen, D.J. Field, Emergence of simple-cell receptive field properties by learning a sparse code for natural images, Nature 381 (1996) 607–609.
[38] M. Olteanu, N. Villa-Vialaneix, M. Cottrell, On-line relational SOM for dissimilarity data, in: CoRR, abs/1212.6316, 2012.
[39] E. Pekalska, R.P. Duin, The Dissimilarity Representation for Pattern Recognition. Foundations and Applications, World Scientific, Singapore, 2005.
[40] O. Penner, P. Grassberger, M. Paczuski, Sequence alignment, mutual information, and dissimilarity measures for constructing phylogenies, PLOS One 6 (1) (2011).
[41] A.K. Qin, P.N. Suganthan, Kernel neural gas algorithms with application to cluster analysis, in: Proceedings of the 17th International Conference on Pattern Recognition, 2004, pp. 617–620.
[42] A.K. Qin, P.N. Suganthan, A novel kernel prototype-based learning algorithm, in: Proceedings of the 17th International Conference on Pattern Recognition, 2004.
[43] J. Rissanen, Modeling by the shortest data description, Automatica 14 (1978) 465–471.
[44] H. Ruiz, I.H. Jarman, P.J.G. Lisboa, S. Ortega-Martorell, A. Vellido, E. Romero, J.D. Martin, Towards interpretable classifiers with blind signal separation, in: Proceedings of the International Joint Conference on Neural Networks, 2012.
[45] A. Sato, K. Yamada, Generalized learning vector quantization, in: NIPS, 1995.
[46] F. Scarselli, M. Gori, A.C. Tsoi, M. Hagenbuchner, G. Monfardini, Computational capabilities of graph neural networks, IEEE Trans. Neural Netw. 20 (1) (2009) 81–102.
[47] F.-M. Schleif, T. Villmann, B. Hammer, P. Schneider, Efficient kernelized prototype based classification, Int. J. Neural Syst. 21 (6) (2011) 443–457.

[48] P. Schneider, M. Biehl, B. Hammer, Distance learning in discriminative vector quantization, Neural Comput. 21 (2009) 2942–2969.
[49] P. Schneider, M. Biehl, B. Hammer, Adaptive relevance matrices in learning vector quantization, Neural Comput. 21 (2009) 3532–3561.
[50] S. Seo, K. Obermayer, Soft learning vector quantization, Neural Comput. 15 (2003) 1589–1604.
[51] N.Z. Shor, Minimization Methods for Non-differentiable Functions. Springer Series in Computational Mathematics, Springer, Heidelberg, 1985.
[52] I.W. Tsang, J.T. Kwok, P.-M. Cheung, Core vector machines: fast SVM training on very large data sets, J. Mach. Learn. Res. 6 (2005) 363–392.
[53] V. Van Belle, P. Lisboa, Automated selection of interaction effects in sparse kernel methods to predict pregnancy viability, in: Proceedings IEEE CIDM, 2013.
[54] V. Van Belle, B. Van Calster, D. Timmerman, T. Bourne, C. Bottomley, L. Valentin, P. Neven, S. Van Huffel, J. Suykens, S. Boyd, A mathematical model for interpretable clinical decision support with applications in gynecology, PLoS One 7 (3) (2012).
[55] L. van der Maaten, G. Hinton, Visualizing high-dimensional data using t-SNE, J. Mach. Learn. Res. 9 (2008) 2579–2605.
[56] A. Vellido, J.D. Martin-Guerroro, P. Lisboa, Making machine learning models interpretable, in: ESANN'12, 2012.
[57] H. Yin, On the equivalence between kernel self-organising maps and self-organising mixture density networks, Neural Netw. 19 (6–7) (2006) 780–784.

Daniela Hofmann received her Diploma in Computer Science from the Clausthal University of Technology, Germany. Since early 2012 she is a PhD student at the Cognitive Interaction Technology Center of Excellence at Bielefeld University, Germany.

Frank-Michael Schleif received his Ph.D. in Computer Science from the University of Clausthal, Germany, in 2006. From 2004 to 2006 he was working for the R&D Department at Bruker Biosciences. From 2006 to 2009 he was a research assistant in the research group of computational intelligence at the University of Leipzig working on multiple bioinformatic projects. In 2010 he joined the chair of theoretical computer science and did research in multiple projects in machine learning and bioinformatics. From 2014 he will be a member of the University of Birmingham, UK as a Marie Curie Fellow. His areas of expertise include machine learning, signal processing, data analysis and bioinformatics. Several long term research stays have taken him to UK, the USA, the Netherlands and Japan. He is the co-editor of the Machine Learning Reports and reviewer for multiple journals and conferences in the field of machine learning and computational intelligence. He is a founding member of the Institute of Computational Intelligence and Intelligent Data Analysis (CIID) e.V. (Mittweida, Germany), a member of the GI, the DAGM and secretary of the German chapter of the ENNS (GNNS). He is coauthor of more than 70 papers in international journals and conferences on different aspects of Computational Intelligence, most of which can be retrieved from http://www.techfak.uni-bielefeld.de/~fschleif/.

Benjamin Paassen recieved his Bachelors degree in Cognitive Informatics from the Bielefeld University, Germany. Since 2012 he is employed as scientific assistant in the DFG funded research project "Learning Feedback in Intelligent Tutoring Systems" at the Cognitive Interaction Technology Center of Excellence at Bielefeld University, Germany.



Barbara Hammer received her Ph.D. in Computer Science in 1995 and her venia legendi in Computer Science in 2003, both from the University of Osnabrueck, Germany. From 2000 to 2004, she was leader of the junior research group 'Learning with Neural Methods on Structured Data' at University of Osnabrueck before accepting an offer as Professor for Theoretical Computer Science at Clausthal University of Technology, Germany, in 2004. Since 2010, she is holding a professorship for Theoretical Computer Science for Cognitive Systems at the CITEC cluster of excellence at the Bielefeld University, Germany. Several research stays have taken her to Italy, UK, India, France, the Netherlands, and the USA. Her areas of expertise include hybrid systems, self-organizing maps, clustering, and recurrent networks as well as applications in bioinformatics, industrial process monitoring, or cognitive science. She is currently leading the IEEE CIS Technical Committee on Data Mining, and the Fachgruppe Neural Networks of the GI.