Sparse Prototype Representation by Core Sets

Frank-Michael Schleif, Xibin Zhu, and Barbara Hammer

CITEC Centre of Excellence, Bielefeld University, 33615 Bielefeld, Germany
fschleif@techfak.uni-bielefeld.de

Abstract. Due to the increasing amount of large data sets, efficient learning algorithms are necessary. Also the interpretation of the final model is desirable to draw efficient conclusions from the model results. Prototype based learning algorithms have been extended recently to proximity learners to analyze data given in non-standard data formats. The supervised methods of this type are of special interest but suffer from a large number of optimization parameters to model the prototypes. In this contribution we derive an efficient core set based preprocessing to restrict the number of model parameters to $O(\frac{n}{\epsilon^2})$ with n as the number of prototypes. Accordingly, the number of model parameters gets independent of the size of the data sets but scales with the requested precision ϵ of the core sets. Experimental results show that our approach does not significantly degrade the performance while significantly reducing the memory complexity.

1 Introduction

Modern measurement technologies e.g. in the life sciences provide large data sets with multiple sources of domain knowledge, which can be used to optimize the analysis of the data. Accordingly methods which can integrate domain knowledge and are interpretable are of special interest [1]. Prototype algorithms are interpretable models and have been recently extended to simplify domain knowledge integration in different ways. They represent their decisions in terms of typical representatives contained in the input space or by approximations thereof. Prototypes can directly be inspected by human experts in the same way as data points: for example, physicians can inspect prototypical medical cases, prototypical images can directly be displayed on the computer screen, prototypical action sequences of robots can be performed in a robotic simulation, etc. Since the decision in prototype-based techniques usually depends on the similarity of a given input to the prototypes stored in the model, a direct inspection of the taken decision in terms of the responsible prototype becomes possible.

Different algorithms have been proposed to obtain supervised and unsupervised prototype models. One key element is the representation of the data, either by an appropriate metric [2] or by using similarities or dissimilarities as obtained from domain specific measures of proximities. In the latter case the data are represented by an $N \times N$ matrices $\mathcal{P}$ of proximity measures, with N as the number of samples and the prototypes are modeled using a $n \times N$ coefficient matrix, with n as the number of prototypes.

If $\mathcal{P}$ consists of metric similarities, later denoted as $\mathcal{S}$ [1] , we have a kernel matrix and can use standard kernel classifiers like the Support Vector Machine (SVM) [3] or the

[1] If $\mathcal{P}$ are dissimilarities, as e.g. obtained from Euclidean distances we denote the matrix by $\mathcal{D}$.

H. Yin et al. (Eds.): IDEAL 2013, LNCS 8206, pp. 302–309, 2013.

Probabilistic Classifier Vector Machine (PCVM) [4]. But, the SVM type algorithms determine models which are not interpretable because the model parameters are extreme cases of the data and hence less interpretable by the domain expert [5]. The PCVM is better to interpret but has an initial runtime complexity of $O(N^3)$ which scales down during learning to $O(m^3)$ with m as the number of effectively used basis functions, using sparsity constraints [4]. The memory complexity of PCVM scales in a similar way from $O(N^2)$ to $O(m^2)$ during the optimization. For proximity data supervised prototype based learning algorithms have in general a runtime complexity of $O(n^2 \cdot N)$ using the recent approximation strategies [6] and a memory complexity of $O(n \cdot N)$, with $n \ll N$. In this paper we will derive a strategy to reduce this memory complexity to $O(\frac{n}{\epsilon^2})$, independent of the number of samples, which obviously also improves the practical runtime for larger data sets, although the theoretical runtime complexity lasts with $O(N)$. We will achieve this by using the concept of core sets [7], already used successfully in supervised learning in [8]. First we will give a short introduction to prototype based learning for proximity data and highlight current problems in this field. Subsequently, we give a brief review of core sets and link both approaches. In the experiments we show that the proposed method is efficient compared with the standard strategies.

2 Prototype Based Learning

Prototype based learning is a very generic approach (see e.g. [9]) with good generalization properties [10]. The prototypes, as the main model parameters are derived as representants of the quantized data space. Accordingly the models are sparse. More recent extension of prototype based approaches for proximity data, represent the prototypes by means of linear combinations of the original data [11], where the models become often very dense [5].

Assume data $\mathbf{x}_i \in \mathbb{R}^D, i = 1, \ldots, N$, are given in a D dimensional space. Prototypes are elements $\mathbf{w}_j \in \mathbb{R}^D, j = 1, \ldots, K$, of the same space with $W = \{\mathbf{w}_1, \ldots, \mathbf{w}_K\}$, the set of all prototypes of the model. They decompose data into receptive fields $R(\mathbf{w}_j) = \{\mathbf{x}_i : \forall k \; d(\mathbf{x}_i, \mathbf{w}_j) \leq d(\mathbf{x}_i, \mathbf{w}_k)\}$ based on some metric distance measures, like the squared Euclidean distance $d(\mathbf{x}_i, \mathbf{w}_j) = \|\mathbf{x}_i - \mathbf{w}_j\|^2$. The goal of prototype-based machine learning techniques is to find prototypes which represent a given data set as accurately as possible. We will also assume that the data $\mathbf{x}_i$ are equipped with prior class labels $c(\mathbf{x}_i) \in \{1, \ldots, L\}$ in a finite set of priorly known classes in the crisp case or can be associated to a vector of class assignments $l(\mathbf{x}_i) \in [0, 1]^L$, normalized such that $\sum_k^L l(\mathbf{x}_i)^k = 1$ which we will call soft or fuzzy labels. One of the most popular approaches for supervised prototype learning is the Robust Soft LVQ [12], a large margin classifiers [10], optimizing a cost function using gradient descend.

For proximity data an extended RSLVQ was proposed in [13]. One assumes $w_j = \sum_l \alpha_{jl} x_l$ where $\alpha_{jl} > 0$ with $\sum_l \alpha_{jl} = 1$. Then, we can compute for a given data point x_i: $\|x_i - w_j\|^2_{pq} = s_{ii} - 2\sum_l \alpha_{jl} s_{il} + \sum_{ll'} \alpha_{jl}\alpha_{jl'} s_{ll'}$ Hence we can compute distances of all data points and prototypes based on pairwise data *similarities*: $s_{ij} \in \mathcal{S}$ only, in quadratic time. Further, we do not need to represent prototypes w_j explicitly, rather, the coefficients α_{jl} are sufficient. Similarly, we find $\|x_i - w_j\|^2_{pq} = \sum_l \alpha_{jl} d_{il} - 1/2 \cdot \sum_{ll'} \alpha_{jl}\alpha_{jl'} d_{ll'}$ provided $\sum_l \alpha_{jl} = 1$. Hence, as an alternative, we can compute distances via all pairwise *dissimilarities*: $d_{ij} \in \mathcal{D}$ of data in quadratic time. In the following

we will focus on similarities only, but extensions to metric dissimilarities are straight forward using techniques of [14].

The coefficient matrix Γ, with α_{jl} being the l-th coefficient of the j prototype, has most often $n \times N$ entries and is densely populated in the final model. Strategies to improve this by explicit sparsity concepts have been proposed in [5,13] but introduced additional parameters and are hard to control. Also these techniques are often not applicable in the first steps of the optimization such that, similar like for PCVM the initial complexity is high and a large number of parameters needs to be estimated. There are two issues, one is the potentially large squared proximity matrix $\mathcal{P}$ and the second is the dependent matrix Γ. In [15] a concept, known as core sets was used to approach the first problem, we will use similar ideas to approach the second problem, an issue for prototype based models, due to the explicit modeling of the prototypes by data points.

3 Core Sets

The notion of core-sets appears in solving the approximate minimum enclosing ball (MEB) problem in computational geometry [7].

Definition 1 (Minimum enclosing ball problem). *Let* $\{\mathbf{v}_1, \ldots, \mathbf{v}_m\} \in \mathbb{R}^D$ *a set of points. The objective is to find a minimum enclosing ball with radius* R *and center point* $\mathbf{c}$ *such that* $\|\mathbf{c} - \mathbf{v}_i\|^2 \leq R^2 \; \forall i$. *Hence* $B(c, R) = \{\mathbf{v} | R \geq \|c - \mathbf{v}\|, \mathbf{v} \in V\}$.

As shown in [15], the MEB problem can be equivalently express as a quadratic dual optimization problem: $MEB = \min_{\alpha_i \geq 0, \sum \alpha_i = 1} \alpha K \alpha^\top - \sum_i \alpha_i K(i,i)$, with K the kernel matrix defined on V and $\mathbf{c}$ and $\mathbf{v}$ represented in a kernel space as shown before. The radius R is obtained by solving $R = \sum_i \alpha_i K(i,i) - \alpha K \alpha^\top$. The center is used only in the distance calculation which can be expressed using the kernel trick and is a linear combination of the training points based on the obtained α-vector, $\mathbf{c} = \sum_{i=1} \alpha_i \phi(\mathbf{v}_i)$.

Definition 2 (Core set). *Let* $S = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\} \subseteq V$, *then* $Q \subset S$ *is a core set, if* $S \subset B(c, (1+\epsilon)R)$ *and* $B(c, R) = MEB(Q)$.

An encouraging property of core-sets is that the number of elements in it is independent of the data dimensionality and size [7]. Obviously the notion of a MEB ball is closely related to that of a prototype. The prototype, represents a set of points as a receptive field, which borders are constrained by the other prototypes, whereas the MEB represents a set of points, restricted by the radius R and the center position c. The center of the MEB and the prototype position, defined on the same set of points, may not coincide. Nevertheless, a MEB around a set of points in a receptive field of a prototype is often a good approximation for this field.

4 Prototype Models by Core Sets

The idea is to represent a prototype w_j in the matrix Γ by those coefficients α_{jl} which are used to model the MEB of this set of points. This approximation is reasonable as long as the receptive field is close to a sphere, in the similarity space. This is also an

implicit assumption of prominent prototype learning algorithms like the original Robust Soft LVQ [12], where the prototypes are actually modeled as Gaussians with equal bandwidth. We will derive our approach as a pre-processing step prior to any prototype model. In general prototype learning algorithms are initialized, class wise by unsupervised clustering approaches [12], using e.g. k-means or soft competitive learning (SCL) [16] with the number of prototypes per class specified by the user as a meta parameter. For large scale problems novel approximations of k-means or SCL are available, like approximate and core k-means [17,18] or fast SCL [19]. We will focus on the latter one, which is a batch methods using quasi-newton optimization and is efficient also for $N \geq 1e6$, assuming a small number of clusters n. In each case we assume that the training data have been already partitioned by an unsupervised clustering approach into disjunct subsets. Our approach is summarized in Alg. 1

Algorithm 1. Prototype representation by core sets.

1: **init:**
2: Let T be a set of labeled training points and Γ an empty matrix $n \times N$
3: $\epsilon = 0.1$ ▷ Fixed approximation error of the MEB
4: Let $Z = \{z_1, \ldots, z_n\}$ a set of classwise clustered training data
5: $z_j = \{(x_1, l_1)^j, \ldots, (x_k, l_k)^j\}$ using e.g. kmeans
6: **for all** $z_j \in Z$ **do**
7: $[\alpha_j, S_j] := MEB(z_j, \epsilon)$
8: ▷ α-values of the MEB solution for cluster j, S_j core set indices w.r.t. T
9: $\Gamma_j := \alpha_j$
10: ▷ Represent prototype j by the MEB solution for cluster j at indices S_j
11: **end for**
12: Γ := Simplify(Γ) ▷ Reduce Γ to non-empty columns
13: **return** Γ;

The MEB algorithm used in line 7 of Alg. 1 is the same as used in [8]. Starting with two random initial points the MEB is increased in each iteration by the point which is farthest away until the set is covered by the MEB within a circle of $(1+\epsilon) \times R$. A cluster is described by the α values as obtained from the MEB. In general the MEB needs only few iterations such that the number of core set points for a cluster is often in the range of $1 - 10$, theoretically it is bounded by $O(\frac{1}{\epsilon^2})$.

To identify the point which is farthest away in the MEB algorithm, probabilistic sampling is employed [8]. Assume we have a ranking (rank) of the distances of all points to the current center c. We now choose the set F as the 5% points which are furthest away from c, or have a high rank. The probabilistic sampling strategy ensures that a selection of only 59 random points from the training data will contain a point from F with a probability of 95%. The core set S is typically very small $|S| \ll N$ such that the runtime complexity of the calculation of the quadratic optimization problem in the MEB's in Algorithm 1 is given by $O(|S_j|^2)$. For the addressed scale of N we will consider this complexity as sufficient. For very large N it would also be possible to avoid the calculation of a MEB if we know the expected radius r for our problem.

This could be done following concepts of [20] using an enclosing ball concept, but we will not discuss this in the following.

Obviously, the Γ matrix needs not to be defined in advance and may grow during the steps of Algorithm 1 if we add $|S_j| = q_j$, columns to Γ for each prototype because all z_j are disjunct. The obtained matrix Γ can subsequently be used in any prototype based classifier for proximity data as an initial model. Using e.g. a supervised prototype classifier the prototypes can be further optimized to improve the class discrimination. For a prototype j this can be done by modifying the coefficients in the j row of the matrix Γ. As will be exemplary shown in the following this can be done by adapting not only the already used coefficients $\Gamma_{j\cdot}$ but also those not yet used to represent of the prototype due to the disjunct initialization. As an example we consider a checker board problem with 4×4 clusters in $2 - D$. We represent these data by an Euclidean inner product kernel and define two multi-modal classes as shown in Figure 1, such that one class has 5 clusters and the other 11 clusters. The number of prototypes is defined as 7 per class, which is (theoretically) sufficient to discriminate between the classes, using some boundary effects of the data. Since the Γ matrix is initialized class wise and the number of prototypes is 7 per class, the larger class is underrepresented.

The initial post-labeling accuracy is 90%, using the prototype learner presented in [19] we adapt the positions of the prototypes and also the labeling of the prototypes. After 100 iterations we obtain an optimized prototype representation as shown in Figure 1. We observe that the smaller, over represented class, has lost prototypes to the under represented class. These prototypes now show new non-vanishing coefficients in Γ for the indices of the new class they have been assigned to and have also changed their class labeling. The final accuracy is 99% showing that the core-set representation coupled with supervised post training has sufficient representation power.

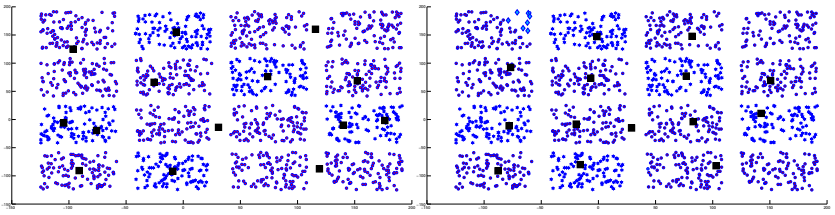

Fig. 1. Checker board data: initial model (left) and after optimization (right) using core sets and a kernel RSLVQ formulation. The top plot shows the data with the given labeling, the right plot the predicted labels on the final model. Initially both classes are modeled by 7 prototypes. The class labels are denoted by different shapes and colors. The prototypes are black squares. Using the initial model a misclassification of 10% is obtained (not illustrated). In the right figure one obtains an accuracy of 99%, few misclassifications can be observed shown as cyan $\diamondsuit$.

5 Experiments

We now show the efficiency of the approach at different data sets, represented by metric proximities, given as a kernel. The number of prototypes was defined such that a reasonable modeling of the data can be achieved guided by the number of classes and prior

Table 1. Experiments with different kernels using the core approach and random selection. Prediction accuracy (Acc), runtime (RT) and memory complexity (used points in Γ (Pts))

Data set	Core			$k\%$			10%		
	Acc	RT	Pts	Acc	RT	Pts	Acc	RT	Pts
Checker	$99.06 \pm 0.52\%$	54.68s	29.4 ± 1.1	66.21 ± 0.94	9.36	31	$99.42 \pm 0.7\%$	74.3s	7200
MNIST	$86.81 \pm 0.5\%$	241s	731.8 ± 3.3	86.13 ± 0.09	113.45	732	$90.37 \pm 1.0\%$	513s	5600
USPS	$87.68 \pm 1\%$	101.6s	714 ± 5.1	$87.51 \pm 1.3\%$	80.9s	714	$88.18 \pm 0.6\%$	95.22s	880
SPAM	$87.48 \pm 2\%$	15.23s	31 ± 1.2	$84.66 \pm 4.1\%$	6.91s	31	$89.72 \pm 0.4\%$	10.71s	369
Phoneme	$83.20 \pm 3.7\%$	14.62s	83.4 ± 0	$75.52 \pm 8.4\%$	9.02s	84	$84.57 \pm 4.0\%$	14.26s	293
Image	$78.62 \pm 1.7\%$	8.2s	27 ± 1.7	$77.7 \pm 1.6\%$	$4.4s$	27	$80.73 \pm 3.9\%$	4.79s	167

experiments of the authors [19]. Please note, that the measure of prediction accuracy on the test data is only used to demonstrate the stability of our approach, we did not do any meta-parameter tuning, although the obtain results are already competitive.

The MNIST data [2] contain $70000, 784$-dimensional binary images from 10 digit classes which have been processed by a neural kernel $k(\mathbf{v_i}, \mathbf{v_j}) = \tanh(av_i^\top v_j + b)$ with $a = 0.0045$ and $b = 0.11$ acc. to [17], we used 10 prototypes per class.

The Checkerboard data is an artificial data set with 90000 $2-$dimensional samples organized on a 3×3 grid with alternating labels in two classes, 5 prototypes have been used per class with a linear kernel to provide the similarities. By construction this data set is highly multimodal.

For all other data sets we use an elm kernel [21], which is a defacto parameter free RBF kernel. USPS [3] contain $11000, 256$-dimensional character feature vectors from 10 classes analyzed by 10 prototypes per class. The SPAM database [4], contain $4601, 57$-dimensional feature vectors in 2 classes which have been modeled by 2 prototypes per class. Further we used the phoneme data set with 3656 points in 20 dimensions with 13, slightly imbalanced classes. For this data set 1 prototype per class was taken. As another example we used the Image data set with 2086 sample in 18 dimensions and 2 classes, modeled by 2 prototypes per class and taken from [22].

All initial clusterings were obtained by fast SCL [19] with a runtime complexity of $O(n^3 \cdot N)$. We evaluate the runtime and memory complexity using our presented core-set approach (core), an alternative representation by selecting a random subset of samples for the prototype modeling with respect to 10% of the data and with $k\%$ where, k is set to get a similar complexity as within the core-set model. As the final prototype classifier we use a full probabilistic formulation of kernel RSLVQ [13,23]. All experiments are done within a $5-$ fold crossvalidation.

The results are shown in Table 1 for multiple data sets, with different data characteristics (number of classes, number of samples, data dimensionality). We also experimented with a different number of columns of the Γ matrix, or different random subsets of representation points, to obtain a similar accuracy than by the core-set approach. The minimal subset size for the Checker data is with around 5% leading to an accuracy of 94% with 301 representation points. For the Phoneme data 5% representation points are

[2] `http://yann.lecun.com/exdb/mnist/`
[3] `http://www.cs.nyu.edu/~roweis/data.html`
[4] `http://archive.ics.uci.edu/ml/datasets`

necessary to get 83% with 147 points. For the MNIST, USPS, SPAM and the Image data a random subset at $k\%$ is already sufficient.

As a general result we find that the proposed approach (*core* column of Table 1) shows similar good prediction results compared to a large model[5], shown in the 10% column. As expected the number of representation points, defined by the core set approach is in general very small, with one exception, for the USPS data where the core-set model is only 20% smaller than the largest considered set.

If we compare our results with a randomly chosen representation set of the same size as the core-set model, shown in column $k\%$, we found that in all cases the core-set model has an equivalent, or in parts substantial better prediction accuracy. For some data sets and using a random selection strategy, the number of representation points has to be substantially increased by a factor of 10 to achieve a similar accuracy.

Depending on the data set a random representation set may substantially degenerate the accuracy of the final model, as shown for the Checker and the Phoneme data set. This can be potentially be explained by the multimodality of these data sets. With respect to the experiments the proposed approach is a systematic and effective strategy to sample a representation set from the training data. For larger data sets a naive random sampling, constrained by the available memory can fail and the obtained models are unnecessary complicated and hard to interpret. Without the information, available here by the core-set approach, it would not easy to guess a reasonable value for $k\%$ but we would rely on adhoc decision like 1% or 10% of the data upper-bounded by the available memory. Using the core-set approach the necessary memory to achieve a very good model representation is $5-10$ times smaller for the considered data than using a adhoc decision for the considered data. Theoretically the memory complexity scales with $O(\frac{n}{\epsilon^2})$, independent of N, such that for very large data sets our strategy is even more interesting.

6 Conclusions

Prototype based models for proximity data are complex and less interpretable for large data sets. This is due to the large coefficient matrix Γ, representing the prototypes by linear combination of original training points. The presented technique automatically identifies a small set of representation points for the modeling of proximity prototypes. The approach achieves competitive prediction accuracy with respect to more complex prototype models. The number of representation points is automatically selected and is independent of the number of training points. The obtained model is not only sparse, with low memory requirements, but also accelerates the model calculation. This effect is most pronounced for large data. The technique can be effectively used to obtain sparse prototype models also for very large proximity data.

Acknowledgements. This research and development project is funded by the German Federal Ministry of Education and Research (BMBF) within the Leading-Edge Cluster Competition and managed by the Project Management Agency Karlsruhe (PTKA). The author is responsible for the contents of this publication. Additional funding was provided by the Cluster of Excellence 277 CITEC funded in the framework of the German Excellence Initiative.

[5] A full model, using N points is often too costly, but some experiments with larger representation sets did not significantly improve the accuracy

References

[1] Belle, V.V., Lisboa, P.J.G.: Research directions in interpretable machine learning models. In: Proc. of ESANN 2013 (2013)

[2] Biehl, M., Hammer, B., Schneider, P., Villmann, T.: Metric learning for prototype-based classification. In: Bianchini, M., Maggini, M., Scarselli, F., Jain, L.C. (eds.) Innovations in Neural Information Paradigms and Applications. SCI, vol. 247, pp. 183–199. Springer, Heidelberg (2009)

[3] Vapnik, V.: The nature of statistical learning theory. Statistics for engineering and information science. Springer (2000)

[4] Chen, H., Tino, P., Yao, X.: Probabilistic classification vector machines. IEEE Transactions on Neural Networks 20(6), 901–914 (2009)

[5] Schleif, F.M., Villmann, T., Hammer, B., Schneider, P.: Efficient kernelized prototype-based classification. Journal of Neural Systems 21(6), 443–457 (2011)

[6] Gisbrecht, A., Mokbel, B., Schleif, F.M., Zhu, X., Hammer, B.: Linear time relational prototype based learning. Journal of Neural Systems (2012) (in press)

[7] Badoiu, M., Har-Peled, S., Indyk, P.: Approximate clustering via core-sets. In: STOC, pp. 250–257 (2002)

[8] Tsang, I.H., Kocsor, A., Kwok, J.Y.: Large-scale maximum margin discriminant analysis using core vector machines. IEEE TNN 19(4), 610–624 (2008)

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

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

[11] Hammer, B., Hasenfuss, A.: Topographic mapping of large dissimilarity data sets. Neural Computation 22(9), 2229–2284 (2010)

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

[13] Hammer, B., Hoffmann, D., Schleif, F.M.: Learning vector quantization for (dis-)similarities. NeuroComputing (in press, 2013)

[14] Schleif, F.-M., Gisbrecht, A.: Data analysis of (Non-)Metric proximities at linear costs. In: Hancock, E., Pelillo, M. (eds.) SIMBAD 2013. LNCS, vol. 7953, pp. 59–74. Springer, Heidelberg (2013)

[15] Tsang, I.W., Kwok, J.T., Cheung, P.M.: Core vector machines: Fast svm training on very large data sets. Journal of Machine Learning Research 6, 363–392 (2005)

[16] Martinetz, T., Schulten, K.: Topology representing networks. Neural Networks 7(3), 507–522 (1994)

[17] Chitta, R., Jin, R., Havens, T., Jain, A.: Approximate kernel k-means: Solution to large scale kernel clustering, pp. 895–903 (2011)

[18] Har-Peled, S., Kushal, A.: Smaller coresets for k-median and k-means clustering. Discrete & Computational Geometry 37(1), 3–19 (2007)

[19] Schleif, F.M., Zhu, X., Gisbrecht, A., Hammer, B.: Fast approximated relational and kernel clustering. In: Proceedings of ICPR 2012, pp. 1229–1232. IEEE (2012)

[20] Tsang, I.W., Kocsor, A., Kwok, J.T.: Simpler core vector machines with enclosing balls. In: Proc. of ICML 2007, pp. 911–918 (2007)

[21] Frénay, B., Verleysen, M.: Parameter-insensitive kernel in extreme learning for non-linear support vector regression. Neurocomputing 74(16), 2526–2531 (2011)

[22] Mika, S., Rätsch, G., Weston, J., Schölkopf, B., Smola, A.J., Müller, K.R.: Invariant feature extraction and classification in kernel spaces. In: Solla, S.A., Leen, T.K., Müller, K.R. (eds.) NIPS, pp. 526–532. The MIT Press (1999)

[23] Schneider, P., Geweniger, T., Schleif, F.M., Biehl, M., Villmann, T.: Multivariate class labeling in robust soft LVQ. In: Proceedings of ESANN 2011, pp. 17–22 (2011)