On Optimizing Dissimilarity-Based Classification Using Prototype Reduction Schemes[⋆]

Sang-Woon Kim[1,⋆⋆] and B. John Oommen[2,⋆⋆⋆]

[1] Dept. of Computer Science and Engineering,
Myongji University, Yongin, 449-728 Korea
kimsw@mju.ac.kr

[2] School of Computer Science, Carleton University,
Ottawa, Canada : K1S 5B6
oommen@scs.carleton.ca

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[1] and his co-authors (see [3], [4], [5], [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 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. Apart from utilizing PRSs, in the paper we also propose simultaneously employing the Mahalanobis distance as the dissimilarity-measurement criterion to increase the DBC's 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.

⋆ 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).

⋆⋆ *Senior Member IEEE.*

⋆⋆⋆ *Fellow of the IEEE.*

[1] 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.

1 Introduction

The field of statistical Pattern Recognition[1], [2] has matured since its infancy in the 1950's, and the aspiration to 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 peculiar 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 [3], [4], [5], [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[2]. 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 numbers 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, $T = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n\}$, is based on pairwise comparisons and is expressed, for example, as an $n \times n$ dissimilarity matrix[3] $D_{T,T}[\cdot,\cdot]$, 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 i^{th} row of $D_{T,T}[\cdot,\cdot]$. The principle behind DBCs is that a new (testing) sample $\boldsymbol{z}$, if

[2] 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.

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

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 [4] discuss a number of methods such as random selections, the k-centers method, and others which will be catalogued 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[4].

1.1 Contributions of the Paper

We claim two modest contributions in this paper based on rigorous tests done on both established benchmark artificial and real-life data sets:

1. First of all, we show that a PRS[5] 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.*
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.

2 Dissimilarity-Based Classification and Prototype Reduction Schemes

2.1 Foundations of DBCs

Let $T = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n\} \in R^p$ be a set of n feature vectors in a p dimensional space. We assume that T is a labeled data set, so that T can be decomposed into, say, c subsets $\{T_1, \cdots, T_c\}$ such that $\forall i \neq j$:
(1) $T_i = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{n_i}\}$, (2) $n = \sum_{i=1}^{c} n_i$, (3) $T = \bigcup_{k=1}^{c} T_k$, and (4) $T_i \cap T_j = \phi$.

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

[4] 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.

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

To achieve this, we assume that from T_i, the training data of class ω_i, we extract a prototype set[6], Y_i, where,

(1)$Y_i = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_{m_i}\}$, and (2) $m = \sum_{i=1}^{c} m_i$.

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 $\boldsymbol{y}_j$. Since, d, is required to be nonnegative, reflexive and symmetric[7]:

$d(\boldsymbol{x}_i, \boldsymbol{y}_j) \geq 0$ with $d(\boldsymbol{x}_i, \boldsymbol{y}_j) = 0$ if $\boldsymbol{x}_i = \boldsymbol{y}_j$, 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 :

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

Here, we define the dissimilarity matrix $D_{T,Y}[\cdot,\cdot]$ to represent 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 i^{th} 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., $m << p$), 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], [7], [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 [8] has been performed from the perspective of the resultant error rates and the 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*, *Random_C*, *KCentres*, *ModeeSeek*, *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.

[6] 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, $T_j, j \neq i$.

[7] Note that $d(\cdot,\cdot)$ need not be a *metric* [8].

2. *RandomC* : This method involves a random selection of m_i samples per class, ω_i, from T_i.
3. *KCentres* : This method[8] 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 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 $Y_i = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_{m_i}\}$ consisting of m_i objects, e.g. randomly chosen from T_i.
   (b) For each $\boldsymbol{x} \in T_i$, find its nearest neighbor in Y_i. Let $N_j, j = 1, \cdots, m_i$, be a subset of T_i consisting of objects that yield the same nearest neighbor $\boldsymbol{y}_j$ in Y_i. This means that $T_i = \cup_{j=1}^{m_i} N_j$.
   (c) For each N_j, find its center $\boldsymbol{c}_j$, 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 center $\boldsymbol{c}_j$, 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 consisting of all the final sets Y_i.

From the experimental results of [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 (see [8]) 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.*

2.3 Dissimilarity Measures Used in DBCs

Fundamental to DBCs is the measure used to quantify the dissimilarity between two vectors[9]. The work in [8] reports extensive experiments conducted using various dissimilarity measures (see Table 2 of [8]). A list of these measures where we quantify the dissimilarity between $\boldsymbol{v}$ and $\boldsymbol{w} \in R^q$, is given below:

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

$$h(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\| \tag{2}$$

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

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

The measures which were tested in [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 pattern recognition 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 [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. (1), in which each individual dissimilarity is computed using one of the measures described in Section 2.
3. For a testing sample $\boldsymbol{z}$, compute a dissimilarity column vector, $\delta_Y(\boldsymbol{z})$, 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 $\delta_Y(\boldsymbol{z})$.

2.5 State-of-the-Art Prototype Reduction Schemes

In non-parametric pattern classification which use the Nearest Neighbour (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.To reduce the number of training vectors, various PRSs have been reported in the literature - two excellent surveys are found in [9], [10]. Rather than embark on yet another survey of the field, we mention here a *few* representative methods of the "zillions" that have been reported. One of the first of its kind is the Condensed Nearest Neighbour (CNN) rule [11]. 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, the Prototypes for Nearest Neighbour (PNN) classifiers [13], the Selective Nearest Neighbour (SNN) rule, two modifications of the CNN [17] Neighbour (ENN) rule and the non-parametric data reduction method [12]. Besides these, the Vector Quantization (VQ) and the Bootstrap [17] techniques and Support Vector Machines (SVM) [15] have also been reported as being extremely effective approaches to data reduction.

In selecting prototypes, vectors near the boundaries between the classes have to be considered to be more significant, and the created prototypes need to be adjusted towards the classification boundaries so as to yield a higher performance. Based on this philosophy, we recently proposed a new hybrid approach that involved two distinct phases, namely, selecting and adjusting [18], [19]. To overcome the computational burden for “large” datasets, we also proposed a recursive PRS mechanism in [20]. In [20], 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 are thus ineffective in the classification, are eliminated at the subsequent invocations of the PRS. As a result, 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 have thus been reported to optimize the design of KNS classifiers in [21], [22]. The details of these are omitted here as they are irrelevant.

3 Proposed Optimization of DBC’s

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[10], 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 preprocessing 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.

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

We propose to enhance DBC's by modifying each of the above 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 [8] report that the performance of the DBC can be enhanced.
   (b) The dissimilarity measures used in [8] (refereed 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 2.5.
3. Compute the dissimilarity matrix $D_{T,Y}[\cdot,\cdot]$, using Eq. (1), 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 $\boldsymbol{z}$, compute a dissimilarity column vector, $\delta_Y(\boldsymbol{z})$, 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, $\delta_Y(\boldsymbol{z})$.

4 Experimental Results : Artificial/Real-Life Data Sets

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. 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.

In our experiments, the three artificial data sets "Random", "Non_normal 2", and "Non_linear 2" were generated with different sizes of testing and training sets of cardinality 400, 1,000, and 1,000 respectively. 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] for numerous experimental set-ups was generated from a mixture of four 8-dimensional Gaussian distributions.

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: $p_1(x) = \{x_1, \frac{1}{2}x_1^2 + y_1\}, p_2(x) = \{x_2, -\frac{1}{2}x_2^2 + y_2\}$, where x_1, x_2, y_1, 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" (in short, "Iono"), "Sonar", "Arrhythmia" (in short, "Arrhy") and "Adult4", which are real benchmark data sets, are cited from the UCI Machine Learning Repository[11]. Their details can be found in the latter site, and also in [17].

In the above, 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 were interchanged.

Experimental Parameters: As in all algorithms, choosing the parameters[12] 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:

1. Parameters for the RAND, RAND_C, and KCentres : In RAND, a total of 10 % of the samples were randomly selected from the original training data set. In RAND_C, for each class, 10 % of the samples were randomly selected as prototypes. In KCentres, initially, 10 % of the samples (for each class) were arbitrarily chosen as the initial cluster centers, after which a k-means clustering algorithm was invoked, as explained earlier.

2. Parameters for the CNN and the PNN : None.

3. Parameters for the HYB : The initial code book size was determined by the SVM. In this experiment, we hybridized the SVM and an LVQ3-type algorithm. 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 [18].

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[13]. Table 1 shows a comparison of the number of prototype vectors extracted from the artificial and real-life data sets using the CNN, PNN, and HYB methods.

[11] http://www.ics.uci.edu/mlearn/MLRepository.html

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

[13] 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.

Table 1. The number of prototype vectors extracted from experimental data sets using the CNN, PNN, and HYB methods. The two values for each data set are the numbers of prototype vectors obtained from the training and test subsets, respectively.

Dataset Types	Dataset 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	$4168, 4168$	$755, 752$	$659, 658$	$430, 448$

From Table 1, for example, we see that the numbers of selected prototype vectors of the "Random" dataset, $(m1, m2)$, 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 dataset (and the consequential computations) without degrading the performance. Once the reduced prototype set $Y = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_m\}$, is obtained, the dimensionality (and the classification processing time) 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 (e.g., 15 not 200).

Experimental Results: We report below the run-time characteristics of the proposed algorithm for the artificial and real-life data sets as shown in Table 2, where the 'Wholeset' 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, each result is the averaged one for the training and the test sets, respectively.

Table 2 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.

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[14]. Thus, although the number of prototypes obtained using the HYB scheme are 430 and 448 for the training and test sets respectively, the number

[14] 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 accuracy rates (%) of the DBCs for the artificial and real-life data sets. The results reported concern the schemes designed 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.

Dataset Types	Dataset Names	Conventional Schemes				Proposed Schemes		
		Wholeset	RAND	RAND_C	KCentres	CNN	PNN	HYB
Artificial	Random	83.75	83.98	82.25	83.50	84.00	84.50	84.25
	Non_normal2	95.10	95.09	95.05	95.40	89.50	91.00	90.10
	Non_linear2	59.50	59.80	60.23	60.00	74.90	76.40	76.30
Real-life	Iris2	77.00	76.90	71.00	83.00	90.00	88.00	88.00
	Ionosphere	71.31	71.70	69.89	69.32	86.08	86.08	86.08
	Sonar	60.10	58.61	56.11	55.29	70.19	69.23	69.71
	Arrhythmia	92.92	87.23	85.40	82.31	92.92	92.92	95.13
	Adult4	72.08	-	71.98	71.57	-	-	-

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 for the 'Wholeset' (72.08%).

The overall remark that we can make from Table 2 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 'Wholeset' 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 'Wholeset' is utilized. Thus, for the set "Iris2", the accuracy for the DBC using the 'Wholeset' is 77%, and this increases to 83% if the KCentres method is used.

With regard to comparing the conventional and the new schemes, we again refer the reader to Table 2. 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 follows, we shall consider the *best* accuracy that a conventional scheme yields, and compare it with the *best* accuracy that a PRS would yield.

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. The details of the results are omitted here in the interest of compactness, but can be found in [17]. Here too, if the average accuracy of the schemes (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. From the other results given in [17], we also see that the processing CPU-times can also be reduced significantly by employing a PRS such as the CNN, PNN, and HYB without sacrificing the accuracy so much.

5 Conclusions

In this paper, we have suggested a novel strategy to enhance the computation for all families of Dissimilarity-Based Classifiers (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 DBC's 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.

References

1. K. Fukunaga, *Introduction to Statistical Pattern Recognition, Second Edition*, Academic Press, San Diego, 1990.
2. A. K. Jain, R. P. W. Duin and J. Mao, "Statistical pattern recognition: A review", *IEEE Trans. Pattern Anal. and Machine Intell.*, vol. PAMI-22, no. 1, pp. 4 - 37, Jan. 2000.

3. R. P. W. Duin, D. Ridder and D. M. J. Tax, “Experiments with a featureless approach to pattern recognition”, *Pattern Recognition Letters*, vol. 18, pp. 1159 - 1166, 1997.
4. R. P. W. Duin, E. Pekalska and D. de Ridder, “Relational discriminant analysis”, *Pattern Recognition Letters*, vol. 20, pp. 1175 - 1181, 1999.
5. E. Pekalska and R. P. W. Duin, “Dissimilarity representations allow for buillding good classifiers”, *Pattern Recognition Letters*, vol. 23, pp. 943 - 956, 2002.
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 neighbor classifiers for high-dimensional patterns in dissimilarity-based classification”, *IEICE Trans. Information & Systems*, vol. J88-D-II, no. 4, pp. 813 - 817, Apr. 2005, in Japanese.
8. E. Pekalska, R. P. W. Duin, and P. Paclik, “Prototype selection for dissimilarity-based classifiers”, *Pattern Recognition*, vol. 39, pp. 189 - 208, 2006.
9. J. C. Bezdek and L. I. Kuncheva, “Nearest prototype classifier designs: An experimental study”, *International Journal of Intelligent Systems*, vol. 16, no. 12, pp. 1445 - 11473, 2001.
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 neighbor rule”, *IEEE Trans. Inform. Theory*, vol. IT-14, pp. 515 - 516, May 1968.
12. K. Fukunaga and J. M. Mantock, “Nonparametric data reduction”, *IEEE Trans. Pattern Anal. and Machine Intell.*, vol. PAMI-6, no. 1, pp. 115 - 118, Jan. 1984.
13. C. L. Chang, “Finding prototypes for nearest neighbor classifiers”, *IEEE Trans. Computers*, vol. C-23, no. 11, pp. 1179 - 1184, Nov. 1974.
14. J. C. Bezdek, T. R. Reichherzer, G. S. Lim, and Y. Attikiouzel, “Multiple-prototype classifier design”, *IEEE Trans. Systems, Man, and Cybernetics - Part C*, vol. SMC-28, no. 1, pp. 67 - 79, Feb. 1998.
15. C. J. C. Burges, “A tutorial on support vector machines for pattern recognition”, *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121 - 167, 1998.
16. T. Kohonen, *Self-Oganizing Maps*, Berlin, Springer - Verlag, 1995.
17. S.-W. Kim and B. J. Oommen, “On Optimizing Dissimilarity-based Classification Using Prototype Reduction Schemes”. *Unabridged version of this paper.*
18. S. -W. Kim and B. J. Oommen, ”Enhancing prototype reduction schemes with LVQ3-type algorithms”, *Pattern Recognition*, vol. 36, no. 5, pp. 1083 - 1093, 2003.
19. S. -W. Kim and B. J. Oommen, “A Brief Taxonomy and Ranking of Creative Prototype Reduction Schemes”, *Pattern Analysis and Applications Journal*, vol. 6, no. 3, pp. 232 - 244, December 2003.
20. S. -W. Kim and B. J. Oommen, “Enhancing Prototype Reduction Schemes with Recursion : A Method Applicable for “Large” Data Sets”, *IEEE Trans. Systems, Man, and Cybernetics - Part B*, vol. SMC-34, no. 3, pp. 1384 - 1397, June 2004.
21. S. -W. Kim and B. J. Oommen, “On using prototype reduction schemes to optimize kernel-based nonlinear subspace methods”, *Pattern Recognition*, vol. 37, no. 2, pp. 227 - 239, 2004.
22. S. -W. Kim and B. J. Oommen, “On using prototype reduction schemes and classifier fusion strategies to optimize kernel-based nonlinear subspace methods”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 3, pp. 455 - 460, March 2005.