Available online at www.sciencedirect.com
SCIENCE DIRECT®
Pattern Recognition 39 (2006) 189–208
PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/patcog

Prototype selection for dissimilarity-based classifiers☆

Elżbieta Pękalska*, Robert P.W. Duin, Pavel Paclík

Faculty of Electrical Engineering, Mathematics and Computer Science, Mekelweg 4, 2628 CD Delft, Delft University of Technology, The Netherlands

Abstract

A conventional way to discriminate between objects represented by dissimilarities is the nearest neighbor method. A more efficient and sometimes a more accurate solution is offered by other dissimilarity-based classifiers. They construct a decision rule based on the entire training set, but they need just a small set of prototypes, the so-called representation set, as a reference for classifying new objects. Such alternative approaches may be especially advantageous for non-Euclidean or even non-metric dissimilarities.

The choice of a proper representation set for dissimilarity-based classifiers is not yet fully investigated. It appears that a random selection may work well. In this paper, a number of experiments has been conducted on various metric and non-metric dissimilarity representations and prototype selection methods. Several procedures, like traditional feature selection methods (here effectively searching for prototypes), mode seeking and linear programming are compared to the random selection. In general, we find out that systematic approaches lead to better results than the random selection, especially for a small number of prototypes. Although there is no single winner as it depends on data characteristics, the *k*-centres works well, in general. For two-class problems, an important observation is that our dissimilarity-based discrimination functions relying on significantly reduced prototype sets (3–10% of the training objects) offer a similar or much better classification accuracy than the best *k*-NN rule on the entire training set. This may be reached for multi-class data as well, however such problems are more difficult.

Keywords: Dissimilarity; Representation; Prototype selection; Normal density based classifiers; Nearest neighbor rule

1. Introduction

Pattern recognition relies on the description of regularities in observations of classes of objects. How this knowledge is extracted and represented is of crucial importance for learning [1,2]. We think that representations which are alternative to the feature-based descriptions of objects should be studied as they may capture different characteristics of the problem we want to analyze.

Proximity underpins the description of a class as a group of objects possessing similar characteristics. This implies that the notion of proximity is more fundamental than the notion of a feature or of a class. Thereby, it should play a crucial role in class constitution [3,4]. This proximity should be possibly modeled such that a class has an efficient and compact description. Following this principle for a number of years we have been advocating the learning from dissimilarity representations [1,5–7]. They are derived from pairwise object comparisons, where the shared degree of commonality between two objects is captured by a dissimilarity value. Such representations are very general, as they can be derived in many ways, e.g. from raw (sensor) measurements such as images, histograms or spectra or from an initial representation by features, strings or graphs [8]. The choice of such representations can also be suggested by an application or data specification. In fact, in all types of problems referring to string-, graph-, shape or template-matching, as well as to all kinds of information retrieval or image retrieval, the use of (dis)similarities seems to be the most feasible approach.

☆ Parts of this article appear in Chap. 9 of 'The Dissimilarity Representation for Pattern Recognition. Foundations and Applications' by Elzbieta Pekalska and Robert P.W. Duin, published by World Scientific, 2005.

* Corresponding author. Tel.: +44 161 2756194.
E-mail addresses: e.pekalska@ewi.tudelft.nl (E. Pękalska), r.p.w.duin@ewi.tudelft.nl (R.P.W. Duin), p.paclik@ewi.tudelft.nl (P. Paclík).

0031-3203/$30.00
doi:10.1016/j.patcog.2005.06.012

The k-nearest neighbor (k-NN) classifier is commonly practiced on dissimilarity representations due to its simplicity and good asymptotic behavior (on metric distances). It has, however, three main disadvantages: large storage requirements, large computational effort for evaluation of new objects and sensitivity to noisy examples. Prototype optimization techniques can diminish these drawbacks, so research efforts have been devoted to this task; see e.g. [9–12]. From the initial prototypes, such as the objects in the training set, the prototype optimization chooses or constructs a small portion of them such that a high classification performance of the 1-NN rule is achieved. This might be especially of interest when the dissimilarity measure is based on expensive object comparisons.

Although the k-NN rule is mostly applied to metric distances, many non-metric distances are often designed to respond better to practical requirements. They are naturally derived when images or shapes are aligned in a template matching process. For instance, in computer vision, it is known that in the presence of partially occluded objects, non-metric measures are preferred [13]. Other examples are pairwise structural alignments of proteins, variants of the Hausdorff distance [14] and normalized edit-distances [8]. By a common-sense reasoning, the principle behind the voting among the nearest neighbors can be applied to non-metric dissimilarities. The k-NN rule may also work well in such cases [15]. It is simply more important that the measure itself is discriminative for the classes than its strict metric properties. However, many traditional prototype optimization methods are not appropriate for non-metric dissimilarities, especially if no accompanying feature-based representation is available, as they often rely on the triangle inequality.

Since all objects in the training set can be initially used in training, we have suggested to construct classifiers defined as weighted linear (or quadratic) combinations of the dissimilarities to a set of selected prototypes. In such a framework the metric requirements are not essential. In our previous experiments we have found out that a random selection of prototypes often works well [5–7,16]. Here, we will also analyze systematic procedures.

The paper is organized as follows. Section 2 introduces the framework of the dissimilarity-based classification and briefly describes the previous work on prototype optimization for the k-NN rule. Section 3 focuses on random and systematic prototype selection methods for decision functions built on dissimilarity representations. Section 4 describes the metric and non-metric data sets used and the experiments conducted. The results are presented in Section 5 and the discussion and overall conclusions are presented in Section 6.

2. Dissimilarity-based classification

Assume a representation set $R := \{p_1, p_2, \ldots, p_n\}$ as a collection of n prototype objects and a dissimilarity measure d, computed or derived from the objects directly, their sensor representations, or some other initial representation. To maintain generality, a notation of $d(x, z)$ is used when objects x and z are quantitatively compared. d is required to be nonnegative and to obey the reflexivity condition, $d(x, x) = 0$, but it might be non-metric. An object x is represented as a vector of the dissimilarities computed between x and the prototypes from R, i.e. $D(x, R) = [d(x, p_1), d(x, p_2), \ldots, d(x, p_n)]$. For a set T of N objects, it extends to an $N \times n$ dissimilarity matrix $D(T, R)$, which is a dissimilarity representation we want to learn from. Given a complete representation $D(T, T)$, the question now arises how a small set R should be selected out of T to guarantee a good tradeoff between the recognition accuracy and the computational complexity when classifiers are built on $D(T, R)$. This issue will be discussed in the subsequent sections.

2.1. The k-NN rule: previous work

A direct approach to dissimilarities leads to the k-NN method [10]. This rule is applied here to $D(S, R)$, such that the test objects in the set S are classified to the class which is most frequently occurring among the k nearest neighbors in R. In a conventional feature space representation, the k-NN rule relies on the (weighted) Euclidean or city block distance. For metric distances, the k-NN is known to be asymptotically optimal in the Bayes sense [17,18]. It can learn complex boundaries and generalize well, provided that an increasing set R of representative prototypes is available and the volumes of the k-neighborhoods becomes arbitrarily close to zero. However, when the data points of the given R are sparsely sampled or have variable characteristics over the space, the classification performance of the k-NN method may significantly differ from its asymptotic behavior. To handle such situations, many variants of the NN rule as well as many distance measures have been invented or adopted for feature-based representations. They take into account a local structure of the data or weight the neighbor contributions appropriately; see e.g. the work of [19–25]. Such approaches are designed to optimize the k-NN rule.

In the basic setup, the k-NN rule uses the entire training set as the representation set, hence $R = T$. Therefore, the usual criticism points at a space requirement to store the complete set T and a high computational cost for the evaluation of new objects. The k-NN rule also shows sensitivity to outliers, i.e. noisy or even erroneously labeled prototypes. To alleviate these drawbacks, various techniques have been developed in feature spaces to tackle the problem of prototype optimization. Two main types of algorithms can be identified: prototype generation and prototype selection. The first group focuses on merging the initial prototypes (e.g. by the average operation) into a small set of prototypes such that the performance of the k-NN rule is optimized. Examples of such techniques are the k-means algorithm [26,27]

or a learning vector quantization algorithm [28], or more recently, for instance the work of [29,30]. The second group of methods aims at the reduction of the initial training set and/or the increase in the accuracy of the NN predictions. This leads to various editing or condensing methods. Condensing algorithms [10,9,12] try to determine a significantly reduced set of prototypes such that the performance of the 1-NN rule on this set is close to the one reached on the complete training set. This is the consistency property [9]. Editing algorithms [31,12] remove noisy instances as well as close border cases, leaving smoother decision boundaries. They aim to leave homogeneous clusters in the data. Basically, they retain all internal points, so they do not reduce the space as much as other reduction algorithms do.

Since the k-NN method is often applied to metric distances, to avoid the expensive computation time, there has been also interest in approximate and fast nearest neighbor search. Many algorithms have been proposed, usually making use of the triangle inequality. Examples can be found in Refs. [32–38].

2.2. Dissimilarity spaces

Many dissimilarity measures designed in practice are non-metric, such as the modified Hausdorff measure and its variants [14], Mahalanobis distance between probability distributions [27,4] or the normalized edit-distance [8,15]. There are also situations, where the classes are badly sampled due to the measurement costs or problem characteristics, as occur in machine or health diagnostics. In such cases, the k-NN rule, even for a large k and a very large training set will suffer from noisy examples. Yet, we think that much more can be gained when other decision functions are constructed on dissimilarity representations.

In our *dissimilarity space approach* [5,6], a dissimilarity representation $D(T, R)$ is addressed as a data-dependent mapping $D(\cdot, R) : X \to \mathbb{R}^n$ from some initial representation (or measurements) X to the so-called dissimilarity space, specified by the set R. In such a space, each dimension corresponds to a dissimilarity to a prototype from R, i.e. $D(\cdot, p_i)$. Since dissimilarities are nonnegative, all the data examples are projected as points to a nonnegative orthotope of that vector space. In this way, arbitrary structural or statistical dissimilarity measures can be used.

A justification for the construction of classifiers in dissimilarity spaces is as follows. The property that a dissimilarity should be small for similar objects (belonging to the same class) and large for distinct objects, gives a possibility for a discrimination. Thereby, a vector $D(\cdot, p_i)$ of the dissimilarities to the prototype p_i can be interpreted as a feature. If the measure is metric and the dissimilarity $d(p_i, p_j)$ is small, then $d(x, p_i) \approx d(x, p_j)$ for other objects x, which is guaranteed by the backward triangle inequality [4]. This means that in fact either p_i or p_j can be taken as a prototype for the NN rule. This reasoning does not hold for a non-metric measure. To handle such situations, the representation set should be chosen such that for two similar objects x and y, the vectors $D(x, R)$ and $D(y, R)$ are correlated, even if for a particular prototype p_i the dissimilarity values $d(x, p_i)$ and $d(y, p_i)$ differ. Then, the vectors $D(x, R)$ and $D(y, R)$ lie close in the dissimilarity space. Consequently, classifiers constructed there should be useful for non-metric measures. Another important point is that the dimensions of a dissimilarity space, defined by $D(\cdot, p_i)$, convey homogeneous type of information. In that sense, the dimensions are equally important. This is not valid for a general feature-based representation, where features have different characters (e.g. related to different physical quantities) and ranges, as for instance weight or length.

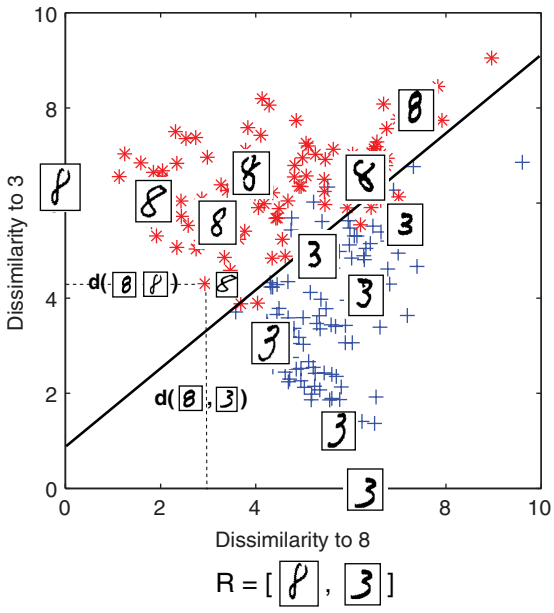

Fig. 1. Example of a 2D dissimilarity spaces and a linear classifier for a subset of handwritten NIST digits 3 and 8. The dissimilarity representation $D(T, R)$ is based on the Euclidean distance between the Gaussian-smoothed binary images. R is randomly chosen and consists of two examples, one for each digit.

If the dissimilarity measure d is metric, then all vectors $D(x, R)$ lie in an n-dimensional prism, bounded from below by a hyperplane on which the objects from R are and bounded from above if the measure is bounded. An example of such a 2D metric representation and the corresponding prism is shown in Fig. 1. Note that in vector spaces of the dimensionality three and higher, the prism is asymmetric and the vertices of its base do not lie on the axes (e.g. in a 3D space the vertices lie in the xy, yz and xz planes). For a non-metric measure, $D(x, R)$ will also lie outside the prism.

2.3. Classifiers in dissimilarity spaces

Defining a well-discriminating dissimilarity measure for a non-trivial learning problem is difficult. Designing such a measure is equivalent to defining good features in a

traditional feature-based classification problem. If a good measure is found and a training set T is representative, then the k-NN rule based on T is expected to perform well. However, when a small representation set (or a condensed set) R is selected, the performance of the k-NN (or 1-NN) rule can significantly deteriorate.

In the case of small representation sets or non-representative training sets, a better generalization can be achieved by a classifier built in a dissimilarity space. Many traditional decision rules can be applied there [4–7]. For instance, a linear classifier becomes a weighted linear combination of the dissimilarities $d(x, p_i)$ between a given object x and the prototypes p_i. The weights are optimized on the training set and large weights (in magnitude) emphasize objects which play an essential role during the discrimination. By doing this, a more global classifier can be built, by which its sensitivity to noisy prototypes is reduced. Our experience confirms that linear or quadratic classifiers tend to generalize better than the k-NN rule, especially for small representation sets [5–7].

In our study [5–7,15,16] we have found out that Bayesian classifiers, i.e. the linear and quadratic normal density based classifiers, perform well in dissimilarity spaces. For a two-class problem, such a linear decision function (BayesNL) based on the representation set R is given by

$$f(D(x,R)) = [D(x,R) - \frac{1}{2}(\boldsymbol{m}_{(1)} + \boldsymbol{m}_{(2)})]^{\mathrm{T}} \times C^{-1}(\boldsymbol{m}_{(1)} - \boldsymbol{m}_{(2)}) + \log \frac{P_{(1)}}{P_{(2)}} \quad (1)$$

and the quadratic function (BayesNQ) becomes

$$f(D(x,R)) = \sum_{i=1}^{2} (-1)^i \, (D(x,R) - \boldsymbol{m}_{(i)})^{\mathrm{T}} \times C_{(i)}^{-1}(D(x,R) - \boldsymbol{m}_{(i)}) + 2 \log \frac{p_{(1)}}{p_{(2)}} + \log \frac{|C_{(1)}|}{|C_{(2)}|}, \quad (2)$$

where $\boldsymbol{m}_{(1)}$ and $\boldsymbol{m}_{(2)}$ are the mean vectors, C is the sample covariance matrix and $C_{(1)}$ and $C_{(2)}$ are the estimated class covariance matrices, all computed in the dissimilarity space $D(T, R)$. $p_{(1)}$ and $p_{(2)}$ are the class prior probabilities. When the covariance matrices become singular, they are regularized. In this paper, we make use the following regularization strategy [39]: $C_{reg}^{\lambda} = (1-\lambda)\,C + \lambda\,\mathrm{diag}(C)$. The regularization term λ is expressed relatively to the variances, so it can be determined more easily. In practice, λ equals 0.01 or less. We keep it fixed in multi-class problems, where the decision is based on the maximum a posterior probability, estimated according to the assumed normal density models [39]. For simplicity, all prior probabilities are considered to be equal, even if the classes are not evenly distributed.

3. Prototype selection and the representation set

The selection of a representation set for the construction of classifiers in a dissimilarity space serves a similar goal as the selection of prototypes to be used by the NN rule: the minimization of the set of dissimilarities to be measured for the classification of new incoming objects. There is, however, an important difference with respect to the demands. Once selected, the set of prototypes defines the NN classifiers independently of the remaining part of the training set. The selection of the representation set, on the other hand, is less crucial, as it will define a dissimilarity space in which the entire training set is used to train a classifier. For this reason, even a randomly selected representation set may work well [5]. That is why, the random selection will serve as a basic procedure for comparing more advanced techniques.

Similar objects will yield a similar contribution to the representation. It may, thereby, be worthwhile to avoid the selection of objects with small dissimilarity values. Moreover, if the data describe a multi-modal problem, it may be advantageous to select objects related to each of the modes. Consequently, the use of procedures like vector quantization or cluster analysis can be useful for the selection of prototypes. The following procedures will be compared for the selection of a representation set: *Random*, *RandomC*, *KCentres*, *ModeSeek*, *LinProg*, *FeatSel*, *KCentres-LP* and *EdiCon*.

Assume c classes: $\omega_1, \ldots, \omega_c$. Let T be a training set and let T_{ω_i} describe the training objects of the class ω_i. Each method selects K objects for the representation set R. If the algorithm is applied to each class separately, then k objects per class are chosen such that $ck = K$. The approaches are explained below.

Random. A random selection of K objects from T.

RandomC. A random selection of k objects per class.

KCentres. This technique [39] is applied to each class separately. For each class ω_i, it tries to choose k objects such that they are evenly distributed with respect to the dissimilarity information $D(T_{\omega_i}, T_{\omega_i})$. The algorithm proceeds as follows:

(1) Select an initial set $R_{\omega_i} := \{p_1^{(i)}, p_2^{(i)}, \ldots, p_k^{(i)}\}$ consisting of k objects, e.g. randomly chosen, from T_{ω_i}.
(2) For each $x \in T_{\omega_i}$ find its nearest neighbor in R_{ω_i}. Let J_j, $j = 1, 2, \ldots, k$, be a subset of T_{ω_i} consisting of objects that yield the same nearest neighbor $p_j^{(i)}$ in R_{ω_i}. This means that $T_{\omega_i} = \cup_{j=1}^{k} J_j$.
(3) For each J_j find its center c_j, that is the object for which the maximum distance to all other objects in J_j is minimum (this value is called the radius of J_j).
(4) For each center c_j, if $c_j \neq p_j^{(i)}$, then replace $p_j^{(i)}$ by c_j in R_{ω_i}. If any replacement is done, then return to (2), otherwise STOP. The final representation set R consists of all sets R_{ω_i}.

Except for step (3), this routine is identical to the k-means [26] performed in a vector space. The result of the k-centres procedure heavily depends on the initialization. For that reason we use it with some precautions. To determine the set R_{ω_i} of k objects, we start from a chosen center for the entire set and then more centers are gradually added. At any point, a group of objects belongs to each center. R_{ω_i} is enlarged by splitting the group of the largest radius into two and replacing its center by two other members of that group. This stops, when k centers are determined. The entire procedure is repeated $M = 30$ times, resulting in M potential sets from which the one yielding the minimum of the largest final subset radius is selected.

ModeSeek. This method [40] focuses on the modes in the dissimilarity data in the specified neighborhood size s. It is used in a class-wise way. For each class ω_i, the algorithm proceeds as follows:

(1) Set a relative neighborhood size as an integer $s > 1$.
(2) For each object $x \in T_{\omega_i}$ find the dissimilarity $d_{s-NN}(x)$ to its sth neighbor.
(3) Find a set R_{ω_i} consisting of all objects $x_j \in T_{\omega_i}$ for which $d_{s-NN}(x_j)$ is minimum within its set of s neighbors.

The final representation set R consists of all the sets R_{ω_i}. The objects found in this way are the estimated modes of the class distribution. The final cardinality of R depends on the choice of s. The larger s, the smaller R_{ω_i}. If a representation set R of the cardinality $K = kc$ is searched, then for each class ω_i, the neighborhood size s is selected such that it generates the largest set R_{ω_i} which is not larger than the demanded size k.

All these procedures may be called unsupervised, in spite of the fact that they are used in a class-wise way. They aim at various heuristics, but they do not consider the quality of the resulting representation set in terms of the class separability. A traditional procedure to do that is a feature selection.

FeatSel. The original dissimilarity representation $D(T, T)$ is reduced to $D(T, R)$ by selecting an optimal set of K features $D(\cdot, p_i), i = 1, 2, \ldots, K$, according to some separability measure. Here, we will use the forward feature selection [41] and the leave-one-out 1-NN error as a selection criterion. The difference to the standard approach is, however, that the features are judged in a dissimilarity space, but the 1-NN error is computed on the given dissimilarities $D(T, T)$ directly. The method is thereby fast as it is entirely based on comparisons and sorting. Ties can easily occur by the same number of misclassified objects for different representation sets. They are solved by selecting the set R for which the sum of dissimilarities is minimum.

LinProg. Here, the selection of prototypes is done automatically by training a properly formulated separating hyperplane $f(D(x, R)) = \sum_{j=1}^{n} w_j\, d(x, p_j) + w_0 = \boldsymbol{w}^{\mathrm{T}} D(x, R) + w_0$ in a dissimilarity space $D(T, R)$. R can be chosen as identical to the training set T, but it can also be different. Such a linear function is obtained by solving a linear programming problem, where a sparse solution is imposed by minimizing the l_1-norm of the weight vector $\boldsymbol{w}$, $||\boldsymbol{w}||_1 = \sum_{j=1}^{r} |w_j|$. To formulate such a minimization task properly, the absolute values $|w_j|$ should be eliminated from the objective function. Therefore, w_j are expressed by non-negative variables α_j and β_j as $w_j = \alpha_j - \beta_j$. When the pairs (α_j, β_j) are determined, then at least one of them is zero. Nonnegative slack variables ξ_i, accounting for possible classification errors, and a regularization parameter C are additionally introduced. For a set of training objects $x_i \in T$ with the class labels $y_i \in \{1, -1\}$, the minimization problem becomes then[1] :

$$\begin{aligned} &\text{Minimize} \quad \sum_{i=1}^{n} (\alpha_i + \beta_i) + \gamma \sum_{i=1}^{n} \xi_i \\ &\text{subject to} \quad y_i\, f(D(x_i, R)) \geqslant 1 - \xi_i, \quad i = 1, \ldots, n \\ &\qquad\qquad \alpha_i,\ \beta_i,\ \xi_i \geqslant 0. \end{aligned} \tag{3}$$

In this approach, a sparse solution $\boldsymbol{w}$ is obtained, which means that many weights w_i become zero. The objects from the initial set R ($R = T$, for instance), corresponding to non-zero weights are the selected prototypes, so the representation set R_{LP}. Although, the prototypes are found in the optimization for a particular separating hyperplane, they can be used by other discrimination functions as well. We have found out that the choice of the tradeoff parameter as $\gamma = 1$ seems to be reasonable for many problems, so we fix it in our experiments.

This selection of objects described above is similar to the selection of features by linear programming in a standard classification task [43]. The important point to realize is that we do not have a control over the number of selected prototypes. This can be slightly influenced by varying the constant γ (hence influencing the tradeoff between the classifier norm $\|\boldsymbol{w}\|_1$ and the training classification errors), but not much. From the computational point of view, this procedure is advantageous for two-class problems, since multi-class problems may result in a large set R_{LP}. This occurs since different prototypes are often selected by different classifiers when a multi-class classifier is derived in the one-against-all strategy or even more severely in the pairwise strategy.

KCentres-LP. The KCentres algorithm is applied to $D(T, T)$ as described above, preselecting the representation set R_{KC}. This is then followed by a reduction based on the linear programming procedure applied to $D(T, R_{KC})$. In this way, we can somewhat influence the cardinality of the resulting prototype set. Still, if R_{KC} is not sufficiently large, the linear programming will need all the objects. Consequently, this procedure reduces to the KCentres if R_{KC} is small.

[1] Another more flexible linear programming formulation has been proposed in Ref. [42], but here we limit ourselves to this case only.

EdiCon. An editing and condensing algorithm is applied to an entire dissimilarity representation $D(T, T)$, resulting in a final condensed set R. Editing takes care that the noisy objects are first removed so that the prototypes can be chosen to guarantee good performance of the 1-NN rule. Similarly as for the LinProg procedure, the number of prototypes is determined automatically.

In summary, random, clustering-based unsupervised and supervised prototype selection procedures are considered in dissimilarity spaces.

4. Experimental setup

A set of experiments is conducted to compare various prototype selection methods for classification problems in dissimilarity spaces. Smaller representation sets are of interest because of a low complexity for both representation and evaluation of new objects. Both linear (BayesNL) and quadratic (BayesNQ) classifiers can be considered in dissimilarity spaces. However, we will present the results only for the BayesNQ, since it performs mostly better than the BayesNL, provided that sufficiently large training sets are available. This holds for two-class classification problems, which is our main focus point here. Note, however, that in high dimensional dissimilarity spaces, i.e. for large representation sets, the BayesNQ is computationally much more expensive than the BayesNL.

Prototype selection strategies are compared by the performance of a single classifier. As a result, the performance of the linear programming machine (LinProg) is not used, as it is incomparable to the performance of the BayesNQ for a representation set selected by other methods. The LinProg procedure is only used for the selection of prototypes.

In each experiment, the data sets are divided into a training set T and a test set S. The BayesNQ is trained on the dissimilarity representation $D(T, R)$ and tested on $D(S, R)$, where $R \subset T$ is a representation set of K prototypes chosen according to some specified criteria, as described in Section 3. The 1-NN and the k-NN results defined on the entire training set (hence tested on $D(S, T)$) are provided as a reference. Also, as a comparison, the k-NN rule is directly applied to the given dissimilarities $D(T, R)$, when R is selected by the KCentres algorithm and to the k-NN rule applied to Euclidean distances computed over $D(T, R)$ (which corresponds to the k-NN rule performed in the dissimilarity space). The k-NN rule optimizes the parameter k in a leave-one-out procedure over the training set [39].

Most of our experiments are performed on two-class classification tasks, since such problems should be understood first. Although a part of the investigation concerns example multi-class problems, in fact, a separate study is needed to find adequate prototype selection procedures. This is left for further research.

Table 1
Characteristics of the data sets used in experiments

Data	# Classes	# Objects per class (in total)	α per class
Polydisth	2	2×2000	0.25
Polydistm	2	2×2000	0.25
NIST-38	2	2×1000	0.10
Zongker-12	2	2×100	0.50
RoadSign	2	2×300	0.50
GeoSam	2	2×500	0.50
GeoShape	2	2×500	0.50
Wine	3	59/71/48	0.60
Ecoli-p1	3	143/77/52	0.60
Ecoli-p08	3	143/77/52	0.60
ProDom	4	878/404/271/1051	0.35
Zongker-all	10	10×100	0.50

α stands for the fraction of objects selected for training in each repetition.

4.1. Data sets

In all our experiments the data sets are divided into training and test sets of various sizes; details can be found in Table 1. We have chosen a number of problems possessing various characteristics: defined by both metric (Euclidean or non-Euclidean) and non-metric dissimilarity measures, as well as, concerning small and large sample size problems. Seven data sets are used in our study: randomly generated polygons, NIST scanned digits, geophysical spectra and road sign vs. non road sign images, proteins and their localization sites and wine types, resulting in 12 dissimilarity representations (for some data sets, two different measures are considered). The data sets refer to two-, three-, four- and ten-class classification problems.

If a dissimilarity d is Euclidean, then the square $N \times N$ dissimilarity representation $D(T, T)$ can be perfectly embedded in a Euclidean space. This means that a configuration X can be found such that the Euclidean distances between the vectors of X correspond to the original ones. This is equivalent to the statement that the Gram matrix $G = -\frac{1}{2} J D^{*2} J$, where $D^{*2} = (d_{ij}^2)$ and $J = I - \frac{1}{n} 11^{\mathrm{T}}$, is positive semidefinite i.e. all its eigenvalues are nonnegative. A non-Euclidean representation D can be embedded in a pseudo-Euclidean space [5,44], which is composed of a direct orthogonal sum of two Euclidean subspaces with an inner product being positive definite in the first one and negative definite in the other. A configuration X in a pseudo-Euclidean space is determined by the eigendecomposition of $G = Q \Lambda Q^{\mathrm{T}}$, where Λ is a diagonal matrix of decreasing positive eigenvalues followed by decreasing (in magnitude) negative eigenvalues and then zeros, and Q is an orthogonal matrix of the corresponding eigenvectors. X is found as $X = Q_m |\Lambda_m|^{1/2}$, where m corresponds to the number of non-zero eigenvalues. See [4,5,15] for details.

Let us denote the eigenvalues by λ's. Hence, the magnitudes of negative eigenvalues indicate the amount of deviation from the Euclidean behavior. This is captured by the

Table 2
Properties of the data sets used in experiments

Data	Dissimilarity	Property	r_{mm}^{nE} (%)	r_{rel}^{nE} (%)	r_{tr}^{nM} (%)
Polydisth	Hausdorff	M, nE	25.1	38.1	0.00
Polydistm	Mod. Hausdorff	nM	11.0	31.4	0.01
NIST-38	Euclidean	E	0.0	0.0	0.00
Zongker-12	Template-match	nM	13.3	30.1	0.70
RoadSign	Correlation	E	0.0	0.0	0.00
GeoSam	SAM [47]	M,nE	0.1	0.1	0.00
GeoShape	Shape l_1	M, nE	2.6	7.2	0.00
Wine	Euclidean distance	E	0.0	0.0	0.00
Ecoli-p1	l_1 distance	M, nE	9.8	22.0	0.00
Ecoli-p08	$l_{0.8}$ distance	nM	13.4	24.7	3.84
ProDom	Structural	nM	1.3	0.9	10^{-5}
Zongker-all	Template-match	nM	38.9	35.0	0.41

The following abbreviations are used: M, metric, E, Euclidean, nM, non-metric, nE, non-Euclidean. The values r_{mm}^{nE} and r_{rel}^{nE} indicate the deviations from the Euclidean behavior, as defined in formula (4) and r_{tr}^{nM} describes the percentage of disobeyed triangle inequalities.

following indices:

$$r_{mm}^{nE} = \frac{|\lambda_{\min}|}{\lambda_{\max}} \times 100$$
$$r_{rel}^{nE} = \frac{\sum_{\lambda_i<0}|\lambda_i|}{\sum_{j=1}^{N}|\lambda_j|} \times 100. \quad (4)$$

r_{mm}^{nE} is the ratio of the smallest negative eigenvalue to the largest positive one, while r_{rel}^{nE} describes the contribution of negative eigenvalues. Additionally, an indication of the non-metric behavior can be expressed by the percentage of disobeyed triangle inequalities, r_{tr}^{nM}.

Table 2 provides suitable information on the Euclidean and metric aspects of the measures considered. The Hausdorff representation of the polygon data is strongly non-Euclidean. The modified Hausdorff representation of the polygon data, as well as template-matching representation of the digits data are moderately non-Euclidean and non-metric. Concerning the geophysical data, the shape dissimilarity representation is slightly non-Euclidean, while the SAM representation is nearly Euclidean. Both are metric. For the *Ecoli* data, two representations are used: the metric, moderately non-Euclidean l_1 distance representation and the non-metric $l_{0.8}$ distance representation. ProDom representation is slightly non-metric and slightly non-Euclidean. The remaining three data sets: road signs, NIST digits and Wine have Euclidean representations.

For the purpose of visualization also 2D approximate embeddings of dissimilarity representations have been found. They rely on linear projections from the corresponding Gram matrices, as described above; see also [4,5,15]. The sum of the first two largest eigenvalues with respect to the total sum of all eigenvalue magnitudes indicates how much of the original dissimilarities is reflected in the projections. This can be observed in Figs. 2–8. There we also show all the eigenvalues of the Gram matrices (derived from the dissimilarity matrices), hence the deviation from the Euclidean behavior can be visually judged. The number of eigenvalues significantly different from zero indicates the intrinsic dimensionality of a problem. The study on embeddings is beyond the scope of this paper; they are treated here for the purpose of exploratory data analysis. As judged from two-class problems, Figs. 2–5, the polygon data seems the most complex, while the *Zongker-12* data seems the easiest. On the other hand, the ten-class *Zongker-all* data is the most complex.

Polygon data. The data consist of two classes of randomly generated polygons, convex quadrilaterals and irregular heptagons. The classes are represented by 2000 examples. The polygons are first scaled and then the Hausdorff and modified Hausdorff distances [14] between their vertices are computed, yielding the *Polydisth* and *Polydistm* data, respectively. Let A and B be two polygons. Then, the Hausdorff distance between them is computed as $d_H(A,B) = \max\{d_H^{\triangleright}(A,B), d_H^{\triangleright}(B,A)\}$, where $d_H^{\triangleright}(A,B) = \max_{a\in A}\min_{b\in B} d(a,b)$ is a directed Hausdorff distance and $d(a,b)$ is the Euclidean distance between the corners of two polygons. The modified Hausdorff distance is computed as $d_{MH}(A,B) = \max\{d_{avr}^{\triangleright}(A,B), d_{avr}^{\triangleright}(B,A)\}$, where $d_{avr}^{\triangleright}(A,B) = \frac{1}{|A|}\sum_{a\in A}\min_{b\in B} d(a,b)$. The Hausdorff distance is metric, while the modified Hausdorff distance is not [4,14]. See also Fig. 9.

NIST digit data. The data describe the scanned digits [45], originally given as 128×128 binary images; see Fig. 10. In total, there are 10 classes, each represented by 200 examples. The images are first smoothed with the Gaussian kernel with $\sigma = 8$ pixels and then the Euclidean distances between such blurred images are derived. The smoothing is done to make the distance representation somewhat robust against tilting or shifting. Only the digits 3 and 8 were used in our experiments here, yielding the *NIST-38* data. See also Fig. 10.

Zongker and Jain digit data. The data describe the NIST digits [45], originally given as 128×128 binary images. Here, the similarity measure, based on deformable template matching, as defined by and Jain Zongker [46], is used. Let

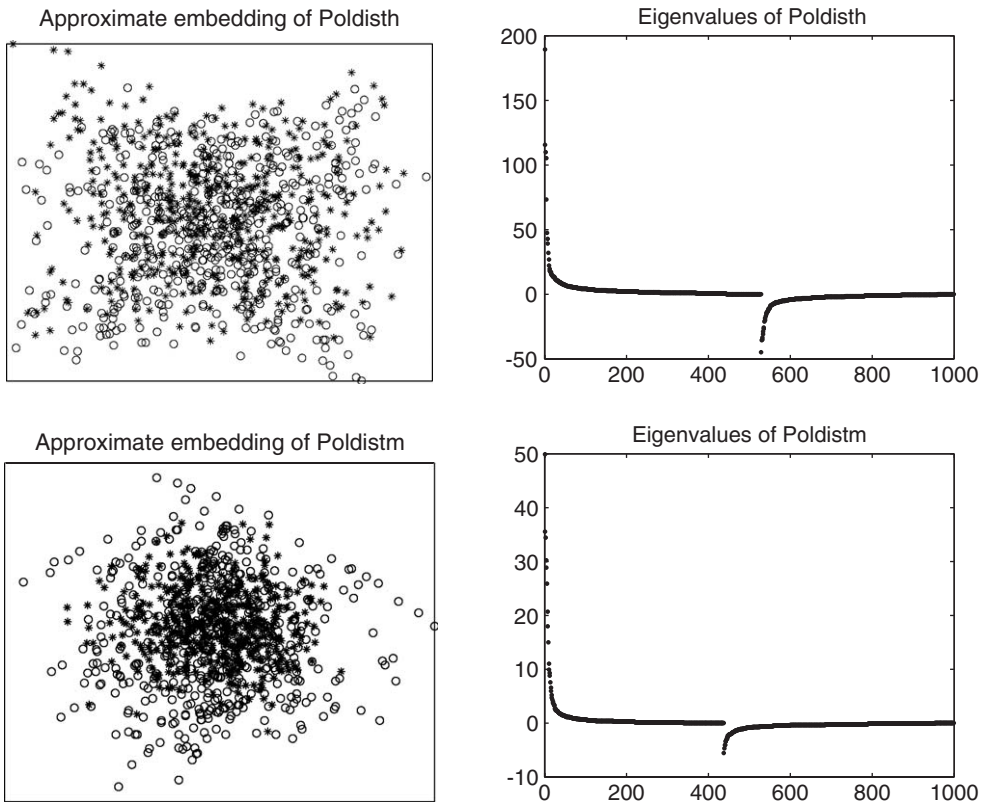

Fig. 2. Left: approximate 2D embedding of the dissimilarity representations $D(T, T)$ for the polygon data. Right: all eigenvalues derived in the embedding process.

$S=(s_{ij})$ denote the similarities. The off-diagonal symmetric dissimilarities $D=(d_{ij})$ are computed as: $d_{ij}=(s_{ii}+s_{jj}-s_{ij}-s_{ji})^{1/2}$ for $i \neq j$, since the data are slightly asymmetric. D is significantly non-metric. In our experiments, the digits 1 and 2 (the *Zongker-12* problem) are used, as well as all the classes (*Zongker-all*).

Geophysical spectra. The data set describes two multi-modal classes. Each class is represented by 500 examples. The classes are described by high-dimensional wavelength spectra. Since the data are confidential we cannot provide more details. The spectra are first normalized to a unit area and then two dissimilarity representations are derived. The first one relies on the spectral angle mapper distance (SAM) [47] defined for the spectra $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ as $d_{SAM}(\boldsymbol{x}_i, \boldsymbol{x}_j)=\arccos(\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j/\|\boldsymbol{x}_j\|_2\,\|\boldsymbol{x}_j\|_2)$. The second dissimilarity is based on the l_1 (city block) distance between the Gaussian smoothed (with $\sigma=2$ bins) first order derivatives of the spectra [7,16]. Since by the use of the first derivative, the shape of the spectra is somewhat taken into account, we will refer to this measure as to shape dissimilarity. The data sets are named *GeoSam* and *GeoShape*, respectively.

Road signs. The *RoadSign* data set consists of gray level images of circular road signs scaled to 32×32 pixel raster. Some examples are shown in Fig. 11. Three hundred road sign images (highly multi-modal) and 300 non-road sign images acquired under general illumination are considered [48]. The latter images are identified by a sign detector using a circular template based on local edge orientations. Since the circular template was used to detect the boards, this a priori knowledge was used to remove the pixels in the background. The resulting data set contains 793 of original 1024 dimensions (pixels). Normalized cross-correlation (similarity) is computed between the images. Let s_{ij} denote the similarities. Then, the dissimilarities are computed as $d_{ij}=(1-s_{ij})^{1/2}$.

Wine data. The *Wine* data come from Machine Learning Repository [49] and describe three types of wines described by 13 features. In each experiment, when the data are split into the training and test sets, the features are standardized as they have different ranges. A Euclidean distance is chosen for the representation.

Ecoli data. The data come from Machine Learning Repository [49] and describe eight protein localization sites. Since the number of examples in all these classes is not sufficient for a prototype selection study, three largest localization sites are selected as a sub-problem. These localization classes are: cytoplasm (143 examples), inner membrane without signal sequence (77 examples) and periplasm (52 examples). Since the features are some type of scores between 0 and 1, they are not normalized. Five numerical attributes are taken into account to derive the l_1 and $l_{0.8}$ distance representations, denoted as *Ecoli-p1* and *Ecoli-p08*, respectively. Remember

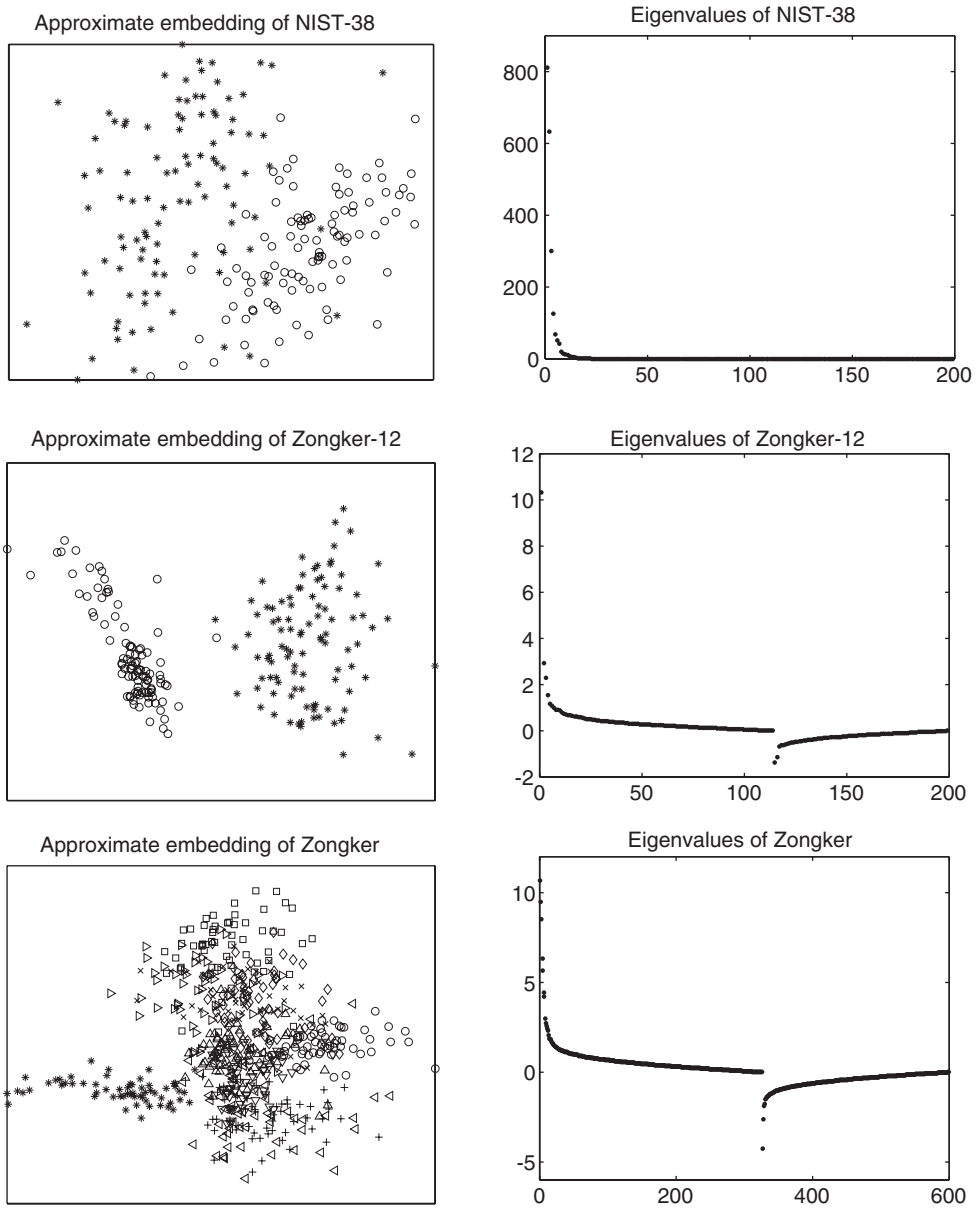

Fig. 3. Left: approximate 2D embedding of the dissimilarity representations $D(T, T)$ for the NIST data. Right: all eigenvalues derived in the embedding process.

that the l_p distance between two vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is computed $d_p(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\sum_{z=1}^{m} |x_{iz} - x_{jz}|^p)^{1/p}$ and it is metric for $p \geqslant 1$.

ProDom. ProDom is a comprehensive set of protein domain families [50]. A *ProDom* subset of 2604 protein domain sequences from the ProDom set [50] was selected by Roth [51]. These are chosen based on a high similarity to at least one sequence contained in the first four folds of the SCOP database. The pairwise structural alignments are computed by Roth [51]. Each SCOP sequence belongs to a group, as labeled by the experts [52]. We use the same four-class problem in our investigations. Originally, a structural similarities s_{ij} are derived, from which the dissimilarities are derived as $d_{ij} = (s_{ii} + s_{jj} - 2s_{ij})^{1/2}$ for $i \neq j$. $D = (d_{ij})$ is slightly non-Euclidean and slightly non-metric.

5. Results

The results of our experiments are presented in Figs. 12–19. They show the generalization errors of the BayesNQ classifier as a function of the number of prototypes chosen by various selection methods. These error curves are compared to some variants of the NN rule. Note that in order to emphasize a small number of prototypes, the horizontal axis is logarithmic. The prototype selection

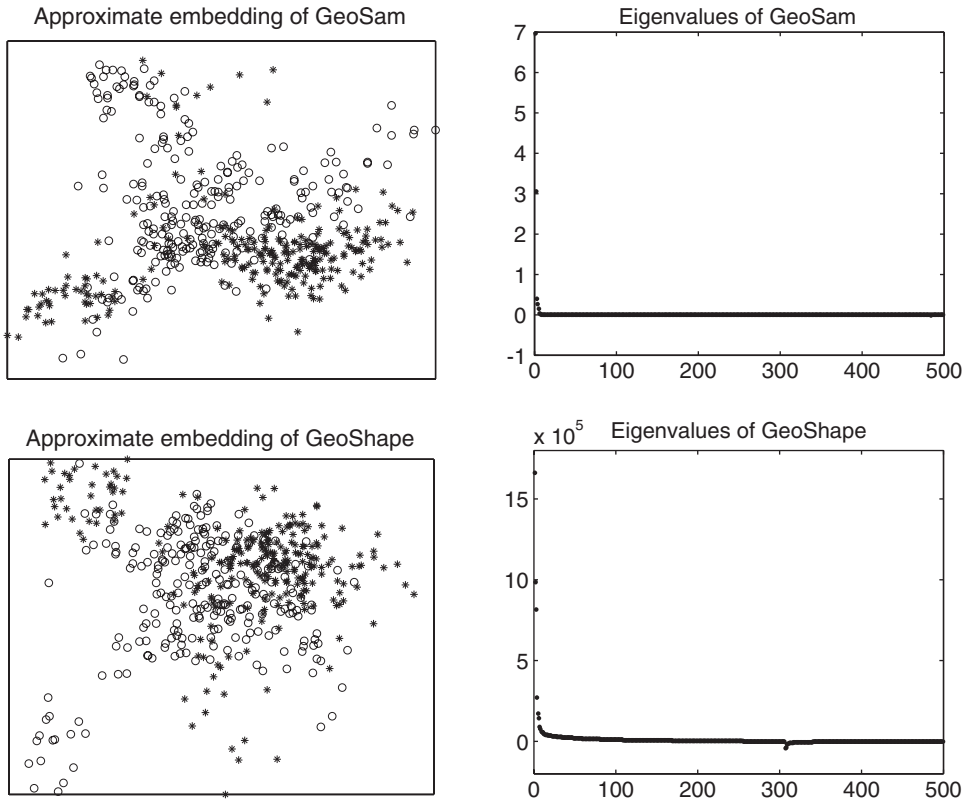

Fig. 4. Left: approximate 2D embedding of the dissimilarity representations $D(T, T)$ for the geophysical spectra data. Right: all eigenvalues derived in the embedding process.

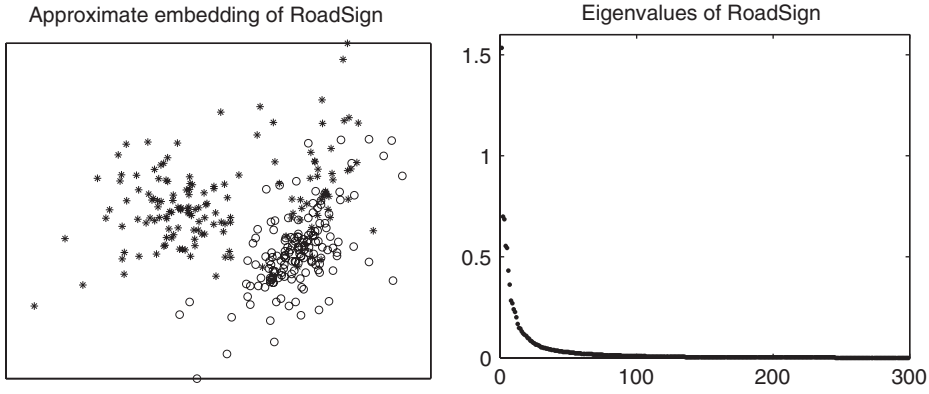

Fig. 5. Left: approximate 2D embedding of the dissimilarity representation $D(T, T)$ for the road signs data. Right: all eigenvalues derived in the embedding process.

methods mentioned in the legends are explained in Section 3. Concerning the NN methods, the following abbreviations are used. The 1-NN-final and the k-NN-final stand for the NN results obtained by using the entire training set T, hence such errors are plotted as horizontal lines. They are our reference. k-NN is the k-NN rule directly applied to $D(T, R)$, while the k-NN-DS is the Euclidean distance k-NN rule computed in $D(T, R)$ dissimilarity spaces (this means that a new Euclidean distance representation is derived from the vectors $D(x, R)$). In both cases, the representation set R is chosen by the *KCentres* algorithm. *EdiCon-1-NN* presents the 1-NN result for the prototypes chosen by the editing and condensing *EdiCon* criterion. The optimal parameter k in all the k-NN rules used is determined by the minimization of the leave-one-out error on the training set. Sometimes, k is found to be 1 and sometimes, some other value.

The performances of all procedures mentioned in the legends, from *Random* to *EdiCon* are based on the BayesNQ

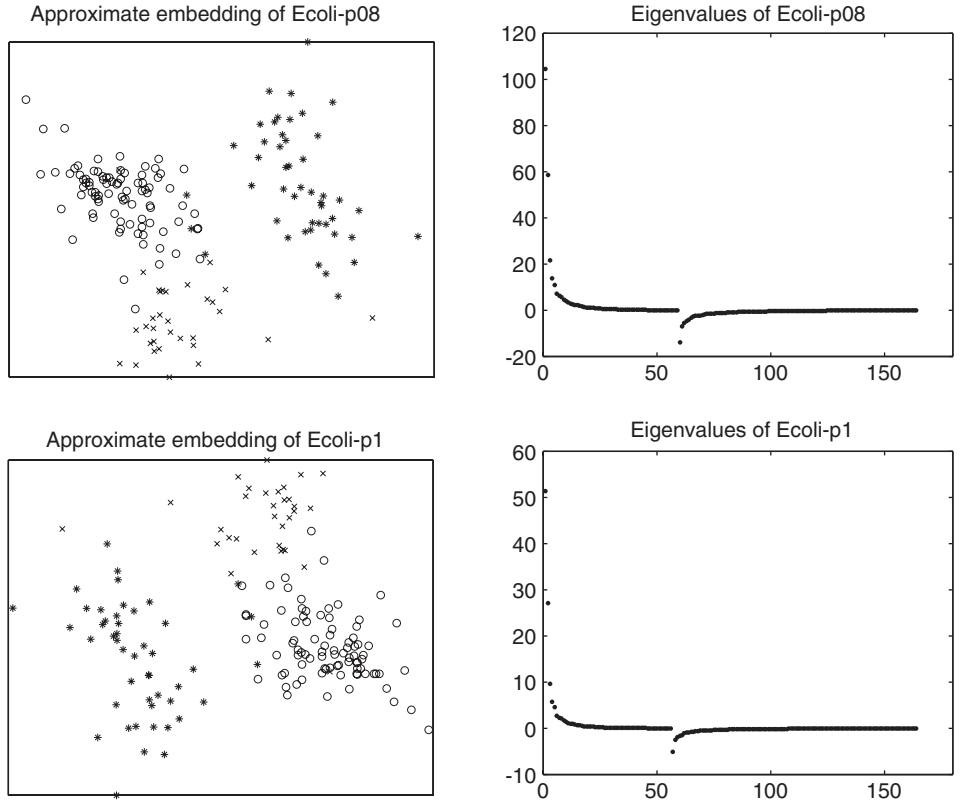

Fig. 6. Left: approximate 2D embedding of the dissimilarity representations $D(T, T)$ for the *Ecoli* data. Right: all eigenvalues derived in the embedding process.

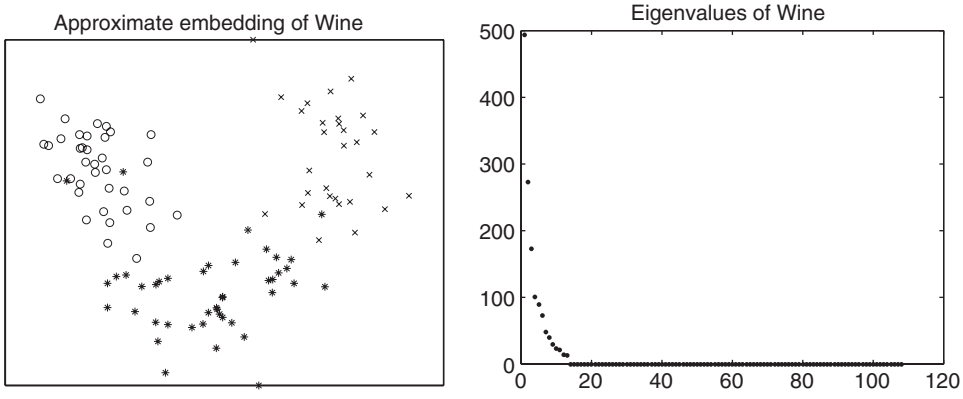

Fig. 7. Left: approximate 2D embedding of the dissimilarity representation $D(T, T)$ for the *Wine* data. Right: all eigenvalues derived in the embedding process.

classifier in the dissimilarity space defined by the selected set of prototypes R. So, they need just the computation of the reduced set of similarities for testing purposes, but they profit indirectly from the availability of the entire training set T.

To enhance the interpretability of the results, we used the following patterns in all plots. The supervised methods *KCentres-LP* and *FeatSel* are plotted by continuous lines, the unsupervised, clustering selections are plotted by dash-dotted lines and the random methods are plotted by dashed lines.

Our experiments are based on M repetitions, that is M random selections of a training set. $M = 10$ for the *Prodom* and *Zongker-all* dissimilarity data and $M = 25$, otherwise. The remaining part of the data is used for testing. Different selection procedures used the same collections of the training and test sets. The averaged test errors are shown in the figures. We do not present the resulting standard deviations

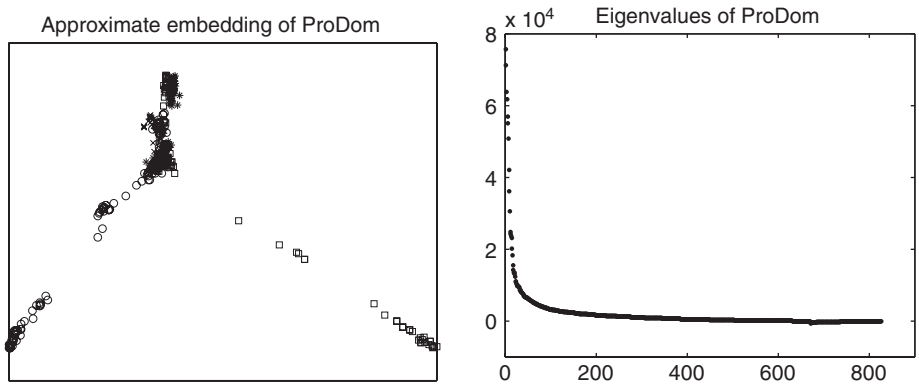

Fig. 8. Left: approximate 2D embedding of the dissimilarity representation $D(T, T)$ for the *ProDom* data. Right: all eigenvalues derived in the embedding process.

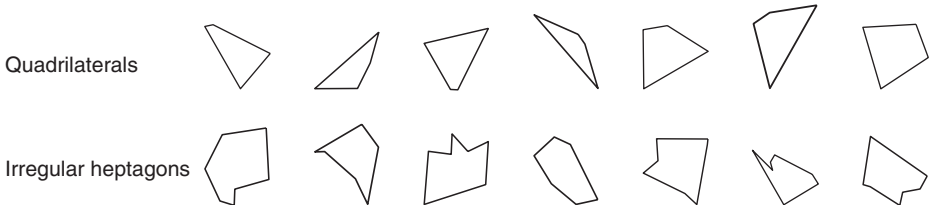

Fig. 9. Examples of the polygon data.

Fig. 10. Examples of the NIST digits, re-sampled to 16×16 pixels.

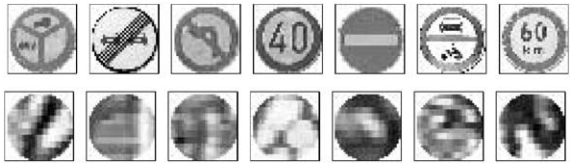

Fig. 11. Examples of the road signs and non-sign regions. The background pixels were discarded.

to maintain the clarity of the plots. In general, we found that the standard deviations are vary between 3% and 7% of the averaged errors.

5.1. Discussion on experimental results

Here we will discuss some details of Figs. 12–18. Fig. 12 presents the results for the two dissimilarity measures derived from the same set of polygons. Remember that the *Polydisth* is metric and *Polydistm* is not. The first striking observation is that in spite of its non-metric behavior, the *Polydistm* results are better: lower NN errors, less prototypes needed to yield a good result. Just 20 prototypes out of 1000 objects are needed to obtain a better error than found by the NN rules. In the k-NN classifiers, the average optimal k appeared to be 127 (*Polydisth*) or 194 (*Polydistm*). These large values correspond to the observation made before in relation to the scatter plots (Fig. 2) that this is a difficult data set. Nevertheless, in the case of the *Polydistm* data, the linear programming technique finds a small set of 55 prototypes for which the BayesNQ error is very low (0.4%). The systematic procedures KCentres (KCentres-LP) and FeatSel perform significantly better than the other ones. The feature selection is also optimal for small representation sets. Notice also the large difference between the two results for editing and condensing. They are based on the same sets of prototypes, but the classification error of the 1-NN rule (in fact a nearest prototype rule), *EdiCon-1-NN*, is much worse than of the BayesNQ classifier, *EdiCon*, which is trained on $D(T, R)$. This remains true for all the considered problems as well, as can be observed in other plots.

Fig. 13 shows the results for two of the NIST digit classification problems. The *NIST-38* data set is based on a Euclidean distance measure, while the *Zongker-12* relies on a non-metric shape comparison. The k-NN classifier does not improve over the 1-NN rule, indicating that the data set sizes (100 objects per class) are too small to model the digit variabilities properly. Again, the systematic procedures do well for small representation sets, but they are outperformed by the *KCentres* routine for a larger number of prototypes. The KCentres method distributes the prototypes evenly over the classes in a spatial way, that is related to the dissimilarity information. For small training sets (here 100

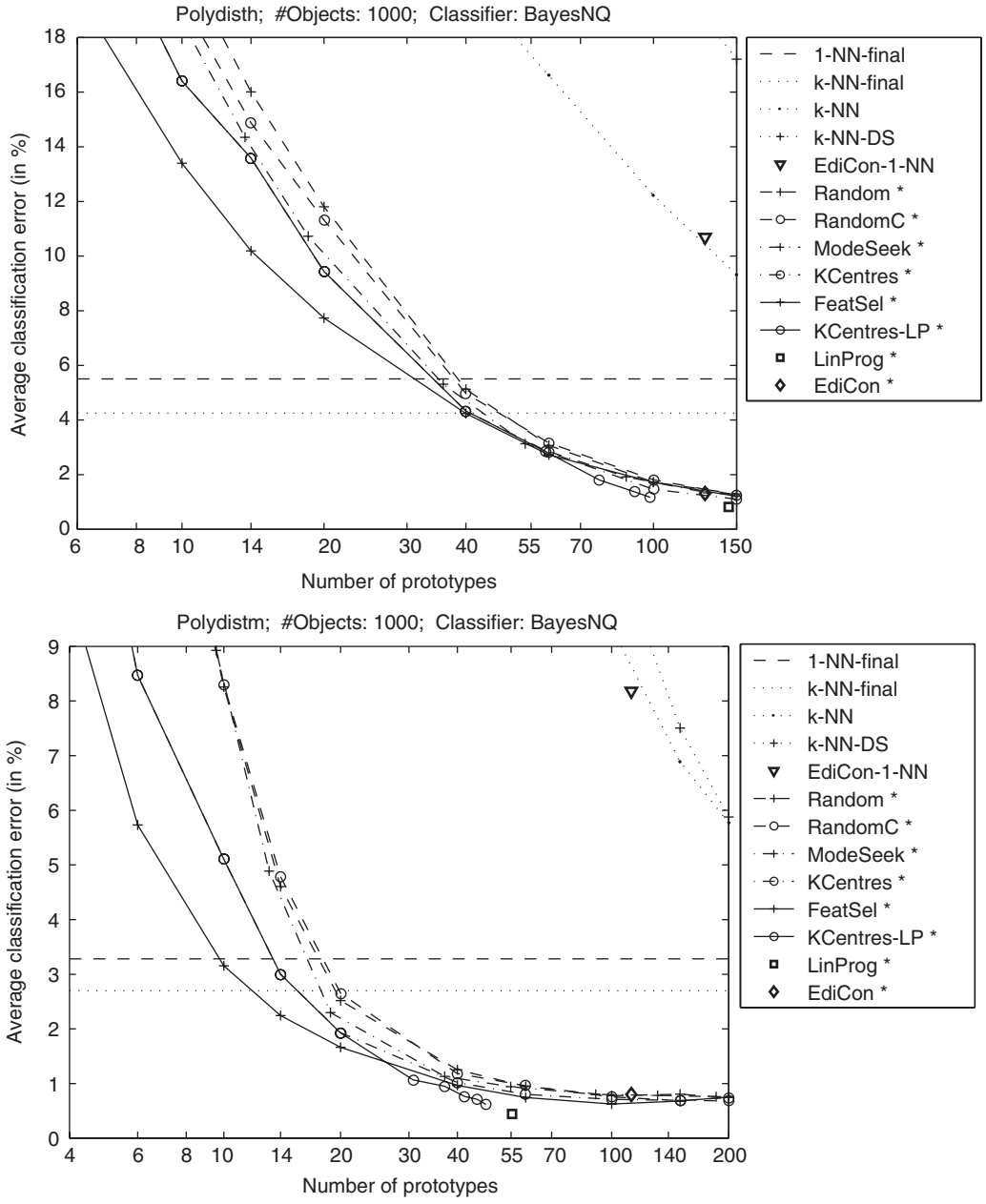

Fig. 12. Polygon data. Average classification error of the *BayesNQ** and the *k*-NN classifiers in dissimilarity spaces as well as of the direct *k*-NN rule as a function of the number of selected prototypes.

examples per class), this may be a better than an advanced optimization.

Fig. 14 presents the results for the two dissimilarity representations of the geophysical data sets. From other experiments it is known that they are highly multi-modal, which may explain the good performance of the *ModeSeek* for the *GeoShape* problem and the *KCentres* for the *GeoSam* problem. Editing and condensing does also relatively well. Feature selection works also well for a small number of prototypes. Overall, the linear programming yields good results. Recall that we take the *KCentres* results as a start (except from the final result indicated by the square marker that starts from the entire training set), so the *KCentres* curve is for lower numbers of prototypes underneath it. In this problem we can hardly improve over the NN performance, but still need just 5–10% of the training set size for prototypes. In the next section, however, it is shown that these results can still be significantly improved by modifying the dissimilarity measure.

In Fig. 15, the results for the *RoadSign* problem are shown. An interesting observation for these Euclidean dissimilarity data is that 5% of the training examples are needed as

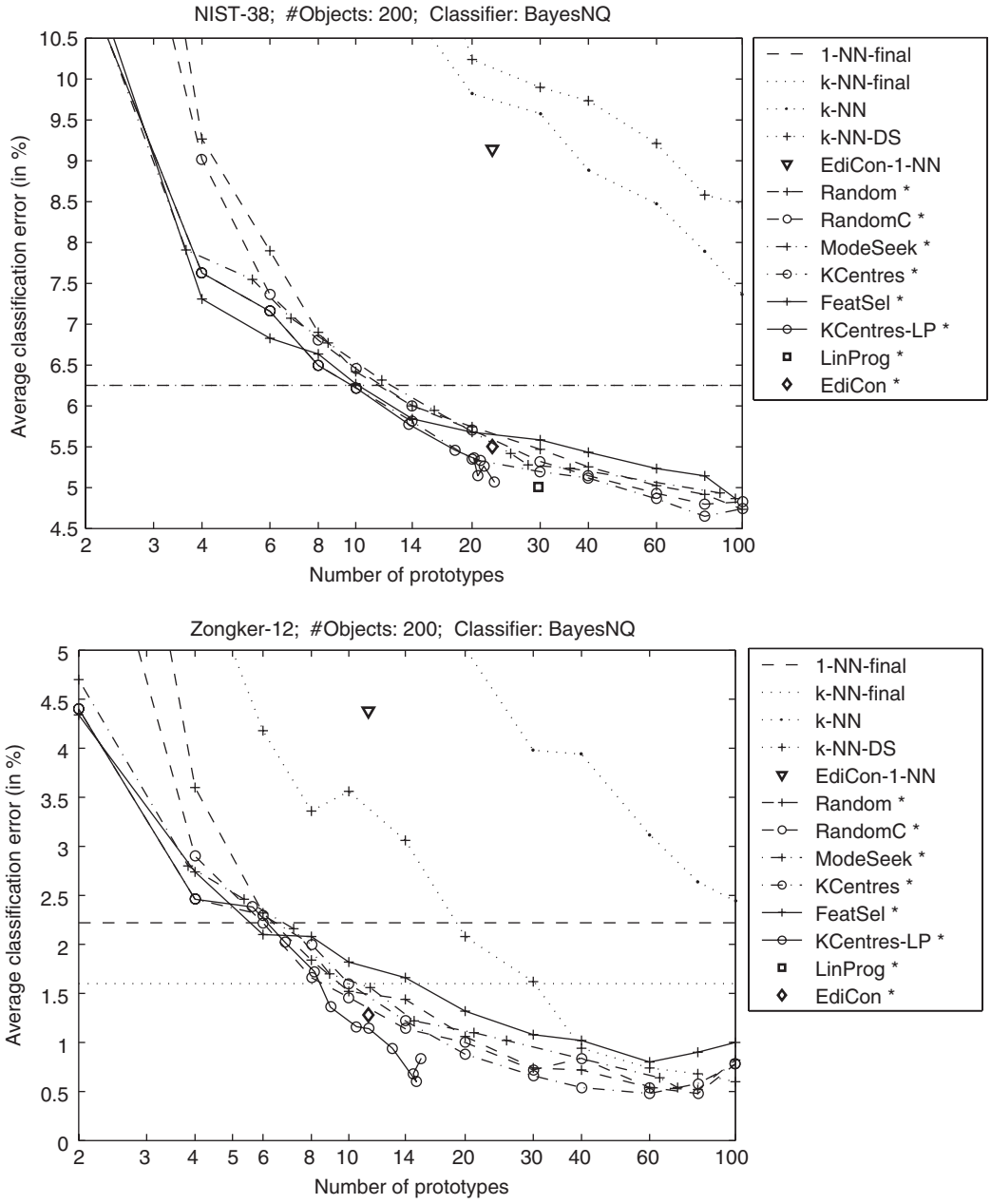

Fig. 13. NIST digit data. Average classification error of the *BayesNQ** and the *k*-NN classifiers in dissimilarity spaces as well as of the direct *k*-NN rule as a function of the number of selected prototypes.

prototypes to allow the BayesNQ to reach the result of the best *k*-NN. Moreover, for more prototypes, the NN is significantly outperformed by the BayesNQ.

So far, we have focused on two-class problems. In order to illustrate what may happen in multi-class situations, the following problems are also considered: the three-class *Wine* and *Ecoli* data, the four-class *ProDom* data and the ten-digit *Zongker-all* data. Although the *Wine* and *Ecoli* data are originally represented by features, their l_p distance representations can be used to show our point. In all the experiments with the BayesNQ classifier, a small regularization is used λ=0.01 (see Section 2.3). A regularization is necessary since for large representation sets, the number of training objects per class is insufficient for a proper estimation of the class covariance matrices. For instance, 100 training objects per class are used for the *Zongker-all* data. The results for more than 100 prototypes are based on the quadratic BayesNQ classifier trained in more than 100 dimensions. The peak for exactly 100 prototypes, see Fig. 19, upper plot, is caused by a dimension resonance phenomenon that has been fully

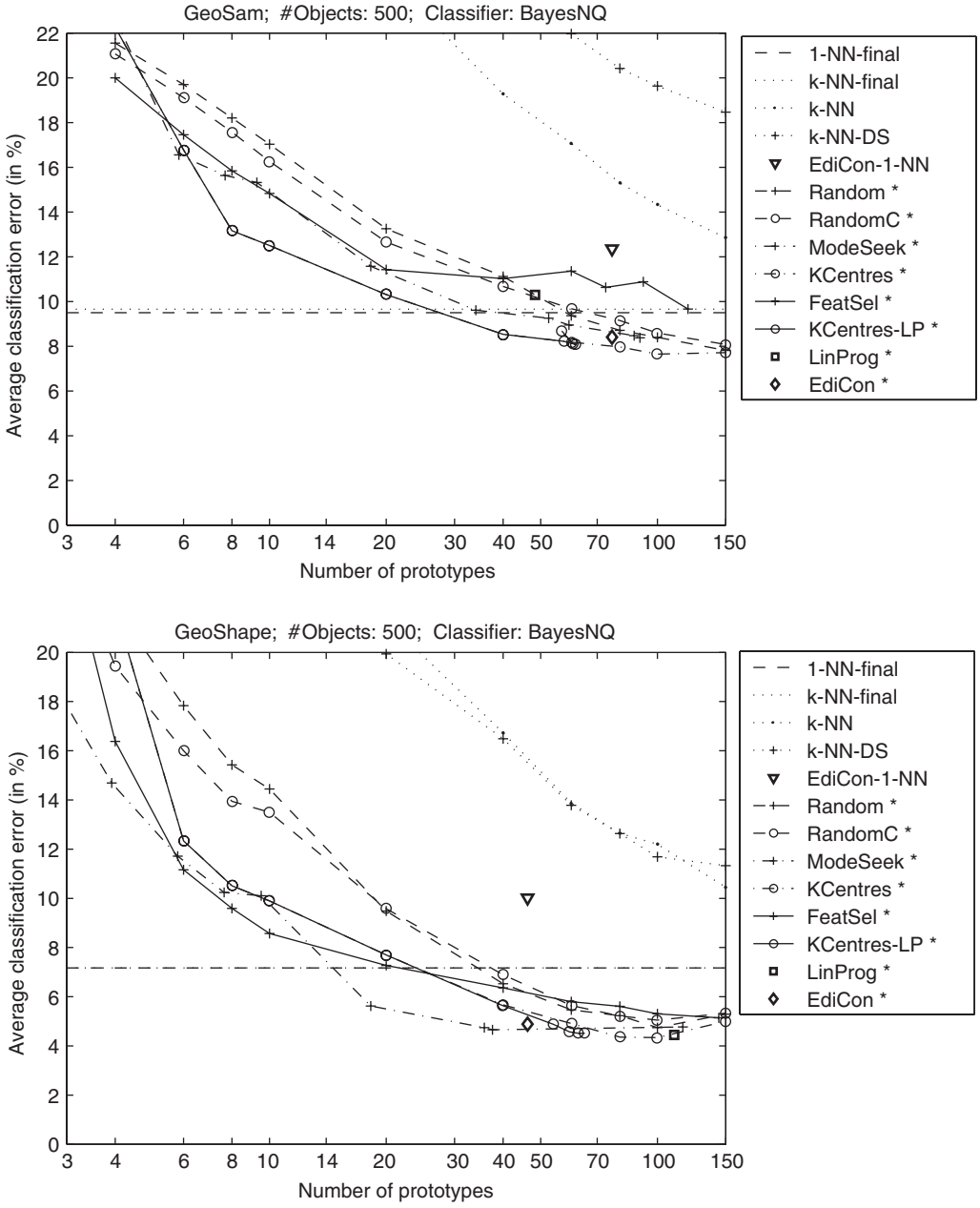

Fig. 14. Geophysical data. Average classification error of the *BayesNQ** and the *k*-NN classifiers in dissimilarity spaces as well as of the direct *k*-NN rule as a function of the number of selected prototypes.

examined for the linear normal density based classifier in [53]. When a larger regularization is used in this case, the BayesNQ performs much better, as can be seen in the bottom plot of the same figure.

Fig. 16 shows the results for the Euclidean representation of the *Wine* data. The *ModeSeek* seems to work the best, however since the number of test objects is small (70 in total), all the selection procedures behave similarly for more than 10 prototypes. The latter observation also holds for the *Ecoli-p1* and *Ecoli-p08* data, as observed in Fig. 17. The number of test objects is also small (107 in total). Here, however, the BayesNQ does not improve over the *k*-NN on the complete training set. Still, 20 (or less) prototypes are needed for the same performance.

Fig. 18 illustrates the study on prototype selection for the *ProDom* data. The data are multi-modal, as it can be judged from the 2D approximate embedding shown in Fig. 8. Some of the modes in the data seem to be very small, possibly some outliers. This may cause the *ModeSeek* procedure to focus on such examples, and be worse than the class-wise random selection. The *KCentres* and the *FeatSel* methods perform the best. For 100 (an more) prototypes, the BayesNQ

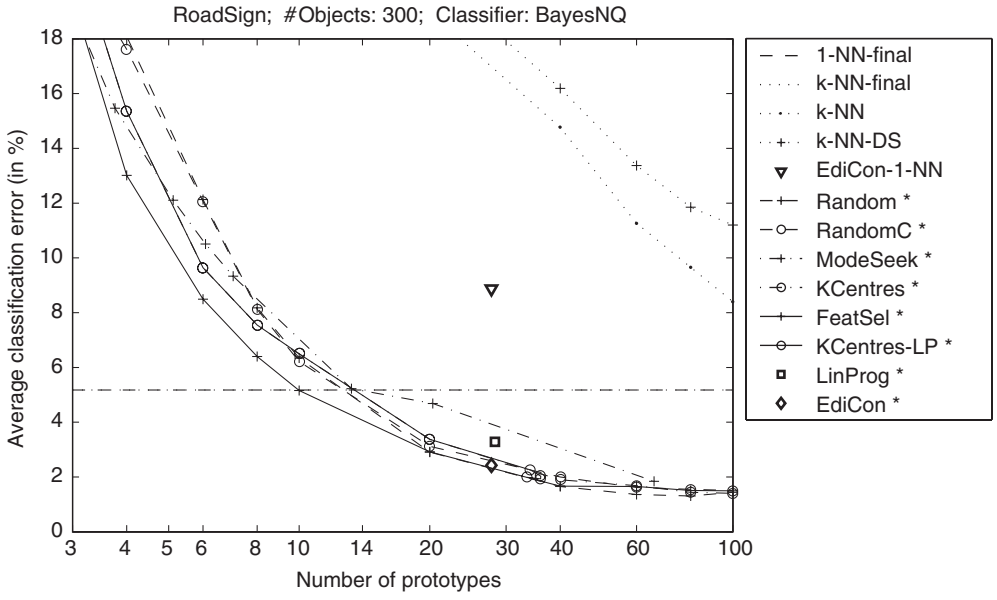

Fig. 15. Road sign data. Average classification error of the *BayesNQ** and the *k*-NN classifiers in dissimilarity spaces as well as of the direct *k*-NN rule as a function of the number of selected prototypes.

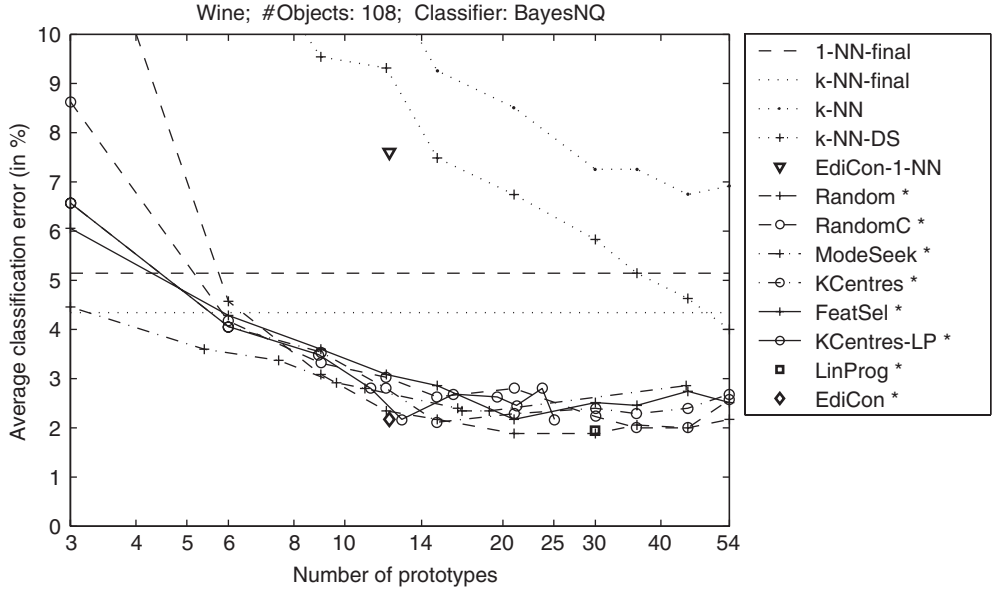

Fig. 16. Wine data. Average classification error of the *BayesNQ** and the *k*-NN classifiers in dissimilarity spaces as well as of the direct *k*-NN rule as a function of the number of selected prototypes.

reaches the error of the *k*-NN on a complete training set, however, it does not improve it. This might be partly caused by unequal class cardinalities and too-small regularization parameter.

The *Zongker-all* data are highly non-Euclidean and non-metric. When a proper regularization ($\lambda = 0.05$) is used, the BayesNQ classifier significantly outperforms the best *k*-NN rule. However, when the size of the representation set is too large (450 prototypes in bottom plot), the BayesNQ starts to suffer. Only 3% of the training examples allow this decision rule to reach the same performance as the *k*-NN rule on the entire training set. In general, the *KCentres* works the best. Edited and condensed set seems to give a good representation set, as well.

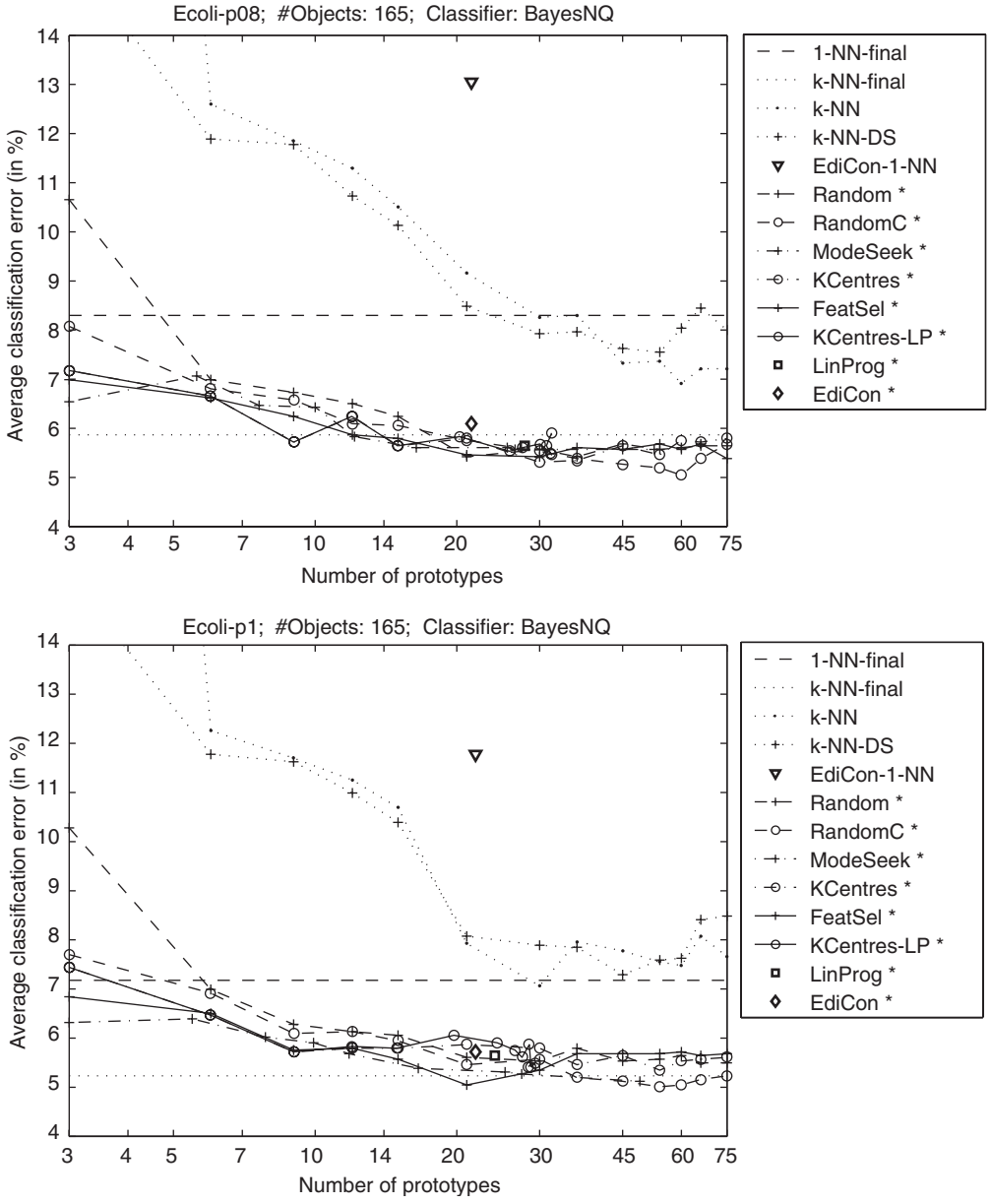

Fig. 17. Ecoli data. Average classification error of the *BayesNQ** and the *k*-NN classifiers in dissimilarity spaces as well as of the direct *k*-NN rule as a function of the number of selected prototypes.

Some observations are of interest for multi-class problems. First, in contrast to the two-class problems, a suitable regularization is necessary, since it can significantly influence the performance of the BayesNQ. If the regularization is appropriate, a significant improvement over the *k*-NN results on the complete training set may be found by the use of a regularized BayesNQ. Next, as in the two-class problems we find that just 3–12% of the training set gives a sufficient number of prototypes for the BayesNQ to reach the same performance as the *k*-NN rule. Like before, systematic selections of prototypes perform best. Finally, the *EdiCon* works well and tends to determine less prototypes than the *LinProg*.

In summary, we see that systematic selections perform better than the random selection, but the differences are sometimes small. The way we have ranked the algorithms in the legends from *Random* to *KCentres-LP*, roughly corresponds to the way they globally perform over the set of conducted experiments.

6. Discussion and conclusions

Prototype selection is an important topic for dissimilarity-based classification. By using a few, but well chosen

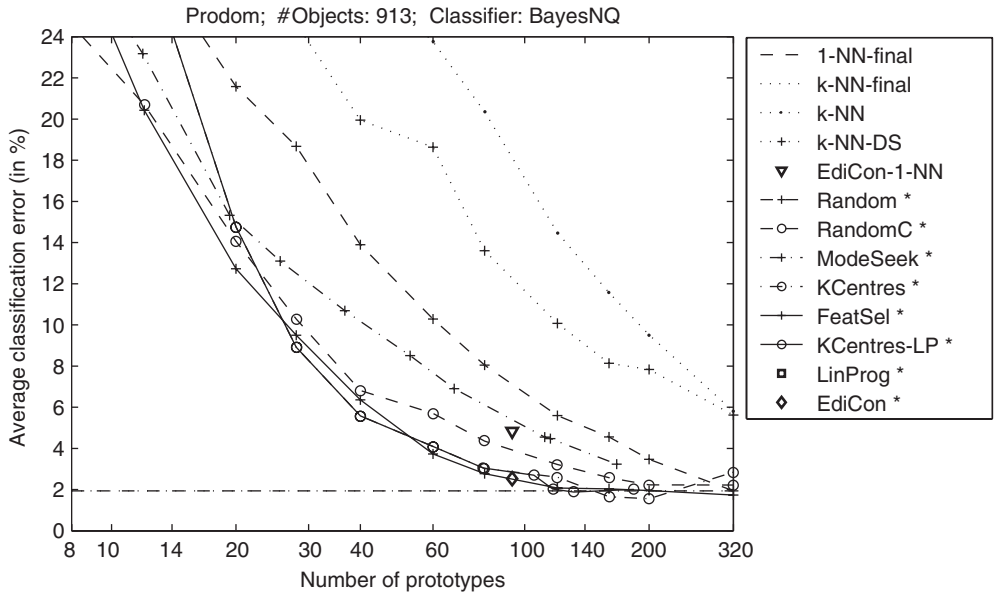

Fig. 18. Four-class *ProDom* problem. Average classification error of the *BayesNQ** and the *k*-NN classifiers in dissimilarity spaces, as well as the direct *k*-NN as a function of the selected prototypes. The result for the *LinProg* is not visible, since it finds a representation set of 491 objects.

prototypes, it is possible to achieve a better classification performance in both speed and accuracy than by using all the training samples. Usually, prototype selection methods are investigated in the context of the metric *k*-NN classification considered for feature-based representations. In our proposal, a dissimilarity representation $D(T, T)$ is interpreted as a vector space, where each dimension corresponds to a dissimilarity to an object from T. This allows us to construct traditional decision rules, such as linear or quadratic classifiers on such representations. Hence, the prototype selection relies on the selection of the representation set $R \subset T$ such that the chosen classifier performs well in a dissimilarity space $D(\cdot, R)$. Since the classifier is then trained on $D(T, R)$, a better accuracy can be reached than by using the *k*-NN rule defined on the set R.

In this paper, various selection procedures, both random and systematic, have been empirically investigated for the normal density based quadratic classifier (BayesNQ) built in dissimilarity spaces. The *k*-NN method, defined both on a complete training set T and a representation set R is used as a reference.

The following conclusions can be made from our study with respect to the investigated data sets:

(1) By building the BayesNQ in dissimilarity spaces just a very small number of prototypes (such as 3–12% of the training set size) is needed to obtain a similar performance as the *k*-NN rule on the entire training set.
(2) For large representation sets, consisting of, for instance 20% of the training examples, significantly better classification results are obtained for the BayesNQ than for the best *k*-NN. This holds for two-class problems and not necessarily for multi-class problems, unless a suitable regularization parameter is found.
(3) Overall, a systematic selection of prototypes does better than a random selection. Concerning the procedures which have a control over the number of selected prototypes, the KCentres procedure performs well, in general. In other cases, the linear programming performs well for two-class problems, while editing and condensing sets should be preferred for multi-class problems.

In our investigation, multi-class problems are more difficult as they need a proper regularization for the BayesNQ discrimination function. Moreover, this classifier becomes computationally more expensive. Therefore, there is a need for a further research to study more suitable classifiers and other prototype selection techniques for multi-class problems.

Acknowledgements

This work is supported by the Dutch Organization for Scientific Research (NWO) and the Dutch Organization for Applied Research (STW). We thank prof. Anil Jain and dr Zongker for providing the NIST dissimilarity data and dr Roth for the Prodom data.

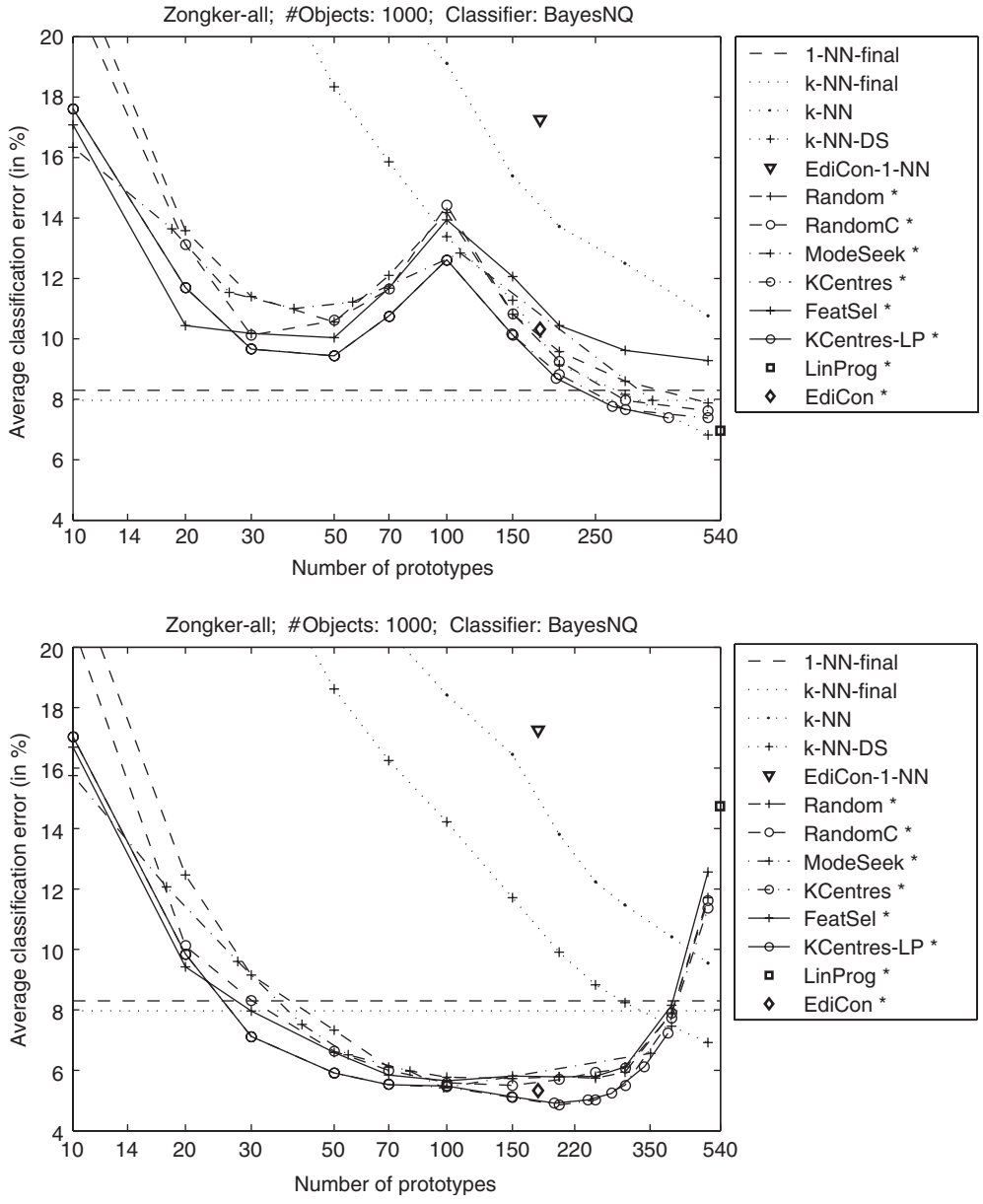

Fig. 19. Ten-class *Zongker-all* problem. Average classification error of the *BayesNQ**, with the regularization of $\lambda = 0.01$ (upper plot) and $\lambda = 0.05$ (bottom plot), and the k-NN classifiers in dissimilarity spaces as well as of the direct k-NN rule as a function of the number of selected prototypes.

References

[1] R.P.W. Duin, D. de Ridder, D.M.J. Tax, Featureless pattern classification, Kybernetika 34 (4) (1998) 399–404.

[2] R.P.W. Duin, E. Pękalska, P. Paclík, D.M.J. Tax, The dissimilarity representation, a basis for domain based pattern recognition? Pattern representation and the future of pattern recognition, ICPR 2004 Workshop Proceedings, Cambridge, United Kingdom, 2004, pp. 43–56.

[3] S. Edelman, Representation and Recognition in Vision, MIT Press, Cambridge, 1999.

[4] E. Pękalska, Dissimilarity representations in pattern recognition. Concepts, theory and applications, Ph.D. thesis, Delft University of Technology, ASCI Dissertation Series, 109, The Netherlands, January 2005.

[5] E. Pękalska, P. Paclík, R.P.W. Duin, A generalized Kernel approach to dissimilarity based classification, J. Mach. Learn. Res. 5 (2001) 175–211.

[6] E. Pękalska, R.P.W. Duin, Dissimilarity representations allow for building good classifiers, Pattern Recognition Lett. 23 (8) (2002) 943–956.

[7] P. Paclík, R.P.W. Duin, Dissimilarity-based classification of spectra: computational issues, Real Time Imaging 9 (4) (2003) 237–244.

[8] H. Bunke, S. Günter, X. Jiang, Towards bridging the gap between statistical and structural pattern recognition: two new concepts in graph matching, Conference on Advances in Pattern Recognition; LNCS 2013, 2001, pp. 1–11.

[9] B.V. Dasarathy (Ed.), Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques, Los Alamitos, USA, IEEE Computer Society Press, 1991.

[10] P.E. Hart, The condensed nearest neighbor rule, IEEE Trans. Inform. Theory 14 (5) (1968) 515–516.
[11] J.S. Sanchez, F. Pla, F.J. Ferri, Prototype selection for the nearest-neighbor rule through proximity graphs, Pattern Recognition Lett. 18 (6) (1997) 507–513.
[12] D.R. Wilson, T.R. Martinez, Reduction techniques for instance-based learning algorithms, Mach. Learn. 38 (3) (2000) 257–286.
[13] D.W. Jacobs, D. Weinshall, Y. Gdalyahu, Classification with non-metric distances: image retrieval and class representation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (6) (2000) 583–600.
[14] M.P. Dubuisson, A.K. Jain, Modified hausdorff distance for object matching, Proceedings of the International Conference on Pattern Recognition, vol. 1, 1994, pp. 566–568.
[15] E. Pękalska, R.P.W. Duin, S. Günter, H. Bunke, On not making dissimilarities Euclidean, Proceedings of Structural and Statistical Pattern Recognition, Lisbon, Portugal, 2004, pp. 1143–1151.
[16] P. Paclík, R.P.W. Duin, Classifying spectral data using relational representation, Proceedings of the Spectral Imaging Workshop, Graz, Austria, 2003.
[17] T.M. Cover, P.E. Hart, Nearest neighbor pattern classification, IEEE Trans. Inform. Theory 13 (1) (1967) 21–27.
[18] L. Devroye, L. Györfi, G. Lugosi, A Probabilistic Theory of Pattern Recognition, Springer, Berlin, 1996.
[19] R. Paredes, E. Vidal, A class-dependent weighted dissimilarity measure for nearest neighbor classification problems, Pattern Recognition Lett. 21 (12) (2000) 1027–1036.
[20] J.S. Sanchez, F. Pla, F.J. Ferri, Improving the k-NN classification rule through heuristic modifications, Pattern Recognition Lett. 19 (13) (1998) 1165–1170.
[21] D.R. Wilson, T.R. Martinez, Improved heterogeneous distance functions, J. Artif. Intell. Res. 6 (1–34) (1997).
[22] T. Hastie, R. Tibshirani, Discriminant adaptive nearest neighbor classification, IEEE Trans. Pattern Anal. Mach. Intell. 18 (6) (1996) 607–616.
[23] C. Domeniconi, J. Peng, D. Gunopulos, Locally adaptive metric nearest-neighbor classification, IEEE Trans. Pattern Anal. Mach. Intell. 24 (9) (2002) 1281–1285.
[24] P. Avesani, E. Blanzieri, F. Ricci, Advanced metrics for class-driven similarity search, proceedings of the International Workshop on Database and Expert Systems Applications, Italy, 1999, pp. 223–227.
[25] D.G. Lowe, Similarity metric learning for a variable-Kernel classifier, Neural Comput. 7 (1) (1995) 72–85.
[26] J.B. MacQueen, Some methods for classification and analysis of multivariate observations, Berkeley Symposium on Mathematical Statistics and Probability, 1967, pp. 281–297.
[27] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, second ed., Wiley, New York, 2001.
[28] T. Kohonen, Self-Organizing Maps, Springer, Germany, 1995.
[29] Y.S. Huang, C.C. Chiang, J.W. Shieh, W.E.L. Grimson, Prototype optimization for nearest-neighbor classification, Pattern Recognition 35 (6) (2002) 1237–1245.
[30] R.A. Mollineda, F.J. Ferri, E. Vidal, An efficient prototype merging strategy for the condensed 1-NN rule through class-conditional hierarchical clustering, Pattern Recognition 35 (12) (2002) 2771–2782.
[31] P.A. Devijver, J. Kittler, Pattern Recognition: A Statistical Approach, Prentice-Hall, London, 1982.
[32] S. Berchtold, B. Ertl, D.A. Keim, H.-P. Kriegel, T. Seidl, Fast nearest neighbor search in high-dimensional spaces, in: Proceedings of the 14th International Conference on Data Engineering, Orlando, Florida, 1998.
[33] P.J. Grother, G.T. Candela, J.L. Blue, Fast implementations of nearest-neighbor classifiers, Pattern Recognition 30 (3) (1997) 459–465.
[34] J. McNames, A Fast nearest-neighbor algorithm based on a principal axis search tree, IEEE Trans. Pattern Anal. Mach. Intell. 23 (9) (2001) 964–976.
[35] L. Micó, J. Oncina, R.C. Carrasco, A fast branch & bound nearest neighbour classifier in metric spaces, Pattern Recognition Lett. 17 (7) (1996) 731–739.
[36] L. Micó, J. Oncina, Comparison of fast nearest neighbour classifiers for handwritten character recognition, Pattern Recognition Lett. 19 (3–4) (1998) 351–356.
[37] F. Moreno-Seco, L. Micó, J. Oncina, A modification of the LAESA algorithm for approximated k-NN classification, Pattern Recognition Lett. 24 (1–3) (2003) 47–53.
[38] V. Ramasubramanian, K.K. Paliwal, Fast nearest-neighbor search algorithms based on approximation-elimination search, Pattern Recognition 33 (9) (2000) 1497–1510.
[39] R.P.W. Duin, P. Juszczak, D. de Ridder, P. Paclík, E. Pękalska, D.M.J. Tax, PR-Tools, a Matlab toolbox for pattern recognition, http://www.prtools.org, 2004.
[40] Y. Cheng, Mean shift, mode seeking, and clustering, IEEE Trans. Pattern Anal. Mach. Intell. 17 (8) (1995) 790–799.
[41] A. Jain, D. Zongker, Feature selection: evaluation, application, and small sample performance, IEEE Trans. Pattern Anal. Mach. Intell. 19 (12) (1997) 153–158.
[42] T. Graepel, R. Herbrich, B. Schölkopf, A. Smola, P. Bartlett, K.-R. Müller, K. Obermayer, R. Williamson, Classification on Proximity Data with LP-Machine, Proceedings of the International Conference on Artificial Neural Networks, 1999, pp. 304–309.
[43] P.S. Bradley, O.L. Mangasarian, W.N. Street, Feature selection via mathematical programming, INFORMS J. Comput. 10 (2) (1998) 209–217.
[44] L. Goldfarb, A New Approach to Pattern Recognition, in: L.N. Kanal, A. Rosenfeld (Eds.), Progress in Pattern Recognition, vol. 2, Elsevier Science Publishers BV, 1985, pp. 241–402.
[45] C.L. Wilson, M.D. Garris, Handprinted character database 3, National Institute of Standards and Technology, Advanced Systems division, 1992.
[46] A.K. Jain, D. Zongker, Representation and recognition of handwritten digits using deformable templates, IEEE Trans. Pattern Anal. Mach. Intell. 19 (12) (1997) 1386–1391.
[47] D. Landgrebe, Signal Theory Methods in Multispectral Remote Sensing, Wiley, New York, 2003.
[48] P. Paclík, J. Novovičová, P. Somol, P. Pudil, Road sign classification using Laplace Kernel classifier, Pattern Recognition Lett. 21 (13–14) (2000) 1165–1173.
[49] C.L. Blake, C.J. Merz, UCI Repository of machine learning databases, http://www.ics.uci.edu/m˜learn/MLRepository.html.
[50] F. Corpet, F. Servant, J. Gouzy, D. Kahn, ProDom and ProDom-CG: tools for protein domain analysis and whole genome comparisons, Nucleid Acids Res. 28 (2000) 267–269.
[51] V. Roth, J. Laub, J.M. Buhmann, K.-R. Müller, Going metric: denoising pairwise data, Advances in Neural Information Processing Systems, MIT Press, 2003, pp. 841–856.
[52] A.G. Murzin, S.E. Brenner, T. Hubbard, C. Chothia, SCOP: a structural classification of proteins database for the investigation of sequences and structures, J. Mol. Biol. 247 (1995) 536–540.
[53] S. Raudys, R.P.W. Duin, Expected classification error of the Fisher linear classifier with pseudo-inverse covariance matrix, Pattern Recognition Lett. 19 (5–6) (1998) 385–392.