Computational Complexities of Dimensionality Reduction Schemes for Dissimilarity-Based Classification

Sang-Woon KIM† and Jian GAO†

† Dept. of Computer Engineering, Myongji University Yongin, 449–728 Korea
E-mail: †{kimsw,marsgao}@mju.ac.kr

Abstract Dissimilarity-Based Classifiers (DBCs) are a way of defining classifiers based on a suitable dissimilarity measure between individual patterns. In the attempt to find the most appropriate dimensionality reduction method for DBCs, in this paper, we report a comparison between the computational complexities of prototype selection methods (PSM) and dimensionality reduction schemes (DRS). This is done by theoretically and experimentally demonstrating the strength in terms of processing time and classification accuracy.

Key words Dissimilarity-Based Classifiers (DBCs), Prototype Selection Methods (PSM), Dimensionality Reduction Schemes (DRS)

1. Introduction

One of the most recent and novel developments in pattern classification is the concept of dissimilarity-based classifiers (DBCs) proposed by Duin and his co-authors [1], [2]. 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 [3].

The problem with this strategy, however, is that we need to select a representative set of data that is both compact and capable of representing the entire data set. In DBCs, a good selection of prototypes seems to be crucial to succeed with the classification algorithm in the dissimilarity space. The prototypes should avoid redundancies in terms of selection of similar samples, and prototypes should include as much information as possible [1], [2], [3]. However, it is difficult to find the optimal number of prototypes, and there is also a possibility that we lose some useful information for discrimination when selecting the prototypes.

Recently, to avoid these problems, researchers [4], [5] proposed an alternative approach where they used *all* available samples from the training set as prototypes, and subsequently apply dimensionality reduction schemes [5]. In the attempt to find the most appropriate reduction method, in this paper, we report a comparison between the computational complexities of prototype selection methods (PSM) and dimensionality reduction schemes (DRS) for dissimilarity-based classification. This is done by theoretically and experimentally demonstrating its strength in terms of processing time and classification accuracy.

2. Dissimilarity-Based Classification

Foundations of DBCs : A dissimilarity representation of a set of samples, $T = \{\mathbf{x}_i\}_{i=1}^n \in \Re^d$, is based on pairwise comparisons and is expressed as an $n \times m$ dissimilarity matrix $D_{T,Y}[\cdot,\cdot]$, where $Y = \{\mathbf{y}_1, \cdots, \mathbf{y}_m\}$, a prototype set, is extracted from T, and the subscripts of D represent the set of elements, on which the dissimilarities are evaluated. Thus, each entry $D_{T,Y}[i,j]$ corresponds to the dissimilarity between the pairs of objects $\langle \mathbf{x}_i, \mathbf{y}_j \rangle$, where $\mathbf{x}_i \in T$ and $\mathbf{y}_j \in Y$. Consequently, an object $\mathbf{x}_i$ is represented as a column vector as follows:

$$[d(\mathbf{x}_i, \mathbf{y}_1), d(\mathbf{x}_i, \mathbf{y}_2), \cdots, d(\mathbf{x}_i, \mathbf{y}_m)]^T, 1 \leqq i \leqq n. \qquad (1)$$

Here, the dissimilarity matrix $D_{T,Y}[\cdot,\cdot]$ is defined as a *dissimilarity space*, on which the d-dimensional object, $\mathbf{x}$, given in the feature space, is represented as an m-dimensional vector $\delta_Y(\mathbf{x})$.

For a training set $\{\mathbf{x}_i\}_{i=1}^n$ and an evaluation sample $\mathbf{z}$, the modified training set and sample now become $\{\delta_Y(\mathbf{x}_i)\}_{i=1}^n$ and $\delta_Y(\mathbf{z})$, respectively. From this perspective, we can see that the dissimilarity representation can be considered as a *mapping*, by which any arbitrary $\mathbf{x}$ is translated into $\delta_Y(\mathbf{x})$, 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 [1] reports the use of many traditional decision classifiers, including k-NN rule and the lin-

ear/quadratic normal-density-based classifiers, to the task of classifying $\mathbf{z}$ using $\delta_Y(\mathbf{z})$ in the dissimilarity space.

Prototype Selection Methods (PSM) : To select the representative set that is compact and capable of simultaneously representing the entire data set, Duin and colleagues [1] discussed the followings : *Random*, *RandomC*, *KCentres*, *ModeSeek*, *LinProg*, *FeatSeal*, *KCentres-LP*, and *EdiCon*.

In the interest of completeness, we briefly explain below the methods that are pertinent to our present study. Here, we assume c classes, a training set T, and the training subset T_i of the class ω_i. Each method selects m objects for the prototype set Y. If the algorithm is applied to each class separately, then m_i objects are chosen, such that $m = \sum_{i=1}^{c} m_i$.

(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 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 $Y_i = \{\mathbf{y}_1, \cdots, \mathbf{y}_{m_i}\}$ consisting of m_i objects, e.g., randomly chosen from T_i. (b) For each $\mathbf{x} \in T_i$, 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 $\mathbf{y}_j$ in Y_i. This means that $T_i = \cup_{j=1}^{m_i} N_j$. (c) For each N_j, find its center $\mathbf{c}_j$, 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 $\mathbf{c}_j$, if $\mathbf{c}_j \neq \mathbf{y}_j$, then replace $\mathbf{y}_j$ by $\mathbf{c}_j$ 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.

(4) *ModeSeek* : This method focuses on the modes in the dissimilarity data in the specified neighborhood size s. For each class ω_i, the algorithm proceeds as follows: (a) Set a relative neighborhood size as an integer $s > 1$. (b) For each $\mathbf{x} \in T_i$, find the dissimilarity $d_{s-NN}(\mathbf{x})$ to its s-th neighbor. (c) Find a set Y_i consisting of all $\mathbf{x}_j \in T_i$ for which $d_{s-NN}(\mathbf{x}_j)$ is minimum within its set of s neighbors.

From the experimental results of [2], the authors seem to have deliberated that systematic approaches lead to better results than those that 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 *FeatSel*, *LinProg*, *KCentres-LP*, and *EdiCon*, are omitted here in the interest of compactness, but can be found in the existing literature, including [1] and [3].

Dimensionality Reduction Schemes (DRS) : Various strategies have been used to tackle the "dimensionality reduction" problem (some of them are [6], [8], [9], [10], and [11]). To optimize DBCs, we can use a strategy of reducing the dimensionality after computing the dissimilarity matrix with the entire training samples. With regard to reducing the dimensionality of the dissimilarity matrix, we make use of the well-known dimensionality reduction schemes (DRSs) proposed in the literature. In the interest of completeness, we briefly explain below the methods that are pertinent to our present study (注1).

(1) *PCA* : We first compute the covariance matrix of the training set T after normalizing T. Next, we determine the eigenvectors $\mathbf{e}_i$ corresponding to the nonzero eigenvalues λ_i of the covariance matrix, where $\lambda_1 \geqq \cdots \geqq \lambda_d \geqq 0$. Then we can reduce the dimensionality of an object by representing it in a new coordinate system defined by the eigenvectors corresponding to the $m(< d)$ highest eigenvalues.

(2) *LDA* : This method uses the concept of a within-class scatter matrix, S_w, and a between-class scatter matrix, S_b, to maximize a separation criterion, such as $J = tr(S_W^{-1} S_B)$. We can obtain the solution by solving an eigenvalues problem on the matrix $S_w^{-1} S_b$, if S_w^{-1} is nonsingular, or on $S_b^{-1} S_w$ if S_b^{-1} is nonsingular. There are at most $c-1$ eigenvectors corresponding to nonzero eigenvalues since the rank of the matrix S_b is bounded by $c-1$. Therefore, the reduced dimension is at most $c-1$.

(3) *PCA-plus-LDA* : In this two-stage algorithm, the discriminant stage is preceded by a dimension reduction stage using PCA. However, its computation is expensive, and the PCA stage may also potentially lose some useful information for discrimination.

(4) *DCV* : This approach extracts the common properties of the training samples T_i of ω_i. The common vectors are then used for recognition. A common vector x_{com}^i is obtained by removing all the features that are in the direction of the eigenvectors $\mathbf{e}$'s corresponding to the nonzero eigenvalues of the scatter matrix of ω_i: set $Q = [\mathbf{e}_1, \cdots, \mathbf{e}_r]$; then $x_{com}^i = x_j^i - QQ^T x_j^i$, $j = 1, \cdots, n_i$, $i = 1, \cdots, c$, where r is the rank of the scatter matrix; finally obtain $Y = Q^T X$.

The details of other methods are omitted here in the interest of compactness, but can be found in the existing literature, including [9], [10], and [11].

3. The Computational Complexity

First, the time complexity of PSM is analyzed. Following

(注1) : Our overview is necessarily brief, but additional details can be found in [8], [9], [10], and [11].

this, the time complexity of DRS is given.

The Time Complexity of PSM : First of all, the time required for the *Random* algorithm is $t_{Rand} = t_{rand}(m)$, where m is the number of prototypes and $t_{rand}(m)$ is the time for generating m random numbers. From this analysis, the reader can observe that the time complexity of the algorithm is $O(m)$.

Next, the time complexity of the *RandomC* algorithm can be analyzed as follows: First, let the computation times for the operations of addition (or subtraction), substitution (or comparison), and multiplication (or division) be t_a, t_s, and t_m, respectively. The time required for initializing the algorithm is $t_1 = t_{rand}(m - m_i c) + (m - m_i t_s + 1)(t_m + t_a + t_s)$. Then, the time required for iterating a sub-step c times is $t_2 = c \times ((2 + m_i)t_a + t_{rand}(m_i) + 2m_i t_s + (m_i + 1)t_m)$. Thus, the total time required for the entire procedure to process many kinds (classes) of images under the condition $t_1 << t_2$ is $t_{RandC} = t_1 + t_2 \simeq t_2 = c \times (m_i(t_a + 2t_s + t_m) + t_{rand}(m_i))$. From the above analysis, the reader can observe that the time complexity of the algorithm is $O(m_i c) \simeq O(m)$, and the required time primarily depends on the parameter of m.

Third, the time complexity of the *KCentres* algorithm can be analyzed as follows: First, the time required for initializing the algorithm is $t_1 = t_{min}(n) + t_{min}(d) + t_s$. Then, the time required for the other steps is a sum of times to iterate the followings η times: $t_{(21)} = 3kt_{min}(d) + 3kt_s + t_{rand}(n)$; $t_{(22)} = k(d+1)t_a + k(n_i+1)t_{min}(n_i)$. Thus, the total time required for the entire procedure under the condition $t_1 << t_2$ is $t_{KCent} \simeq \eta(t_{(21)} + t_{(22)}) = \eta(3kt_{min}(d) + t_{rand}(n) + 3kt_s + dt_{min}(k) + kn_i t_{min}(n_i) + kdt_a)$, where k is the k in k-NN strategy, $t_{min}(n)$ is the time required to search for the minimum number from n numbers, and η, an experimental constant, is the number of trials to achieve the selection. From the above analysis, the reader can observe that the time complexity of the algorithm is $O(kn_i^2 + kd + n + k)$, and the required time primarily depends on the parameters of n, d, and k.

Finally, the time complexity of the *ModeSeek* algorithm can be analyzed as follows: First, the time required for step (a) is $t_{(a)} = (n + dn + nk)t_s$. Next, the time required for step (b) is $t_{(b)} = 2(n-2)dt_a + (d+1)(n-2)t_m + 2(n-2)t_s$. Then, the time needed for step (c) is a sum of times to compute the three sub-steps of (c1), (c2), and (c3) for all the n samples: $t_{(c1)} = (dn + 2n + 3)t_s + t_{min}(n)$; $t_{(c2)} = (k-1)(t_{min}(n) + 2t_s)$; $t_{(c3)} = n((2d+1)t_s + t_a)$. Thus, the total time required for repeating the three steps c times is $t_{ModeS} = c \times (t_{(a)} + t_{(b)} + nt_{(c)}) = c \times ((2dn + 5n + 2k - 6 + nk)t_s + (3n-4)t_a + (d+1)(n-2)t_m + t_{min}(n))$. From the above analysis, the reader can observe that the time complexity of the algorithm under the condition $k << d$ is $O(ncd + nck) \simeq O(ncd)$, and the required time primarily depends on the parameters of n, c, and d.

表 1 A comparison of the time complexities of the PSM and DRS techniques. The details of the table are discussed in the text.

Reduction Methods	Big-oh Computation
Random	$O(m)$
RandomC	$O(m)$
KCentres	$O(kn_i^2 + kd + n + k)$
ModeSeek	$O(ncd + nck) \simeq O(ncd)$
PCA	$O(n^2 d)$
LDA	$O(n^2 d)$
PCALDA	$O(n^2 d)$
DCV	$O(n^2 d + c^2 d)$

The Time Complexity of DRS : In [7], it is reported that the time complexities of LDA based methods, such as PCA, PCA+LDA, LDA/GSVD, and RLDA, respectively, are $O(n^2 d)$, $O(n^2 d)$, $O((n+c)^2 d)$, and $O(n^2 d)$, and their space complexities are all the same as $O(nd)$. The details of the above analysis are omitted here in the interest of compactness. Also, in [11], it is reported that DCV requires approximately $(2d(n-c)^2 + 4dnc)$ flops. From this report, the reader can observe that the time complexity of the algorithm is $O(n^2 d + c^2 d)$, and the required time primarily depends on the parameters of n, d, and c.

In summary, to examine the rationality of employing the dimensionality reduction schemes for DBCs, the time complexity required to reduce the dimensionality has been investigated. Table 1 shows a comparison between the time complexities of PSM and DRS techniques.

4. Experimental Results

The PSM and DRS based techniques have been tested and compared. This was done by performing experiments on the well-known benchmark database, namely, "UMIST" face database(注2).

We reduced the dimensionality of the dissimilarity matrix with a DRS, such as PCA [8], LDA [10], PCA-plus-LDA [9], or DCV [11]. In the DRS based techniques, we reduced the dimension $n-1$ to $c-1$, where n is the total number of training samples and c is the number of classes. In the PSM based techniques of *Random*, *RandomC*, *KCentres*, and *ModeSeek*, on the other hand, we selected $c-1$, c, c, and c samples from the training data set as the prototypes of DBCs.

We experimented different classifiers, such as the k-nearest neighbor classifiers (1-NN, 3-NN, 5-NN, 7-NN), the nearest mean classifiers (NMC), the support vector classifier (SVC), and the regularized normal density-based linear/quadratic classifiers (RLDC, RQDC). The classifiers were implemented

(注2) : http://images.ee.umist.ac.uk/danny/database.html

表 2 A comparison of the classification accuracy rates (%) of DBCs for the UMIST database between the PSM and DRS based techniques.

Experimental	1NN	3NN	5NN	7NN
Methods	NMC	RLDC	QLDC	SVC
Random	98.00	98.67	98.67	97.33
	95.33	95.33	99.33	98.67
RandomC	99.33	99.33	98.00	97.33
	97.33	98.00	99.33	100
KCentres	98.67	98.67	98.67	98.00
	96.00	88.00	93.00	98.67
ModeSeek	99.33	99.33	99.33	99.33
	99.33	99.33	99.33	98.67
PCA	99.33	99.33	99.33	99.33
	99.33	99.33	99.33	99.33
LDA	99.33	99.33	99.33	99.33
	99.33	99.33	99.33	99.33
PCALDA	99.33	99.33	99.33	99.33
	99.33	99.33	99.33	100
DCV	99.33	99.33	99.33	99.33
	99.33	99.33	98.67	100

with PRTools (注3).

In comparing the PSM and DRS based techniques, first of all, we measured the classification accuracies (%) of the DBCs for the real benchmark database. Table 2 shows a comparison of the classification accuracy rates (%) (for the process of prototype selection or dimensionality reduction) of DBCs for UMIST.

From Table 2, the reader can observe that the classification performances of the classifiers are improved with the DRS based techniques.

In comparing the PSM and DRS based techniques, we also measured the processing CPU-times (seconds) of the DBCs for the face database. Table 3 shows a comparison of the averaged processing CPU-times (for the process of prototype selection or dimensionality reduction) of DBCs for UMIST. Here, to measure the dissimilarities, we used Euclidean distance (ED), Hamming distance (HD), regional distance (RD) [12], or spatially weighted gray-level Hausdorff distance (WGHD) [13].

From Table 3, it is clearly observed that the processing CPU-times (seconds) increases when the DRS technique is applied.

5. Conclusion

In the attempt to reduce the dimensionality of dissimilarity representation, we can use dimensionality reduction schemes (DRS) as well as prototype selection methods (PSM). In this paper, we considered a comparison between the computational complexities of PSM and DRS techniques. This has been done by theoretically and experimentally analyzing their computational complexities. Our experimental results for the well-known benchmark facial images demonstrated the possibility that DRS could be used efficiently for dissimilarity-based classifiers (DBCs). However, it was also observed that the processing CPU-times (seconds) increased when the DRS technique was applied. The research concerning the reduction of the processing CPU-times is a future aim of the authors.

(注3)：http://www.prtools.org/

表 3 A comparison of the averaged processing CPU-times (seconds) of DBCs for the UMIST database. Each number of the table is obtained by averaging the results of five iterations on a Windows platform (CPU: 2.40 GHz, RAM: 2GB).

Experimental	UMIST			
Methods	ED	HD	RD	WGHD
Random	0.15	0.10	0.10	0.10
RandomC	0.27	0.25	0.26	0.27
KCentres	24.10	23.27	24.44	22.06
ModeSeek	5.91	5.95	5.85	5.85
PCA	52.76	46.32	44.91	43.00
LDA	1.94	1.32	1.35	1.33
PCALDA	42.500	42.50	38.98	41.29
DCV	13.88	13.63	13.751	13.86

Acknowledgments This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD- KRF-2007-313-D00714).

文　献

[1] E. Pekalska and R. P. W. Duin, *The Dissimilarity Representation for Pattern Recognition: Foundations and Applications*, World Scientific Publishing, Singapore, 2005.

[2] E. Pekalska, R. P. W. Duin, and P. Paclik, "Prototype selection for dissimilarity-based classifiers", *Patt. Recogn.*, vol. 39, pp. 189 - 208, 2006.

[3] S. -W. Kim and B. J. Oommen, "On using prototype reduction schemes to optimize dissimilarity-based classification", *Patt. Recogn.*, vol. 40, pp. 2946–2957, 2007.

[4] K. Riesen, V. Kilchherr, and H. Bunke, "Reducing the dimensionality of vector space embeddings of graphs", In *Proceedings of 5th Int. Conf. on Machine Learning and Data Mining*, vol. LNAI-4571, pp. 563–573, 2007.

[5] S. -W. Kim and J. Gao, "On Using Dimensionality Reduction Schemes to Optimize Dissimilarity-Based Classifiers", In *Proceedings of CIARP 2008*, Havana, Cuba, vol. LNCS-5197, pp. 302–309, 2008.

[6] M. Loog and R. P. W. Duin, "Linear dimensionality reduction via a heteroscedastic extension of LDA: The ChernoCriterion", *IEEE Trans. Pattern Anal. and Machine Intell.*, vol. PAMI-26, no. 6, pp. 732–739, 2004.

[7] J. Ye and Q. Li, "A two-stage linear discriminant analysis via QR-decomposition", *IEEE Trans. Pattern Anal. and Machine Intell.*, vol. 27, no. 6, pp. 929 - 941, Jun. 2005.

[8] 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, 2000.

[9] P. N. Belhumeour, J. P. Hespanha, and D. J. Kriegman,

"Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection", *IEEE Trans. Pattern Anal. and Machine Intell.*, vol. PAMI-19, no. 7, pp. 711–720, 1997.

[10] H. Yu and J. Yang, "A direct LDA algorithm for high-dimensional data - with application to face recognition", *Patt. Recogn.*, vol. 34, pp. 2067–2070, 2001.

[11] H. Cevikalp, M. Neamtu, M. Wilkes, and A. Barkana, "Discriminative common vectors for face recognition" , *IEEE Trans. Pattern Anal. and Machine Intell.*, vol. PAMI-27, no. 1, pp. 4–13, 2005.

[12] Y. Adini, Y. Moses, and S. Ullman, "Face Recognition: The problem of compensating for changes in illumination direction", *IEEE Trans. Pattern Anal. and Machine Intell.*, vol. PAMI-19, no. 7, pp. 721 - 732, 1997.

[13] S. -W. Kim, "Optimizing dissimilarity-based classifiers using a newly modified Hausdorff distance", In *Proceedings of 2006 Pacific Knowledge Acquisition Workshop*, Guilin, China, vol. LNAI-4303, pp. 177–186, 2006.