Available online at www.sciencedirect.com
ScienceDirect
Pattern Recognition 40 (2007) 2946–2957
PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/pr

On using prototype reduction schemes to optimize dissimilarity-based classification☆

Sang-Woon Kim[a], B. John Oommen[b,*,1]

[a]*Department of Computer Science and Engineering, Myongji University, Yongin 449-728, Republic of Korea*
[b]*School of Computer Science, Carleton University, Ottawa, Canada K1S 5B6*

Received 21 February 2006; accepted 1 March 2007

Abstract

The aim of this paper is to present a strategy by which a new philosophy for pattern classification, namely that pertaining to dissimilarity-based classifiers (DBCs), can be efficiently implemented. This methodology, proposed by Duin and his co-authors (see Refs. [Experiments with a featureless approach to pattern recognition, Pattern Recognition Lett. 18 (1997) 1159–1166; Relational discriminant analysis, Pattern Recognition Lett. 20 (1999) 1175–1181; Dissimilarity representations allow for buillding good classifiers, Pattern Recognition Lett. 23 (2002) 943–956; Dissimilarity representations in pattern recognition, Concepts, theory and applications, Ph.D. Thesis, Delft University of Technology, Delft, The Netherlands, 2005; Prototype selection for dissimilarity-based classifiers, Pattern Recognition 39 (2006) 189–208]), is a way of defining classifiers between the classes, and is not based on the feature measurements of the individual patterns, but rather on a suitable dissimilarity measure between them. The advantage of this methodology is that since it does not operate on the class-conditional distributions, the accuracy can exceed the Bayes' error bound. The problem with this strategy is, however, the need to compute, store and process the inter-pattern dissimilarities for all the training samples, and thus, the accuracy of the classifier designed in the dissimilarity space is dependent on the methods used to achieve this. In this paper, we suggest a novel strategy to enhance the computation for all families of DBCs. Rather than compute, store and process the DBC based on the entire data set, we advocate that the training set be first reduced into a smaller representative subset. Also, rather than determine this subset on the basis of random selection, or clustering, etc., we advocate the use of a prototype reduction scheme (PRS), whose output yields the points to be utilized by the DBC. The rationale for this is explained in the paper. Apart from utilizing PRSs, in the paper we also propose simultaneously employing the Mahalanobis distance as the dissimilarity-measurement criterion to increase the DBCs classification accuracy. Our experimental results demonstrate that the proposed mechanism increases the classification accuracy when compared with the "conventional" approaches for samples involving real-life as well as artificial data sets—*even though the resulting dissimilarity criterion is not symmetric*.

Keywords: Dissimilarity representation; Dissimilarity-based classification; Prototype reduction schemes (PRSs); Mahalanobis distances (MDs)

1. Introduction

The field of statistical pattern recognition[2] (PR) [1,2] has matured since its infancy in the 1950s, and the aspiration to

☆ The work of the second author was done while visiting at Myongji University, Yongin, Korea. The second author was partially supported by NSERC, the Natural Sciences and Engineering Research Council of Canada, and a grant from the Korean Research Foundation. This work was generously supported by the Korea Research Foundation Grant funded by the Korea Government (MOEHRD-KRF-2005-042-D00265). A preliminary version of this paper was presented at the ICIAR-2006, the 2006 International Conference on Image Analysis and Recognition, in Povoa de Varzim, Portugal, in September 2006.

* Corresponding author. Tel.: +1 613 520 2600.
E-mail addresses: kimsw@mju.ac.kr (S.-W. Kim), oommen@scs.carleton.ca (B.J. Oommen).

[1] This author is also an Adjunct Professor with the Department of Information and Communication Technology, Agder University College, Grimstad, Norway.

[2] Throughout this paper, we assume that we are working with the following general model of statistical PR systems [1,2]. The system is provided with a set of data represented in terms of its feature measurements, and the feature *vector* is a conjunction of these measured values for the sample considered. Given a set of so-called "training samples", the PR system builds a classifier which is, for example, of a parametric form, or of a non-parametric form. Subsequently, a sample to be tested is provided by means of *its* feature measurements and appropriately classified. The quality of the classifier is measured by quantifying the accuracy by which these samples are classified.

0031-3203/$30.00 © 2007 Pattern Recognition Society. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2007.03.006

enlarge the horizons has led to numerous philosophically new avenues of research. The fundamental questions tackled involve (among others) increasing the accuracy of the classifier system, minimizing the time required for training and testing, reducing the effects of the curse of dimensionality, and reducing the effects of the peculiarities of the data distributions.

One of the most recent novel developments in this field is the concept of dissimilarity-based classifiers (DBCs) proposed by Duin and his co-authors (see Refs. [3–6,8]). Philosophically, the motivation for DBCs is the following: if we assume that "Similar" objects can be grouped together to form a class, a "class" is nothing more than a set of these "similar" objects. Based on this idea, Duin and his colleagues argue that the notion of proximity (similarity or dissimilarity) is actually more fundamental than that of a feature or a class. Indeed, it is probably more likely that the brain uses an intuitive DBC-based methodology than that of taking measurements, inverting matrices, etc. Thus, DBCs are a way of defining classifiers between the classes, which are not based on the feature measurements of the individual patterns, but rather on a suitable *dissimilarity measure* between them. The advantage of this methodology is that since it does not operate on the class-conditional distributions, the accuracy can exceed the Bayes' error bound—which is, in our opinion, remarkable.[3] Another salient advantage of such a paradigm is that it does not have to confront the problems associated with feature spaces such as the "curse of dimensionality", and the issue of estimating a large number of parameters. The problem with this strategy is, however, the need to compute, store and process the inter-pattern dissimilarities for (in the worst case) all the training samples, and thus, the accuracy of the classifier designed in the dissimilarity space is dependent on the methods used to achieve this.

A dissimilarity representation of a set of samples, $T = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, is based on pairwise comparisons and is expressed, for example, as an $n \times n$ dissimilarity matrix[4] $D_{T,T}[\cdot, \cdot]$, where the subscripts of D represent the set of elements on which the dissimilarities are evaluated. Thus, each entry $D_{T,T}[i, j]$ corresponds to the dissimilarity between the pairs of objects $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$, $\boldsymbol{x}_i, \boldsymbol{x}_j \in T$. When it concerns testing any object in the sample space, $\boldsymbol{x}$, the latter is represented by a vector of proximities $\delta(\boldsymbol{x}, Z)$ to the objects in a specific set Z, which is used for the testing purposes. Thus, if $\boldsymbol{x} = \boldsymbol{x}_i$, and $Z = T$, $\delta(\boldsymbol{x}_i, T)$ is the ith row of $D_{T,T}[\cdot, \cdot]$. The principle behind DBCs is that a new (testing) sample $\boldsymbol{z}$, if represented by $\delta(\boldsymbol{z}, T)$, is classified to a specific class if it is sufficiently similar to one or more objects within that class.

The problem we study in this paper deals with how DBCs between classes represented in this manner can be effectively computed. The *families* of strategies investigated in this endeavour are many. First of all, by selecting a set of prototypes or support vectors, the problem of dimension reduction can be drastically simplified. In order to select such a representative set from the training set, the authors of Ref. [4] discuss a number of methods such as random selections, the k-centres method, and others which will be cataloged presently. Alternatively, some work has also gone into the area of determining appropriate measures of dissimilarity using measures such as various L_p norms (including the Euclidean and $L_{0.8}$), the Hausdorff and modified Hausdorff norm, and some traditional PR-based measures such as those used in template matching, and correlation-based analysis. These too which will be listed presently.[5]

Rationale for this paper: In this paper we propose to utilize a prototype reduction scheme (PRS) as a method for selecting the prototype vectors required for the dissimilarity representation. We also advocate the specific use of the Mahalanobis distance as a dissimilarity measure to design the DBC. To be more specific, with regard to the former, we propose to increase the performance of a DBC by systematically selecting or *creating* a representative set without sacrificing the performance, where the latter is achieved by using a PRS. The interesting feature of using the Mahalanobis distance is the following: the resulting dissimilarity criterion underlying the DBC becomes *asymmetric*. The curious result is that the DBC works expediently even though this criterion is violated. The question of why this occurs remains open.

Over the years, PRSs have been explored for various purposes, and this has resulted in the development of many algorithms[6] [9,10]. One of the first of its kind, was a method that led to a smaller prototype set, the condensed nearest neighbour (CNN) rule [11]. Since the development of the CNN, other methods have been proposed successively, such as the prototypes for nearest neighbour (PNN) classifiers [14] (including a modified Chang's method proposed by Bezdek [15]), the vector quantization (VQ) technique [16], etc. The method of obtaining improved prototypes by invoking a SVM augmented with LVQ3-type adjusting has also been reported [20], and a brief survey of these methods will be given in Section 3.

Format of the paper: This paper is organized as follows: To pose the problem in the right perspective, we first, in Section 2 summarize the state-of-the-art of DBC methods, following which we present an overview of the existing PRSs in Section 3. After this, in Section 4 we propose our contributions to design DBCs by incorporating the use of PRSs and the Mahalanobis distance. Experimental results for artificial and real-life benchmark data sets are provided in Section 5. Finally, Section 6 concludes the paper.

[3] In our opinion, the theory of DBCs is one of the major contributions to the field of statistical PR in the last decade. Duin and his colleagues [3] have ventured to call this paradigm a *featureless* approach to PR by insisting that there is a clear distinction between feature-based and non-feature-based approaches. We anticipate that a lot of new research energy will be expended in this direction in the future.

[4] If the dissimilarity is not stored, but rather computed when needed, it would be more appropriate to regard it as a *function* $D_{T,T}(\cdot, \cdot)$.

[5] For want of a better term, DBCs enhanced with these prototype selection methods and the latter distance measures will be referred to as "conventional" schemes.

[6] Bezdek et al. [9], who have composed an excellent survey of the field, report that there are "zillions!" of methods for finding prototypes (see [9, p. 1459]).

1.1. Contributions of the Paper

We claim two modest contributions in this paper:

1. First of all, we show that a PRS can also be used as a *tool* to achieve an intermediate goal, namely, to minimize the number of samples that are *subsequently* used in any DBC system. This subset, will, *in turn* be utilized to design the classifier—which, as we shall argue, must be done in conjunction with an appropriate dissimilarity measure. This, in itself, is novel to the field when it concerns the *applications of PRSs*, which have typically been used as methods by which prototypes have been selected or created to design of PR systems.
2. The second contribution of this paper is the fact that we have shown that using second-order distance measures, such the Mahalanobis distance, together with appropriate PRS, has a distinct advantage when they are used to implement the DBC. Amazingly enough, this is true even though the resulting dissimilarity criterion underlying the DBC is rendered *asymmetric*.

The conclusions are based on rigorous tests done on both established benchmark artificial and real-life data sets.

2. Dissimilarity-based classification

2.1. Foundations of DBCs

Let $T = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \in \mathscr{R}^p$ be a set of n feature vectors in a p-dimensional space. We assume that T is a labelled data set, so that T can be decomposed into, say, c subsets $\{T_1, \ldots, T_c\}$ such that

1. $T_i = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n_i}\}$, with $n = \sum_{i=1}^{c} n_i$.
2. $T = \bigcup_{k=1}^{c} T_k$.
3. $T_i \cap T_j = \phi, \forall i \neq j$.

Our goal is to design a DBC in an appropriate dissimilarity space constructed with this *training data* set, and to classify an input sample $\boldsymbol{z}$ into an appropriate class.

To achieve this, we assume that from T_i, the training data of class ω_i, we extract a prototype set,[7] Y_i, where

$$Y_i = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{m_i}\}, \quad m = \sum_{i=1}^{c} m_i. \tag{1}$$

Every DBC assumes the use of a dissimilarity measure, d, computed from the samples, where, $d(\boldsymbol{x}_i, \boldsymbol{y}_j)$ represents the dissimilarity between two samples $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$. The measure, d, is required to be non-negative, reflexive and symmetric,[8] and thus,

$$d(\boldsymbol{x}_i, \boldsymbol{y}_j) \geqslant 0 \quad \text{with } d(\boldsymbol{x}_i, \boldsymbol{y}_j) = 0 \text{ if } \boldsymbol{x}_i = \boldsymbol{y}_j, \quad \text{and}$$
$$d(\boldsymbol{x}_i, \boldsymbol{y}_j) = d(\boldsymbol{y}_j, \boldsymbol{x}_i).$$

The dissimilarity computed between T and Y leads to a $n \times m$ matrix, $D_{T,Y}[i, j]$, where $\boldsymbol{x}_i \in T$ and $\boldsymbol{y}_j \in Y$. Consequently, an object $\boldsymbol{x}_i$ is represented as a column vector as following:

$$[d(\boldsymbol{x}_i, \boldsymbol{y}_1), d(\boldsymbol{x}_i, \boldsymbol{y}_2), \ldots, d(\boldsymbol{x}_i, \boldsymbol{y}_m)]^T, \quad 1 \leqslant i \leqslant n. \tag{2}$$

Here, we define the dissimilarity matrix $D_{T,Y}[\cdot, \cdot]$ as a *dissimilarity space* on which the p-dimensional object, $\boldsymbol{x}$, given in the feature space, is represented as an m-dimensional vector $\delta(\boldsymbol{x}, Y)$, where if $\boldsymbol{x} = \boldsymbol{x}_i$, $\delta(\boldsymbol{x}_i, Y)$ is the ith row of $D_{T,Y}[\cdot, \cdot]$. In this paper, the column vector $\delta(\boldsymbol{x}, Y)$ is simply denoted by $\delta_Y(\boldsymbol{x})$, where the latter is an m-dimensional vector, while $\boldsymbol{x}$ is p-dimensional.

For a training set $\{\boldsymbol{x}_i\}_{i=1}^{n}$, and an evaluation sample $\boldsymbol{z}$, the modified training set and sample now become $\{\delta_Y(\boldsymbol{x}_i)\}_{i=1}^{n}$ and $\delta_Y(\boldsymbol{z})$, respectively. From this perspective, we can see that the dissimilarity representation can be considered as a *mapping* by which any arbitrary $\boldsymbol{x}$ is translated into $\delta_Y(\boldsymbol{x})$, and thus, if m is selected sufficiently small (i.e., $m \ll p$), we are essentially working in a space with much smaller dimensions. The literature reports the use of many traditional decision classifiers including k-NN rule and the linear/quadratic normal-density-based classifiers to the task of classifying $\boldsymbol{z}$ using $\delta_Y(\boldsymbol{z})$ in the dissimilarity space.

2.2. Prototype selection methods for DBCs

We first consider the reported methods [6–8] by which each T_i is pruned to yield a set of representative *prototypes*, Y_i, where without loss of generality $|Y_i| < |T_i|$. The intention is to guarantee a good tradeoff between the recognition accuracy and the computational complexity when the DBC is built on $D_{T,Y}(\cdot, \cdot)$ rather than $D_{T,T}(\cdot, \cdot)$. The reported comparison of Ref. [8] has been performed from the perspective of the resultant error rates and the number of prototypes obtained. The experiments were conducted with seven artificial and real-life data sets. Eight selection methods employed for the experiments were *Random*, *RandomC*, *KCentres*, *ModeSeek*, *LinProg*, *PeatSeal*, *KCentres-LP*, and *EdiCon*. In the interest of completeness, we briefly explain below (using the notation introduced above) the methods that are pertinent to our present study.

1. *Random*: This method involves a *random* selection of m samples from the training data set T.
2. *RandomC*: This method involves a random selection of m_i samples per class, ω_i, from T_i.
3. *KCentres*: This method[9] consists of a procedure that is applied to each class separately. For each class ω_i, the algorithm is invoked so as to choose m_i samples

[7] Since we are invoking a PRS to obtain Y_i from T_i, we do not require that $Y_i \subseteq T_i$. Rather Y_i may be created or selected from T_i, and its computation may also involve the other sets, T_j, $j \neq i$.

[8] Note that $d(\cdot, \cdot)$ need not be a *metric* [8]. However, we shall show experimentally that we can relax the condition of symmetry and still have a very good DBC.

[9] This procedure is essentially identical to the k-means clustering algorithm performed in a vector space, and is thus heavily dependent on the initialization.

which are "evenly" distributed with respect to the dissimilarity matrix $D_{T_i,T_i}[\cdot,\cdot]$. The algorithm can be summarized as follows:

(a) Select an initial set $Y_i = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{m_i}\}$ consisting of m_i objects, e.g., randomly chosen from T_i.
(b) For each $\boldsymbol{x} \in T_i$, find its nearest neighbour in Y_i. Let N_j, $j = 1, \ldots, m_i$, be a subset of T_i consisting of objects that yield the same nearest neighbour $\boldsymbol{y}_j$ in Y_i. This means that $T_i = \bigcup_{j=1}^{m_i} N_j$.
(c) For each N_j, find its centre $\boldsymbol{c}_j$, which is the object for which the maximum distance to all other objects in N_j is minimum (this value is called the radius of N_j).
(d) For each centre $\boldsymbol{c}_j$, if $\boldsymbol{c}_j \neq \boldsymbol{y}_j$, then replace $\boldsymbol{y}_j$ by $\boldsymbol{c}_j$ in Y_i. If any replacement is done, then return to Step (b). Otherwise exit.
(e) Return the final representation set Y which consists of all the final sets Y_i.

From the experimental results of Ref. [8], the authors seem to have deliberated that systematic approaches lead to better results than those which rely on random selection, especially when the number of prototypes is small. Furthermore, although there is no single winner (inasmuch as the results depend on the characteristics of the data), they indicate that, in general, the KCentres works well.

The details of the other methods such as *ModeSeek*, *FeatSel*, *LinProg*, *KCentres-LP*, and *EdiCon* are omitted here as they are not directly related to the premise of our work. Whereas our present work and the above three methods pursue a pruning in the *original feature space*, the methods omitted here attempt the same in the *dissimilarity space*. The details of these methods can be found in Ref. [8].

2.3. Dissimilarity measures used in DBCs

Fundamental to DBCs is the measure used to quantify the dissimilarity between two vectors.[10] The work in Ref. [8] reports extensive experiments conducted using various dissimilarity measures (see Ref. [8, Table 2]). A list of these measures where we quantify the dissimilarity between $\boldsymbol{v}$ and $\boldsymbol{w} \in \mathscr{R}^q$ is given below:

1. *City Block Norm* : $D_1 = \sum_{i=1}^{q} |v_i - w_i|$.
2. *Euclidean Norm* : $D_E (\text{or } D_2) = \sqrt{(\boldsymbol{v}-\boldsymbol{w})^T(\boldsymbol{v}-\boldsymbol{w})}$.
3. *Max Norm* : $D_{max} = \text{Max}_i |v_i - w_i|$.
4. L_p or *Minkowski Norm* : $D_p = (\sum_{i=1}^{q} |v_i - w_i|^p)^{1/p}$, $p \geqslant 1$, $p \neq 2$.
5. *Hausdorff Norm* :

$$h(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\|.$$

The measures which were tested in Ref. [8] essentially fall into three categories: (a) the City Block, $L_{0.8}$, Euclidean, and Max Norm, which are special cases of the L_p metric for $p = 1, 0.8, 2$ and ∞, respectively, (b) the Hausdorff Norm, and its variants, which involve Max–Min computations, and (c) traditional PR norms such as the template matching and correlation norms. The details of the other measures such as the *Median* and *Cosine*, are omitted here in the interest of compactness, but can be found in Ref. [6].

2.4. State-of-the-art DBC optimization

Based on the above, in all brevity, we state that the state-of-the-art strategy applicable for optimizing DBCs involves the following steps:

1. Select the representative set, Y, from the training set T by resorting to one of the methods given in Section 2.2.
2. Compute the dissimilarity matrix $D_{T,Y}[\cdot,\cdot]$, using Eq. (2), in which each individual dissimilarity is computed using one of the measures described in Section 2.3.
3. For a testing sample z, compute a dissimilarity column vector, $\delta_Y(z)$, by using the same measure used in Step 2 above.
4. Achieve the classification based on invoking a classifier built in the dissimilarity space and operating on the dissimilarity vector $\delta_Y(z)$.

3. State-of-the-art prototype reduction schemes

In non-parametric pattern classification which use the NN or the k-NN rule, each class is described using a set of sample prototypes, and the class of an unknown vector is decided based on the identity of the closest neighbour(s) which are found among all the prototypes [2]. To enhance computation, it is advantageous to reduce the number of training vectors while simultaneously insisting that the classifiers that are built on the reduced design set perform as well, or nearly as well, as the classifiers built on the original data set.

Various PRSs,[11] which are useful in nearest-neighbour-like classification, have been reported in the literature—two excellent surveys are found in Refs. [9,10]. Bezdek et al. [9], who composed the second and more recent survey of the field, reported that there are "zillions!" of methods for finding prototypes (see Ref. [9, p. 1459]). Rather than embark on yet another survey of the field, we mention here a *few* representative methods of the "zillions" that have been reported. PRSs have been explored for various purposes and has resulted in the development of many algorithms [9,10].

One of the first of its kind was a method that led to a smaller prototype set, the CNN rule [11]. Since the development of the CNN, other methods [12–22] have been proposed successively, such as the PNN classifiers [14] (including a modified Chang's method proposed by Bezdek [15]), the VQ technique [16], etc. Apart from the above PRS methods, the support vector machines (SVMs) [17,18] can also be used as a means of

[10] The details of the binary, categorical, ordinal, symbolic and quantitative features are omitted here, but can be found in Ref. [8].

[11] Our overview is necessarily brief.

selecting prototype vectors. Observe that these new vectors can be subsequently adjusted by means of an LVQ3-type algorithm. Based on this idea, a new prototype reduction method (referred to here as HYB) of hybridizing the SVM and the LVQ3 was introduced in Ref. [20]. The reduced set produced by the CNN, however, customarily includes "interior" samples, which can be completely eliminated, without altering the performance of the resultant classifier. Accordingly, other methods have been proposed successively, such as the reduced nearest neighbour (RNN) rule [12], the PNN classifiers [14], the selective nearest neighbour (SNN) rule, two modifications of the CNN, the edited nearest neighbour (ENN) rule (see Ref. [21] for more details of these) and the non-parametric data reduction method [1,13]. Besides these, in Ref. [16], the VQ and Bootstrap techniques have also been reported as being extremely effective approaches to data reduction.

In designing NN classifiers, however, it seems to be intuitively true that prototypes near the separating boundary between the classes play more important roles than those which are more interior in the feature space. In creating or selecting prototypes, vectors near the boundaries between the classes have to be considered to be more significant, and the created prototypes need to be moved (or adjusted) towards the classification boundaries so as to yield a higher performance. Based on this philosophy, namely, that of *selecting* and *adjusting* the reduced prototypes, we recently proposed a new hybrid approach that involved two distinct phases [20,21]. In the first phase, initial prototypes are *selected* or *created* by any of the conventional reduction methods mentioned earlier. After this selection/creation phase, the technique in Refs. [20,21] suggests a second phase in which the proposed reduced prototypes are migrated to their "optimal" positions by *adjusting* them by invoking an LVQ3-type *learning* scheme. The relative advantages of the scheme in Refs. [20,21] have been demonstrated on both artificial and real-life data sets.

All the PRS methods reported in the literature [9,10], (including the one proposed in Refs. [20,21]) are practical as long as the size of the data set is not "too large". The applicability of these schemes for large-sized data sets is limited because they all suffer from a major disadvantage—they incur an excessive computational burden encountered by processing *all the data points*. It should be noted, however, that points in the interior of the Voronoi space[12] of each class are usually processed for no reason—typically, they do not play any significant role in nearest-neighbour-like classification methods. Indeed, it is not unfair to state that processing the points in the "interior" of the Voronoi space becomes crucial only for "smaller" instantiations of the problem.

To overcome this disadvantage, a recursive PRS mechanism was proposed in Ref. [22]. In Ref. [22], the data set is sub-divided recursively into smaller subsets to filter out the "useless" internal points. Subsequently, a conventional PRS processes the smaller subsets of data points that effectively sample the entire space to yield *subsets* of prototypes—one set of prototypes for each subset. The prototypes, which result from each subset, are then coalesced, and processed again by the PRS to yield more refined prototypes. In this manner, prototypes which are in the interior of the Voronoi boundaries, and which are thus ineffective in the classification, are eliminated at the subsequent invocations of the PRS. A direct consequence of eliminating the "redundant" samples in the PRS computations is that the processing time of the PRS is *significantly* reduced.

Changing now the emphasis, we observe that with regard to designing classifiers, PRS can be employed as a pre-processing module to reduce the data set into a smaller representative subset and has thus been reported to optimize the design of KNS classifiers in Refs. [23,24]. The details of these are omitted here as they are irrelevant.

4. Proposed optimization of DBCs

We have already seen (in Section 2) that the two fundamental avenues by which DBCs can be optimized involve those of reducing the size of T,[13] and determining a suitable dissimilarity measure. The drawbacks which the reported methods have are the following:

1. The reported methods reduce the set T (to design Y) by merely *selecting* elements from the former.
2. When the reported methods compute the dissimilarity measure between two vectors, they ignore the second-order properties of the data.

With regard to reducing the size of the representative points, rather than deciding to discard or retain the training points, we permit the user the choice of either *selecting* some of the training samples using methods such as the CNN, or *creating* a smaller set of samples using the methods such as those advocated in the PNN, VQ, and HYB. This reduced set effectively serves as a new "representative" set for the dissimilarity representation. Additionally, we also permit the user to migrate the resultant set by an LVQ3-type method to further enhance the quality of the reduced samples. To investigate the computational advantage gained by resorting to such a PRS pre-processing phase, we observe, first of all, that the number of the reduced prototypes is *fractional* compared to that of the conventional ones, namely those obtained by random selection or clustering-based operations. Once the reduced prototypes are obtained, the dissimilarity matrix computation is significantly smaller since the computation is now done for a much smaller set, i.e., for an $n \times m$ matrix versus an $n \times n$ one.

[12] Typically, the Voronoi hyperplane between two classes is an equi-bisector of the space, partitioning the points of each class on either side. Classification is achieved by assigning a class index to a sample being tested, and in our context, this is done by computing the location of the tested sample in the Voronoi space, for example, by determining the class of *its* nearest neighbour using any well-established metric.

[13] In general, increasing the cardinality of the representative subset, drastically improves the average classification accuracy of the resultant DBC.

We propose to enhance DBCs by modifying each of the above steps as follows:

1. Reduce the set T (to design Y) by invoking a PRS on the former. The advantages of doing this (over the methods given in Section 2.2) are
   (a) A PRS permits us to obtain Y by either *selecting* the representative set or *creating* it.
   (b) By choosing an appropriate PRS, we are able to obtain the representative samples of each class Y_i by also including the information in the other Y_j's ($j \neq i$). This is especially true of the classes of PRSs which also invoke $LVQ3$-type perturbations on the representative points.
   (c) Most PRSs are designed with the specific task of determining the representative subset of points so as to maximize the class-discriminating properties. But incorporating such information in the subset used for DBCs, we believe that the resultant optimized DBC will be superior to the corresponding DBC which excludes this information. This is, indeed, our experience.
2. Compute the dissimilarity measure between two vectors using the Mahalanobis distance, where the estimated covariance matrix of each class is obtained by using the training samples of that class in T. The advantages of doing this (over the methods given in Section 2.3) are
   (a) Defining a well-discriminating dissimilarity measure for a non-trivial learning problem has always been known to be difficult. Indeed, designing such a measure in a DBC is equivalent to defining good *features* in a traditional feature-based classification problem. If a good measure is found and the training set T is representative, the authors of Ref. [8] report that the performance of the DBC can be enhanced.
   (b) The dissimilarity measures used in Ref. [8] (referred to in Section 2.3) do not utilize the actual spread of the data in the feature space. It is well known in the field that computing distances and achieving classification using the available covariance information can lead to results superior to those obtained by ignoring this information. We intend to take advantage of this.
   (c) Although the reduced set of points Y is used in the DBC, the information concerning the spread of the original data points is contained in the sets $\{T_i\}$. Thus, we believe that we can take advantage of this information by using the DBC obtained by the subset of points in $\{Y_i\}$, but by simultaneously incorporating the variance information contained in the original sets, the T_i's.
   (d) The basic premise for the DBC methodology is that since it does not operate on the class-conditional distributions, the accuracy can exceed the Bayes' errors bound. However, in attempting to achieve this, the state-of-the-art DBC methods ignore the information contained in the class-conditional distributions. Since this information can be summarized in the moments, we intend to use the second-order moments to optimize the design of DBCs. This is done, in our present scheme, by incorporating it in the computation of the Mahalanobis distances.
   (e) Recently, Horikawa [7] experimented on the properties of NN classifiers for high-dimensional patterns in DBCs. From the experimental results reported, the author demonstrated that the performance of the NN classifiers constructed with DBCs increases with the dimensionality of the pattern when the categorical pattern distributions are different from each other. This is, indeed, what we want to take advantage of incorporating *this* distinct information via the Mahalanobis distance computations.

Based on the above, our proposed strategy applicable for optimizing DBCs involves the following steps:

1. From each T_i compute the estimate of Σ_i, the covariance matrix of the class-conditional density.
2. Select the representative set Y from the training set T by resorting to one of the PRS methods given in Section 3.
3. Compute the dissimilarity matrix $D_{T,Y}[\cdot,\cdot]$, using Eq. (2), in which each individual dissimilarity is computed as the Mahalanobis distance evaluated using the value for Σ_i estimated in Step 2 above.
4. For a testing sample z, compute a dissimilarity column vector, $\delta_Y(z)$, by using the Mahalanobis distance used in Step 3 above.
5. Achieve the classification based on invoking a classifier built in the dissimilarity space and operating on *this* dissimilarity vector, $\delta_Y(z)$.

Consider now the question of determining which covariance matrix has to be used when we compute the dissimilarity between prototypes A and B. The rule we have used is simple: The covariance matrix used is always the covariance matrix associated with the *first* sample. Observe now that if both A and B are associated with the same class, the dissimilarity is symmetric. However, if they belong to different classes, the dissimilarity between A and B is computed using the covariance matrix associated with A, while the dissimilarity between B and A is computed using the covariance matrix associated with B—rendering the dissimilarity asymmetric. The DBC works well in spite of this, as we shall see.

5. Experimental results: artificial and real-life data sets

5.1. Experimental data

The proposed method has been tested and compared with the conventional ones. This was done by performing experiments on a number of data sets as summarized in Table 1. The sample vectors of each data set are divided into two subsets of equal size, and used for training and validation, alternately. The training set was used for computing the prototypes and the respective covariance matrices, and the test set was used for evaluating the quality of the corresponding classifier.

Table 1
The artificial and real-life data sets used in the experiments

Type	Datasets	No. of samples	No. of features	No. of classes
Artificial data	Random	400 (200; 200)	2	2
	Non_normal2	1000 (500; 500)	8	2
	Non_linear2	1000 (500; 500)	2	2
Real-life data	Iris2	100 (50; 50)	4	2
	Ionosphere	351 (176; 175)	34	2
	Sonar	208 (104; 104)	60	2
	Arrhythmia	452 (226; 226)	279	16
	Adult4	8336 (4168; 4168)	14	2

The sample vectors of each data set are divided into two subsets of equal size, and used for training and validation, alternately.

The data set described as "Random" is generated randomly with a uniform distribution, but with irregular decision boundaries. In this case, the points are generated uniformly, and the assignment of the points to the respective classes is achieved by *artificially* assigning them to the region they fall into, as per the manually created "irregular decision boundary".

The data set named "Non_normal2", which has also been employed as a benchmark experimental data set [1,16] for numerous experimental set-ups (including those involving the Bootstrap method) was generated from a mixture of four eight-dimensional Gaussian distributions as follows:

$$p_1(x) = \tfrac{1}{2}N(\mu_{11}, I_8) + \tfrac{1}{2}N(\mu_{12}, I_8),$$
$$p_2(x) = \tfrac{1}{2}N(\mu_{21}, I_8) + \tfrac{1}{2}(\mu_{22}, I_8),$$

where $\mu_{11} = [0, 0, \ldots, 0]$, $\mu_{12} = [6.58, 0, \ldots, 0]$, $\mu_{21} = [3.29, 0, \ldots, 0]$, and $\mu_{22} = [9.87, 0, \ldots, 0]$. In these expressions, I_8 is the eight-dimensional *Identity* matrix.

The data set named "Non_linear2", which has a strong non-linearity at its boundary, was generated artificially from a mixture of four variables as follows:

$$p_1(x) = \{x_1, \tfrac{1}{2}x_1^2 + y_1\},$$
$$p_2(x) = \{x_2, -\tfrac{1}{2}x_2^2 + y_2\},$$

where x_1, x_2, y_1, and y_2 are normal random variables whose means and variances are $(0, 10)$, $(10, 5)$, $(3, 10)$, and $(20, 5)$, respectively. The total number of vectors per class is 500.

On the other hand, the data sets "Iris2", "Ionosphere", "Sonar", "Arrhythmia", and "Adult4", which are *Real-life* benchmark data sets, are cited from the UCI Machine Learning Repository [25]. The "Iris2" data set contains 100 vectors. Each sample vector, of two classes, has four attributes which are all continuous numerical values. Originally, the "Iris" data set consists of three classes: Setosa, Versicolor, and Virginica. However, since the subset of Setosa samples is completely separated from the others, it is not difficult to classify it from the other two. Therefore, we have opted to employ a modified set "Iris2", which consisted only of the two classes, Versicolor and Virginica.

The "Ionosphere" data set contains 351 vectors. Each sample vector, of two classes, has 34 attributes which are all continuous numerical values. The "Sonar" data set contains 208 vectors. Each sample vector, of two classes, has 60 attributes which are all continuous numerical values. The "Arrhythmia" data set contains 279 attributes, 206 of which are real-valued and the rest are nominal. In our experiments, the nominal features were replaced by zeros. The aim of the PR exercise was to distinguish between the presence or absence of cardiac arrhythmia, and to classify the feature into one of the 16 groups. In our case, in the interest of uniformity, we merely attempted to classify the total instances into two categories, namely, "normal" and "abnormal".

The "Adult4" data set was extracted from a census bureau database.[14] Each sample vector has 14 attributes. Some of the attributes, such as the age, hours-per-week, etc., are continuous numerical values. The others, such as education, race, etc., are nominal symbols. In this case, the total number of sample vectors is 33,330. Among them, we randomly selected 8336 samples due to the time considerations, which is approximately 25% of the whole set.

In the above data sets, all of the vectors were normalized to be within the range $[-1, 1]$ using their standard deviations, and the data set for class j was randomly split into two subsets, $T_{j,t}$ and $T_{j,V}$, of equal size. One of them was used for choosing the initial prototypes and training the classifiers, and the other one was used in their validation (or testing). Later, the role of these sets was interchanged.

5.2. Experimental parameters

As in all learning algorithms, choosing the parameters[15] of the PRS and the conventional prototype selection schemes play an important role in determining the quality of the solution. The parameters for the reported conventional schemes such as the RAND, RAND_C, and KCentres (the methods referred to as *Random*, *RandomC* and *KCentres* in Section 2.2, respectively) and the PRS-based schemes such as the CNN, PNN, and HYB, are summarized as follows:

1. Parameters for the RAND, RAND_C, and KCentres
   (a) In RAND, a total of 10% of the samples were randomly selected from the original training data set.

[14] http://www.census.gov/ftp/pub/DES/www/welcome.html

[15] The same parameters were used for both the artificial data sets and for the real-life data sets.

(b) In RAND_C, for each class, 10% of the samples were randomly selected as prototypes.
(c) In KCentres, initially, 10% of the samples (for each class) were arbitrarily chosen as the initial cluster centres, after which a k-means clustering algorithm was invoked, as explained earlier.

2. Parameters for the CNN and the PNN
   (a) None.
3. Parameters for the HYB
   (a) The initial code book size was determined by the SVM. In this experiment, we hybridized the SVM and an LVQ3-type algorithm.
   (b) The parameters for the LVQ3 learning, such as the α, the ε, the window length, w, and the iteration length, η, were specified as described in Ref. [20].

5.3. Selecting prototype vectors

In order to evaluate the proposed classification mechanisms, we first selected the prototype vectors from the experimental data sets using the CNN, the PNN, and the HYB algorithms. In the HYB, we selected initial prototypes using a SVM algorithm. After this selection, we invoked a phase in which the optimal positions (i.e., with regard to classification) were learned with an LVQ3-type scheme. For the SVM and LVQ3 programs, we utilized publicly available software packages.[16]

Table 2 shows a comparison of the number of prototype vectors extracted from the artificial and real-life data sets, namely, "Random", "Non_normal2", "Non_linear2", "Iris2", "Ionosphere", "Sonar", "Arrhythmia", and "Adult4", respectively, using the CNN, PNN and HYB methods. The two values for each data set are the number of prototype vectors obtained from the training and test subsets, respectively.

From Table 2, for example, we see that the number of selected prototype vectors of the "Random" data set, $(m1, m2)$, are $(36, 30)$, $(30, 25)$ and $(18, 15)$, respectively. Each of them is considerably smaller than the size of the original data set. Using the selected vectors as a representative of the training data set, we can significantly reduce the dimensionality of the data set (and the consequential computations) without degrading the performance. Observe that once the reduced prototype set $Y = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m\}$ is obtained, the dimensionality of the matrix $D_{T,Y}[\cdot, \cdot]$ can be reduced into $n \times m$, where the dimensionality of the column vector is m, not n (for example, 15 instead of 200). The reduction of the classification processing time follows as a natural consequence. As an observation, we also mention that the reduction rate increased dramatically as the size of the data sets was increased.

5.4. Experimental results

We report below the run-time characteristics of the proposed algorithm for the artificial and real-life data sets, namely, "Random", "Non_normal2", "Non_linear2", "Iris2", "Ionosphere", "Sonar", "Arrhythmia", and "Adult4" as shown in Tables 3–5 . In these tables, the 'Whole Dataset' approach represents the experimental results for the entire original data sets, that is, $D_{T,T}[\cdot, \cdot]$, without employing any selection method. On the other hand, the results of the RAND, RAND_C, and KCentres, and the CNN, PNN, and HYB are obtained by calculating the dissimilarity matrix, $D_{T,Y}[\cdot, \cdot]$, using the representatives obtained with each respective method. Also, for each experimental data set, the first two lines for each set are the results for the training and test subsets, respectively, and the third line (bold faced) is the average of the above two lines.

Table 3 shows the DBC accuracy rates (%) of the classifiers designed with the conventional prototype selection schemes, such as the RAND, RAND_C, and KCentres methods, on the artificial and real-life data sets, in which the dissimilarity measure used is the Euclidean distance. It also presents the optimized DBC accuracy rates (%) on the same data sets, in which the prototype selection schemes are the PRS-based ones such as the CNN, PNN, and HYB methods, which, as explained earlier, use the Mahalanobis distance. An analysis of the results follows.

5.4.1. A comparison of conventional DBCs and PRS-based schemes

First of all, it is worth mentioning that the data set "Adult4" possesses a noticeable class imbalance.[17] Thus, although the numbers of prototypes obtained using the HYB scheme are 430 and 448 for the training and test sets, respectively, the number of prototypes in the second class is only about 10% of the number of prototypes in the first. Since this precludes a meaningful comparison, we shall omit it in further discussions. Suffice it to mention that the accuracy of the conventional DBC (71.98%) is quite comparable to the accuracy when the 'Whole Dataset' (72.08%) is utilized.

The overall remark that we can make from Table 3 is that, for every data set, the accuracy of the conventionally optimized DBC is quite comparable to (and sometimes even more accurate than) the accuracy when the 'Whole Dataset' is utilized. This is true for both the artificial and real-life data sets. The reason for this increased accuracy is that when a k-NN scheme is used in the dissimilarity space, the effects of the outliers become much more prominent when the 'Whole Dataset' is utilized. Thus, for the set "Iris2", the accuracy for the DBC using the 'Whole Dataset' is 77%, and this increases to 83% if the KCentres method is used to select the prototypes.

With regard to comparing the conventional and the new schemes, we again refer the reader to Table 3. Generally speaking, we note that if we consider the *average* accuracy obtained by using any of the conventional prototypes selection schemes (namely, RAND, RAND_C, KCentres) and compare them with the *average* accuracy obtained by using any of the PRSs (namely, CNN, PNN, HYB), the latter is almost always superior. Thus, to render the comparison more fair, in what fol-

[16] These packages can be available from: http://www-ai.cs.uni-dortmund.de/SOFTWARE/SVM_LIGHT/svm_light.eng.html and http://cochlea.hut.fi/research/som_lvq_pak.shtml, respectively.

[17] For example, the sample points of two classes ω_1 and ω_2 of its training set are 3961 and 206, respectively.

Table 2
The number of prototype vectors extracted from experimental data sets using the CNN, PNN, and HYB methods

Data set types	Data set names	Whole dataset ($n1, n2$)	Selected prototypes ($m1, m2$)		
			CNN	PNN	HYB
Artificial data	Random	200, 200	36, 30	30, 25	18, 15
	Non_normal2	500, 500	64, 66	56, 380	63, 57
	Non_linear2	500, 500	96, 109	87, 90	82, 58
Real-life data	Iris2	50, 50	15, 12	10, 7	6, 8
	Ionosphere	176, 176	51, 42	37, 33	44, 46
	Sonar	104, 104	52, 53	34, 33	53, 59
	Arrhythmia	226, 226	32, 28	8, 7	65, 69
	Adult4	4168, 4168	755, 752	659, 658	430, 448

The two values for each data set are the number of prototype vectors obtained from the training and test subsets, respectively.

Table 3
The accuracy rates (%) of the DBCs for the artificial and real-life data sets

Data set types	Data set names	Conventional schemes				Proposed schemes		
		Wholeset	RAND	RAND_C	KCentres	CNN	PNN	HYB
Artificial	Random	86.00	86.05	82.40	86.00	85.50	87.00	85.50
		81.50	81.90	82.10	81.00	82.50	82.00	83.00
		83.75	**83.98**	**82.25**	**83.50**	**84.00**	**84.50**	**84.25**
	Non_normal2	95.60	95.44	95.54	96.00	85.40	88.20	85.40
		94.60	94.74	94.56	94.80	93.60	93.80	94.80
		95.10	**95.09**	**95.05**	**95.40**	**89.50**	**91.00**	**90.10**
	Non_linear2	57.80	58.08	58.62	59.00	71.80	74.60	74.40
		61.20	61.52	61.84	61.00	78.00	78.20	78.20
		59.50	**59.80**	**60.23**	**60.00**	**74.90**	**76.40**	**76.30**
Real-life	Iris2	78.00	78.00	78.00	88.00	84.00	82.00	82.00
		76.00	75.80	64.00	78.00	96.00	94.00	94.00
		77.00	**76.90**	**71.00**	**83.00**	**90.00**	**88.00**	**88.00**
	Ionosphere	72.73	73.07	70.80	70.45	89.77	89.77	89.77
		69.89	70.34	68.98	68.18	82.39	82.39	82.39
		71.31	**71.70**	**69.89**	**69.32**	**86.08**	**86.08**	**86.08**
	Sonar	64.42	61.15	52.21	52.88	65.38	64.42	64.42
		55.77	56.06	60.00	57.69	75.00	74.04	75.00
		60.10	**58.61**	**56.11**	**55.29**	**70.19**	**69.23**	**69.71**
	Arrhythmia	93.81	86.99	86.42	86.73	91.15	90.71	95.13
		92.04	87.48	84.38	77.88	94.69	95.13	95.13
		92.92	**87.23**	**85.40**	**82.31**	**92.92**	**92.92**	**95.13**
	Adult4	72.81	—	72.52	68.78	—	—	—
		71.35	—	71.43	74.37	—	—	—
		72.08	—	**71.98**	**71.57**	—	—	—

The results reported concern the schemes designedd with the conventional prototype selection methods such as the RAND, RAND_C, and KCentres methods which use the Euclidean distance, and the optimized ones which use PRS-based schemes such as the CNN, PNN, and HYB methods and the Mahalanobis distance. The other notation is discussed in the text.

lows, we shall consider the *best* accuracy that a conventional scheme yields, and compare it with the *best* accuracy that a PRS yields.

Consider the artificial data set, "Non_linear2". In this case, the RAND_C method yields the best accuracy (60.23%) of the three conventional selection methods when the Euclidean distance is used. The corresponding accuracy for the optimized methods is obtained when the PNN is the PRS used, and it yields an accuracy of 76.40%. Similarly, consider the real-life "'Arrhythmia" data set. In this case, the RAND method yields the best accuracy (87.23%) of the three conventional selection methods when the Euclidean distance is used. The corresponding accuracy for the optimized methods is obtained when the HYB is the PRS used, leading to an accuracy of 95.13%. The conclusion that we can make is that the optimized methods (the PRSs used in conjunction with the Mahalanobis distance) are almost always superior (and sometimes, much more superior) to the conventional schemes.

It is also interesting to note how a conventional selection scheme (such as the RAND, RAND_C, KCentres) would perform if the Mahalanobis distance is used. This is tabulated in Table 4. Here too, if the average accuracy of the schemes

Table 4
The accuracy rates (%) of the DBCs designed with the conventional prototype selection schemes such as the RAND, RAND_C, and KCentres methods, and which use the Mahalanobis distance

Data set types	Data set names	Wholeset	RAND	RAND_C	KCentres
Artificial	Random	87.00	85.50	86.50	86.00
		82.50	82.45	83.45	83.00
		84.75	**83.98**	**84.98**	**84.50**
	Non_normal2	96.00	95.64	95.72	91.60
		94.80	94.78	94.26	94.60
		95.40	**95.21**	**94.99**	**93.10**
	Non_linear2	71.20	70.68	70.38	71.40
		75.00	74.88	69.92	66.00
		73.10	**72.78**	**70.15**	**68.70**
Real-life	Iris2	82.00	83.20	90.00	80.00
		94.00	95.20	92.00	96.00
		88.00	**89.20**	**91.00**	**88.00**
	Ionosphere	89.77	89.77	89.77	89.77
		82.39	82.39	82.16	81.82
		86.08	**86.08**	**85.97**	**85.80**
	Sonar	66.35	62.50	63.08	63.46
		72.12	72.02	71.73	71.15
		69.23	**67.26**	**67.40**	**67.31**
	Arrhythmia	90.71	88.50	90.84	91.15
		91.59	89.65	92.21	91.15
		91.15	**89.07**	**91.53**	**91.15**
	Adult4	40.65	—	40.59	39.26
		13.32	—	13.32	13.32
		26.99	—	**26.95**	**26.29**

The other notation is discussed in the text.

Table 5
The CPU-processing times (seconds) of the classifiers designed with the conventional prototype selection schemes such as the RAND, RAND_C, and KCentres methods, and the PRS-based ones such as the CNN, PNN, and HYB methods, on the artificial and real-life data sets

Data set names	Distance meas.	Conventional schemes				PRS-based schemes		
		Wholeset	RAND	RAND_C	KCentres	CNN	PNN	HYB
Random	ED	1.19	0.10	0.09	0.10	0.24	0.13	0.11
	MD	3.78	0.36	0.34	0.73	0.69	0.61	0.41
Non_normal2	ED	3.87	0.43	0.43	0.47	1.02	2.17	0.88
	MD	18.94	1.94	1.94	2.53	3.41	9.78	3.02
Non_linear2	ED	3.80	0.42	0.44	0.64	1.04	0.80	0.57
	MD	18.48	1.95	1.92	2.46	4.91	3.77	2.48
Iris2	ED	0.05	0.01	0.01	0.01	0.02	0.02	0.02
	MD	0.27	0.03	0.03	0.06	0.09	0.07	0.06
Ionosphere	ED	0.77	0.10	0.09	0.13	0.25	0.22	0.26
	MD	3.51	0.39	0.34	0.46	1.08	0.81	1.03
Sonar	ED	0.35	0.04	0.05	0.05	0.19	0.16	0.24
	MD	1.53	0.20	0.20	0.21	0.92	1.00	1.07
Arrhythmia	ED	2.76	0.38	0.39	0.45	0.63	0.30	0.98
	MD	84.42	9.92	9.84	12.23	14.97	5.80	30.96
Adult4	ED	1496.00	—	35.78	39.56	63.97	58.55	42.20
	MD	3277.30	—	137.46	154.84	269.06	240.80	164.78

In each case the times taken are measured when the distance is Euclidean (given as ED in the table) and when the distance is the Mahalanobis (given as MD in the table). The other notations are given in the text.

(RAND, RAND_C, KCentres) is compared to the average accuracy of the PRSs (CNN, PNN and HYB), the latter is almost always the better option. The optimized methods continue to be the superior ones if the best of the method is chosen in each case, although the advantage is not so marked.

The general conclusion that we can make from the results is the following: it is always advantageous to use a PRS to select the subset of representative points, but when a PRS is used, *one must not resort to using the Euclidean distance to compute the dissimilarities*. Rather, since the PRS implicitly

takes the data distribution into consideration, it must be used in conjunction with a distribution-based dissimilarity measure, like the Mahalanobis distance.

From the above considerations, it is also worth mentioning that it is not so easy to crown any one scheme to be superior to the others in the context of the PRS method used. But a general observation seems to be that for artificial data the PNN is the most advantageous, and the HYB seems to yield the best results for the real-life data sets.

5.4.2. A comparison of processing-related CPU-times

Table 5 shows the processing CPU-times (in seconds) of the two DBC mechanisms, namely, the conventional DBCs and the optimized PRS-based ones, for the artificial and real-life data sets. In each case the times taken are measured[18] when the distance is Euclidean (given as ED in the table) and when the distance is the Mahalanobis (given as MD in the table). The setting of the experiments is as discussed in the previous subsections.

From Table 5, first of all, we see that the processing CPU-times for the artificial and real-life data sets can be reduced significantly by employing a PRS such as the CNN, PNN, and HYB. Indeed, this is achieved without sacrificing the accuracy so much.

Consider the 'Whole Dataset' approach for the "Non_linear2" data set. If the entire set of size 500 was processed, the classification times taken for the ED and MD are 3.80, and 18.48 s, respectively. However, if the same set was first preprocessed by the CNN, to yield the optimized DBC, the processing times are 1.04, and 4.91 s, respectively. Notice that the classification accuracies of the method are almost the same, as shown in Tables 3 and 4. The CPU-times of the Mahalanobis-based DBCs are marginally longer than those of the Euclidean-based one.

It should also be mentioned that the reduction of processing time increased dramatically when the size of the data sets was increased. Consider the case of the "Adult4" data set. If the entire set of size 4168 was processed, the classification times taken for the ED and MD are 1496.00 and 3277.30 s, respectively. However, if the same set was first pre-processed by a PRS such as the CNN, the processing times are reduced to 63.97 and 269.06 s, respectively. Identical comments, namely that the required time is fractional, can also be made about the PNN and HYB schemes, and for all the artificial and real-life data sets.

As in the case of the accuracy, it is again not so easy to crown any one method to be superior to the others in the optimized DBCs. This re-iterates the basic proposition found in the literature, that the suitability of any prototype selection method depends on the properties of the data set itself [8,9,20]. However, the reader should easily observe that the computational advantage gained is *significant*, and the accuracy lost is *marginal*. But as a matter of comparison, it is clear that with regard to the classification accuracies and the times involved, the proposed PRS-based methods combined with the Mahalanobis distance are noticeably superior to the conventional schemes—*even though the resulting dissimilarity criterion is not symmetric*.

[18] In the case of the KCentres, CNN, PNN, and HYB, the entries do not include the time required to extract the prototypes, because these were pre-computed and stored. In the case of the PRSs, these extraction times are fractional (except for the case of the "Adult4" data set) when compared to the processing required for the 'Whole Data set', and given in Ref. [21]. However, this does not affect the fundamental premise of our conclusions.

6. Conclusions

In this paper we have considered the problem of efficiently implementing dissimilarity-based classifiers (DBCs). This methodology, proposed by Duin and his co-authors (see Refs. [3–6,8]), is a way of defining classifiers between the classes, and is not based on the feature measurements of the individual patterns, but rather on a suitable dissimilarity measure between them. The advantage of DBCs is that since they do not operate on the class-conditional distributions, the accuracy can exceed the error Bayes' bound. DBCs, in their virgin form have a fundamental problem, which is the need to compute, store and process the inter-pattern dissimilarities for all the training samples. In this paper, we have suggested a novel strategy to enhance the computation for all families of DBCs. Rather than compute, store and process the DBC based on the entire data set, we advocate that the training set be first reduced into a smaller representative subset obtained by invoking a prototype reduction scheme (PRS), whose output yields the points to be utilized by the DBC. Apart from utilizing PRSs, in the paper we have also proposed simultaneously employing the Mahalanobis distance as the dissimilarity-measurement criterion to increase the DBCs classification accuracy. Our experimental results demonstrate that the proposed mechanism increases the classification accuracy when compared with the "conventional" approaches for samples involving real-life as well as artificial data sets—*even though the resulting dissimilarity criterion is not symmetric*.

Although we have given a motivation for using the PRS to optimize dissimilarity-based classification, the problem of theoretically analyzing our methodology remains open. We believe that this will be nontrivial. Besides this, in this present paper, we have also investigated the importance of two criterias: The prototype selection strategy and the dissimilarity measure to be used. However, the performance of the DBC could be further improved by studying how the appropriate classifiers in the dissimilarity space themselves can be optimized. The research is currently being undertaken.

Acknowledgement

The authors are grateful to Bob Duin for his friendly and cooperative manner. The first author thanks him and his team for the time spent in Holland in October 2005, and the second author is grateful for the time spent together in Poland in May 2005.

References

[1] K. Fukunaga, Introduction to Statistical Pattern Recognition, second ed., Academic Press, San Diego, 1990.

[2] A.K. Jain, R.P.W. Duin, J. Mao, Statistical pattern recognition: a review, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-22 (1) (2000) 4–37.

[3] R.P.W. Duin, D. Ridder, D.M.J. Tax, Experiments with a featureless approach to pattern recognition, Pattern Recognition Lett. 18 (1997) 1159–1166.

[4] R.P.W. Duin, E. Pekalska, D. de Ridder, Relational discriminant analysis, Pattern Recognition Lett. 20 (1999) 1175–1181.

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

[6] E. Pekalska, Dissimilarity representations in pattern recognition, Concepts, theory and applications, Ph.D. Thesis, Delft University of Technology, Delft, The Netherlands, 2005.

[7] Y. Horikawa, On properties of nearest neighbour classifiers for high-dimensional patterns in dissimilarity-based classification, IEICE Trans. Inf. Syst. J88-D-II (4) (2005) 813–817 (in Japanese).

[8] E. Pekalska, R.P.W. Duin, P. Paclik, Prototype selection for dissimilarity-based classifiers, Pattern Recognition 39 (2006) 189–208.

[9] J.C. Bezdek, L.I. Kuncheva, Nearest prototype classifier designs: an experimental study, International J. Intell. Syst. 16 (12) (2001) 1445–11473.

[10] B.V. Dasarathy, Nearest Neighbor NN Norms: NN Pattern Classification Techniques, IEEE Computer Society Press, Los Alamitos, 1991.

[11] P.E. Hart, The condensed nearest neighbour rule, IEEE Trans. Inf. Theory IT-14 (1968) 515–516.

[12] G.W. Gates, The reduced nearest neighbour rule, IEEE Trans. Inf. Theory IT-18 (1972) 431–433.

[13] K. Fukunaga, J.M. Mantock, Nonparametric data reduction, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-6 (1) (1984) 115–118.

[14] C.L. Chang, Finding prototypes for nearest neighbour classifiers, IEEE Trans. Comput. C-23 (11) (1974) 1179–1184.

[15] J.C. Bezdek, T.R. Reichherzer, G.S. Lim, Y. Attikiouzel, Multiple-prototype classifier design, IEEE Trans. Syst. Man Cybern.—Part C SMC-28 (1) (1998) 67–79.

[16] Q. Xie, C.A. Laszlo, R.K. Ward, Vector quantization techniques for nonparametric classifier design, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-15 (12) (1993) 1326–1330.

[17] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, Data Min. Knowl. Discovery 2 (2) (1998) 121–167.

[18] V.N. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.

[19] T. Kohonen, Self-organizing Maps, Springer, Berlin, 1995.

[20] S.-W. Kim, B.J. Oommen, Enhancing prototype reduction schemes with LVQ3-type algorithms, Pattern Recognition 36 (5) (2003) 1083–1093.

[21] S.-W. Kim, B.J. Oommen, A brief taxonomy and ranking of creative prototype reduction schemes, Pattern Anal. Appl. J. 6 (3) (2003) 232–244.

[22] S.-W. Kim, B.J. Oommen, Enhancing prototype reduction schemes with recursion: a method applicable for "large" data sets, IEEE Trans. Syst. Man Cybern. Part B SMC-34 (3) (2004) 1384–1397.

[23] S.-W. Kim, B.J. Oommen, On using prototype reduction schemes to optimize kernel-based nonlinear subspace methods, Pattern Recognition 37 (2) (2004) 227–239.

[24] S.-W. Kim, B.J. Oommen, On using prototype reduction schemes and classifier fusion strategies to optimize kernel-based nonlinear subspace methods, IEEE Trans. Pattern Anal. Mach. Intell. 27 (3) (2005) 455–460.

[25] D.J. Newman, S. Hettich, C.L. Blake, C.J. Merz, UCI Repository of Machine Learning Databases, Department of Information and Computer Sciences, University of California, Irvine, 1998. ⟨http://www.ics.uci.edu/~mlearn/MLRepository.html⟩.

About the Author—B. JOHN OOMMEN was born in Coonoor, India on September 9, 1953. He obtained his B.Tech. degree from the Indian Institute of Technology, Madras, India in 1975. He obtained his M.E. from the Indian Institute of Science in Bangalore, India in 1977. He then went on for his M.S. and Ph. D. which he obtained from Purdue University, in West Lafayettte, Indiana in 1979 and 1982, respectively. He joined the School of Computer Science at Carleton University in Ottawa, Canada, in the 1981–1982 academic year. He is still at Carleton and holds the rank of a Full Professor. Since July 2006, he has been awarded the honorary rank of *Chancellor's Professor*, which is a lifetime award from Carleton University. His research interests include Automata Learning, Adaptive Data Structures, Statistical and Syntactic Pattern Recognition, Stochastic Algorithms and Partitioning Algorithms. He is the author of more than 265 refereed journal and conference publications, and is a *Fellow of the IEEE* and a *Fellow of the IAPR*. Dr. Oommen is on the Editorial Board of the *IEEE Transactions on Systems, Man and Cybernetics*, and *Pattern Recognition*.

About the Author—SANG-WOON KIM received the B.E. degree from Hankook Aviation University, Korea in 1978, and the M.E. and the Ph.D. degrees from Yonsei University, Seoul, Korea in 1980 and 1988, respectively, both in Electronic Engineering. In 1989, he joined the Department of Computer Science and Engineering, Myongji University, Korea and is currently a Full Professor there. His research interests include Statistical Pattern Recognition, Machine Learning, and Avatar Communications in Virtual Worlds. He is the author or coauthor of 26 regular papers and 12 books. He is a *Senior Member of the IEEE* and a Member of the *IEICE* and the *IEEK*.