An Empirical Comparison of Kernel-Based and Dissimilarity-Based Feature Spaces*

Sang-Woon Kim[1] and Robert P. W. Duin[2]

[1] Dept. of Computer Science and Engineering,
Myongji University, Yongin, 449-728 South Korea
kimsw@mju.ac.kr
[2] Faculty of Electrical Engineering, Mathematics and Computer Science,
Delft University of Technology, The Netherlands
r.p.w.duin@tudelft.nl

Abstract. The aim of this paper is to find an answer to the question: What is the difference between dissimilarity-based classifications(DBCs) and other kernel-based classifications(KBCs)? In DBCs [11], classifiers are defined among classes; they are not based on the feature measurements of individual objects, but rather on a suitable dissimilarity measure among them. In KBCs [15], on the other hand, classifiers are designed in a high-dimensional feature space transformed from the original input feature space through kernels, such as a Mercer kernel. Thus, the difference that exists between the two approaches can be summarized as follows: The *distance* kernel of DBCs represents the discriminative information in a relative manner, i.e. through pairwise dissimilarity relations between two objects, while the *mapping* kernel of KBCs represents the discriminative information uniformly in a fixed way for all objects. In this paper, we report on an empirical evaluation of some classifiers built in the two different representation spaces: the dissimilarity space and the kernel space. Our experimental results, obtained with well-known benchmark databases, demonstrate that when the kernel parameters have not been appropriately chosen, DBCs always achieve better results than KBCs in terms of classification accuracies.

Keywords: kernel-based classifications (KBCs), dissimilarity-based classifications (DBCs), representation spaces, classification accuracies.

1 Introduction

Various kernel methods have been successfully used in the last decade to tackle complicated classification problems by a nonlinear mapping from the original input space to a kernel feature space [15]. Every learning algorithm that only makes use of inner products between data vectors can be transformed into a kernel method by means of replacing the inner product with an arbitrary kernel function [6]. The kernel function

* The work of the first author was partially done while visiting at Delft University of Technology, The Netherlands. We acknowledge financial support from the FET programme within the EU FP7, under the SIMBAD project (contract 213250). This work was supported by the National Research Foundation of Korea funded by the Korean Government (NRF-2009-0071283).

E.R. Hancock et al. (Eds.): SSPR & SPR 2010, LNCS 6218, pp. 559–568, 2010.

is typically viewed as providing an implicit mapping of sample points into a high-dimensional space, with the ability to gain much of the power of that space without paying the computational penalty[1]. Formally, let $\mathcal{X}$ denote the original pattern space and $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ be a function mapping pairs of patterns to real numbers. If the function k satisfies the condition of positive definiteness, there exists a vector space $\mathcal{F}$ and a mapping from $\mathcal{X}$ to $\mathcal{F}$, such that k acts as a dot product in $\mathcal{F}$ [15]. Such functions, k, are commonly called *kernel functions*.

The most popular representatives of kernel methods are support vector machines (SVMs) for classification problems [15]. SVMs are hyperplane classifiers in implicitly defined Euclidean feature spaces. A large number of applications reported in the literature indicate that SVMs are able to generalize well from unseen data and are not prone to overfitting. Other kernel methods for solving feature extraction and classification include principal component analysis [13], Fisher discriminant analysis [3], CLAFIC (CLAss Featuring Information Compression) [1], Gaussian mixture modeling [17], canonical correlation analysis [7], subspace discriminant analysis [4], locally linear embedding [16], and many others [15]. In the interest of brevity, the details of these kernel methods are omitted here, but can be found in the corresponding literature.

On the other hand, Duin and his co-authors [11], [12] proposed an alternative object representation system based on dissimilarities between objects using a generalized kernel approach. The concept of dissimilarity-based classifications is 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. Here, the dissimilarity measure can be defined for not only vectorial inputs, but also arbitrary non-vectorial patterns, such as strings, graphs, shapes, probabilistic models, etc. [9] Thus, this methodology can be considered a unified approach to statistical and structural pattern recognition [5], [9]. Furthermore, the 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 number of parameters [8].

In general, the kernels are understood as symmetric, positive definite functions of two variables, and, thereby, they express similarity between two objects represented in a feature space [15]. From this perspective, it is possible to regard a kernel as defining a *similarity measure* between the two variables. On the other hand, in [11], the kernels are addressed in a more general way, i.e., as a proximity measure. The important difference between these two types of kernels is summarized as follows: The *distance* dissimilarity kernel represents the information in a relative manner, i.e., through pairwise dissimilarity relations between the two objects; the *mapping* similarity kernel represents the information uniformly in a fixed way for all of the available objects.

Although classifications based on similarity kernels (which are referred to as kernel based classifications or KBCs) and classifications based on dissimilarity kernels (dissimilarity based classifications or DBCs) have been explored separately by many researchers, not much analysis has been done comparing the two. Therefore, the aim of this paper is to find an answer to the question: What is the difference between KBCs

[1] In the contrary of mapping objects into a high-dimensional space, a kernel function can also be viewed as a mapping to a low-dimensional space. The details of this kind of kernel method are omitted here, but can be found in [2].

and DBCs? or, more specifically, How different are these systems in their classification accuracies?

In this paper, we report an empirical comparison of KBCs and DBCs, which are built in two different representation feature spaces, respectively: dissimilarity-based feature spaces and kernel-based feature spaces[2]. Although it is hard quantitatively to evaluate the various KBC and DBC schemes, we have attempted to do exactly this. To achieve this goal, we have done a number of experiments with different methods to render this comparative study more complete. In KBCs, all samples are mapped to a higher-dimensional feature space using a kernel function; traditional classifications are then performed in the transformed feature space. In DBCs, on the other hand, dissimilarity-based feature spaces are directly obtained from all of the available objects; the same classifications are then done in the transformed feature space. Our experimental results obtained with well-known benchmark databases demonstrate that the classification performances obtained with KBCs and DBCs are almost the same. However, when the kernel parameters have not been appropriately chosen, it seems that DBCs are better than KBCs in terms of classification accuracy.

The main contribution of this paper is to demonstrate that the discriminative information of the dissimilarity-based feature space is less sensitive than that of the kernel-based feature space in choosing function parameters. This realization has been gained by executing classifications in the two feature spaces obtained with the training data sets and by comparing their strengths in terms of classification accuracy. Although many researchers have investigated the fact that SVMs are vulnerable to function parameters, to the best of our knowledge there is currently no reported empirical comparison of kernel-based and dissimilarity-based feature spaces.

2 Related Work

Kernel-Based Classifications (KBCs): In the implementation of kernel methods, the data is processed using a kernel to create a kernel matrix, which in turn is processed by a learning algorithm to produce a pattern function. This function is used to recognize unseen examples. Here, it is interesting to note that the resulting systems are modular: any kernel can be combined with any learning algorithm and vice versa [15].

Consider an embedding map $\phi : x \in \mathbb{R}^d \longmapsto \phi(x) \in \mathcal{F}$, where the choice of the map, ϕ, aims to convert the nonlinear relations into linear ones. Given a kernel and a training set, we can form a matrix known as a kernel matrix or Gram matrix, a matrix containing the evaluation of the kernel function on all pairs of data points [15]. To put it concretely, given a set of vectors $T = \{x_i\}_{i=1}^n, x_i \in \mathbb{R}^d$, the kernel matrix, K, is defined as the $n \times n$ matrix whose entries are $K_{ij} =< x_i, x_j >$. If we are using a kernel function, k, to evaluate the inner products in a feature space, $\mathcal{F}$, with a feature map, ϕ, the associated Gram matrix has entries: $K_{ij} =< \phi(x_i), \phi(x_j) >= k(x_i, x_j)$. Here, the Gram matrix, which is defined as a *kernel-based feature space*, is *positive semi-definite* (for details, see Proposition (3.7) of [15]).

[2] In this paper, we use the term 'feature space' for what we have called a vector space in pattern recognition unless otherwise mentioned.

The overall procedure for KBCs is summarized as follows:

1. Compute a kernel matrix, K, using a given training data set, $T = \{x_i\}_{i=1}^n$, and a kernel function, $k(\cdot, \cdot)$;

2. Compute the normalized eigenvectors of $K \in \mathbb{R}^{n \times n}$ in $\mathcal{F}$, and select a subspace dimension, q, to generate a transformation matrix, $A \in \mathbb{R}^{n \times q}$;

3. For a testing object, we compute a projection of the object onto the subspace using the transformation matrix A;

4. Achieve the classification through invoking a classifier built in the transformed subspace obtained with A and operating on the projected vector.

In the above algorithm, the kernel functions, $k(x_i, x_j)$, for example, such as *Polynomial*, *Radial basis*, or *Minkowski* function, can be defined as follows: $(x_i^T x_j + 1)^p$, $exp\left(-||x_i - x_j||^2)/p^2\right)$, or $\left(\sum |x_i - x_j|^p\right)^{1/p}$. Here, p's are the function parameters, such as function degree (d), standard deviation (σ), and degree order ($p \geq 1$), respectively. Among these kernels, the *Radial basis* function is the most widely used and has been extensively studied in this field. The parameter σ controls the flexibility of the kernel in a way similar to that of the degree d in the *Polynomial* kernel.

Dissimilarity-Based Classifications (DBCs): A dissimilarity representation of a set of samples, $T = \{x_i\}_{i=1}^n \in \mathbb{R}^{d \times n}$, is based on pairwise comparisons and is expressed, for example, as an $n \times m$ dissimilarity matrix $D_{T,Y}[\cdot, \cdot]$, where $Y = \{y_j\}_{j=1}^m \in \mathbb{R}^{d \times m}$, 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, $\langle x_i, y_j \rangle$, where $x_i \in T$ and $y_j \in Y$.

Here, the dissimilarity matrix, $D_{T,Y}[\cdot, \cdot] \in \mathbb{R}^{n \times m}$, is defined as a *dissimilarity-based feature space*, on which the d-dimensional object, x, given in the feature space, is represented as an m-dimensional vector $\delta(x, Y)$, where if $x = x_i$, $\delta(x_i, Y)$ is the i-th row of $D_{T,Y}[\cdot, \cdot]$. In this paper, the dissimilarity matrix $D_{T,Y}[\cdot, \cdot]$ and the column vector $\delta(x, Y)$ are simply denoted by $D(T, Y)$ and $\delta_Y(x)$ (or $D(x, Y)$), respectively. Here $\delta_Y(x)$ is an m-dimensional vector, while x is d-dimensional.

A conventional algorithm for DBCs is summarized in the following:

1. Select the representative set Y from the training set T by resorting to a selection method, such as *Random*, *RandomC*, or *KCentres* algorithm, as described in [11];

2. Compute the matrix $D(T, Y)$, using T, by employing a measuring system, such as the Euclidean distance, $d_E = ((x - y)^T (x - y))^{1/2}$, for all $x \in T$ and $y \in Y$;

3. For a testing sample z, compute a dissimilarity column vector, $\delta_Y(z)$, by using the same measure used in Step 2;

4. Achieve the classification through invoking a classifier built in the dissimilarity space and operating on the dissimilarity vector $\delta_Y(z)$.

In the above two algorithms, the dimensions of the two classification spaces can be reduced with the cardinality of the representation set and the number of the chosen eigenvectors, respectively. However, to reduce the computational complexity of this experiment, we first construct the dissimilarity matrix D and the kernel matrix K with respect to *all* the training samples. Then, we reduce the dimensionality of the spaces by performing a principal component analysis (PCA).

Kernel Matrix Versus Dissimilarity Matrix: Assume a training set T of n samples, a prototype set Y of m samples, and a nonnegative dissimilarity measure d. Then, an object, x, is represented as a dissimilarity vector of $D(x, Y) = [d(x, y_1), \cdots, d(x, y_m)]^T$. If a similarity measure k is used instead, we will get a similarity representation defined by similarity vectors of $K(x, Y) = [k(x, y_1), \cdots, k(x, y_m)]^T$. Here, if $|T| = |Y|$ and k is *positive semi-definite*, then K is a kernel matrix [12].

If the dissimilarity d is designed first, then k is defined as follows: $k(x_i, y_j) = \frac{1}{2}\left(d^2(x_i, 0) + d^2(0, y_j) - d^2(x_i, y_j)\right)$, where 0 represents a specific element that acts as a reference. On the other hand, if the similarity k is defined first, then d is computed as follows: $d^2(x_i, y_j) = k(x_i, x_i) + k(y_j, y_j) - 2k(x_i, y_j)$. In the interest of compactness, the details of the derivation are omitted here, but can be found in the literature [6],[15].

Kernel methods are powerful, but often cannot handle arbitrary proximities without incorporating necessary corrections, such as Euclidean corrections [12]. For example, a symmetric dissimilarity matrix $D(T, T) \in \mathbb{R}^{n \times n}$ can be embedded in a pseudo-Euclidean space by an isometric mapping [12]. The pseudo Euclidean space $\mathcal{E}(= \mathbb{R}^{(p,q)} = \mathbb{R}^{(p)} \oplus \mathbb{R}^{(q)})$ is denoted with signature (p, q), where the bilinear, but not necessarily positive definite, inner product is defined as $< z, z' >_{pE} := z^T M_{pq} z'$, where M_{pq} is $diag(\mathbf{1}_p, -\mathbf{1}_q)$ and $\mathbf{1}_n$ is an n-element vector of 1's. Also, the squared dissimilarity distance, $\|z - z'\|^2_{pE}$, may not define a metric, as it can violate the triangle inequality. That is, the squared norm and the squared distance can be negative in contrast to the Euclidean case. The details of determining the pseudo-Euclidean space to refine the dissimilarity representation are omitted here, but can be found in the literature, including [11] and [12].

3 Experimental Results

Experimental Data: The two classifying approaches, DBCs and KBCs, have been implemented and compared. This was done by performing experiments on three well-known benchmark image databases, namely Nist38, RoadSign, and Kimia2. The data set captioned "Nist38", chosen from the NIST database [18], consists of two kinds of digits, 3 and 8, for a total of 1000 binary images. The size of each image is 32×32 pixels, for a total dimensionality of 1024 pixels. The data set described as "RoadSign" consists of gray-level images of circular road signs: Three hundred road signs and the same number of outlier images [10], in which each image is 32×32 pixels, for a total dimensionality of 1024 pixels. The data set named "Kimia2" consists of two groups of images, each of 9 categories of 12 objects, obtained from the Kimia database [14]. The size of each image is 64×64 pixels, for a total dimensionality of 4096 pixels.

Experimental Method: In this experiment, first, data sets are split into training sets and test sets in the ratio of 75 : 25. Then, the training and testing procedures are repeated 30 times and the results obtained are averaged. Also, in contrast with many other papers on dissimilarities, we start by a feature representation and not with given dissimilarities between raw objects. That is because we want to make a comparison with kernels that also start in the feature space.

To evaluate DBCs and KBCs, different classifiers, such as k-nearest neighbor classifiers, linear Bayes normal classifier, quadratic Bayes normal classifier, and support

vector classifier, are employed and implemented with PRTools[3], and will be denoted as *knnc*, *ldc*, *qdc*, and *svc*, respectively, in subsequent sections.

In DBCs, the Euclidean distance between two samples is computed to measure their dissimilarity. Also, in KBCs, three mapping functions, *Polynomial*, *Radial basis*, and *Minkowski*, are employed as kernel functions. However, it is well known that selecting a proper kernel parameter with good class separability plays a significant role in kernel-based algorithms. In this experiment, therefore, to find optimal or near-optimal kernel parameters, in the case of the polynomial function, five function degrees, $p = \{s|s = 1, 2, \cdots, 5\}$, are tested. Then, in the case of the Minkowski function, five l_p distances, $p = \left\{2^{(s-1)}|s = 1, 2, \cdots, 5\right\}$, are examined. Finally, for the radial basis function, five deviation values, $p = \{\sigma_o(1.2 - 0.2s)|s = 1, 2, \cdots, 5\}$, are investigated. Here σ_o is determined after estimating the performance of the classifiers through cross-validation.

Experimental Results: The run-time characteristics of the DBC and KBC schemes for the experimental databases are reported below. First, the experimental results obtained with *qdc* and *ldc* trained in the dissimilarity space (shortly D) and the polynomial kernel space (shortly K) were probed into. Fig. 1 shows a 3-dimensional comparison of the error rates of *qdc* trained in the D and K spaces for Nist38. Here, x, y, and z axes are those of dimensions (which are obtained with PCA), kernel parameters (the degrees of the polynomial function), and the estimated error rates, respectively.

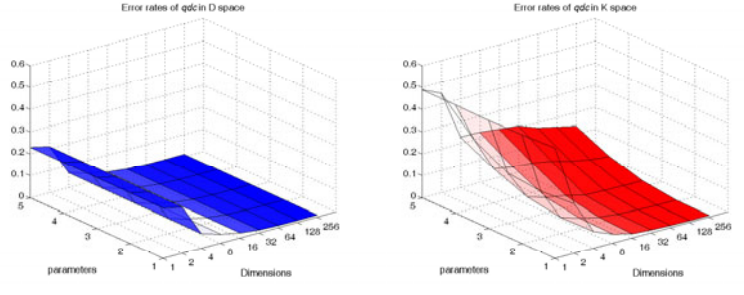

Fig. 1. A 3D comparison of the error rates of *qdc* for Nist38: (a) left and (b)right; (a) and (b) are obtained in D and K spaces, respectively, with different degrees of the polynomial function

From the figure, it can be observed that the two error rates obtained in D and K spaces are different, which implies that selecting an appropriate kernel parameter is essential for KBCs. This characteristic can be observed again in a subsequent experiment.

In principle, the quadratic Bayesian classifier could be better than the linear Bayesian classifier, but it requires far more training samples for estimation of the class covariance matrices. It is also well known that for 2-class problems with equally distributed samples, the quadratic classifier is equivalent to the linear one. Fig. 2 shows a comparison of the error rates of *qdc* and *ldc* trained in D and K spaces for Nist38.

[3] PRTools is a Matlab toolbox for pattern recognition(refer to http://prtools.org/).

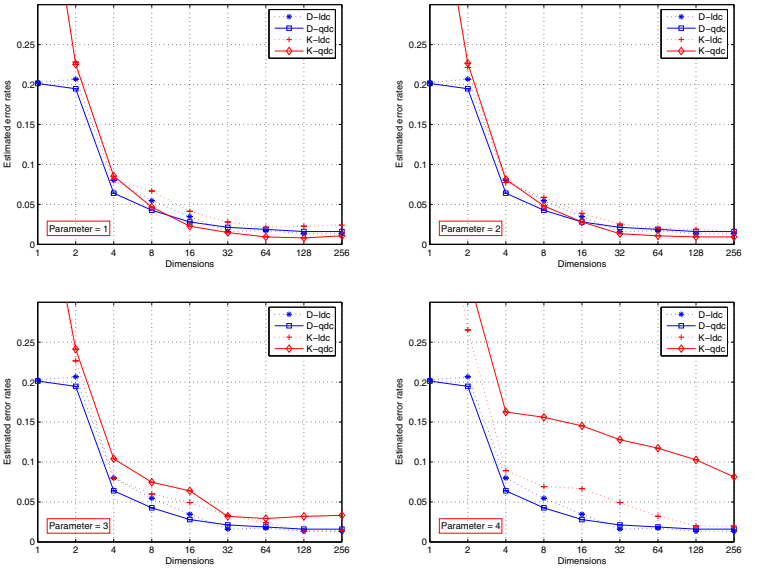

Fig. 2. A comparison of the error rates of *ldc* and *qdc* for Nist38: (a) top left, (b) top right, (c) bottom left, and (d) bottom right; (a) - (d) are obtained in D and K spaces with the four polynomial kernel parameters (degrees) of $s = 1, 2, 3$, and 4, respectively

In the figure, it should be pointed out that the difference in the estimated error rates between *qdc* and *ldc* for Nist38 increases as the value of the parameter increases. This is clearly shown in the error rates represented with two red lines (dashed and solid) in the four pictures of Fig. 2. This comparison shows that the classification accuracy of *qdc* is marginally higher than that of *ldc* when the appropriate parameter is present (refer to Fig. 2 (a) and (b)). However, the situation changes when an inappropriate parameter is chosen (refer to Fig. 2 (d)). From this consideration, the reader should again observe that choosing an appropriate kernel parameter plays an important role in KBCs. The same characteristic could also be seen in the other databases, such as RoadSign and Kimia2. The details for the results of these databases are omitted here to avoid repetition.

Second, as the main result, to investigate the difference of DBCs and KBCs further, the experiment (of estimating error rates) was repeated in other kernel spaces, such as *Polynomial*, *Radial basis*, and *Minkowski* spaces (which are shortly referred to as K_p, K_r, and K_m, respectively). Graphical comparisons of the error rates of the four classifiers trained in the dissimilarity based and the kernel based feature spaces are continually presented. Fig. 3 shows a comparison of the error rates of *knnc*, *ldc*, *qdc*, and *svc*, respectively, for Nist38.

The observations obtained from the figures are the following: (1) In general, the error rates of the classifiers trained in D space decrease constantly as the dimension increases, while those of the classifiers trained in $3K$'s spaces strongly depend on the kernel parameters. (2) As can be observed in the pictures in the left column of Fig. 3, when choosing an appropriate function parameter, all of the classifiers built in D and

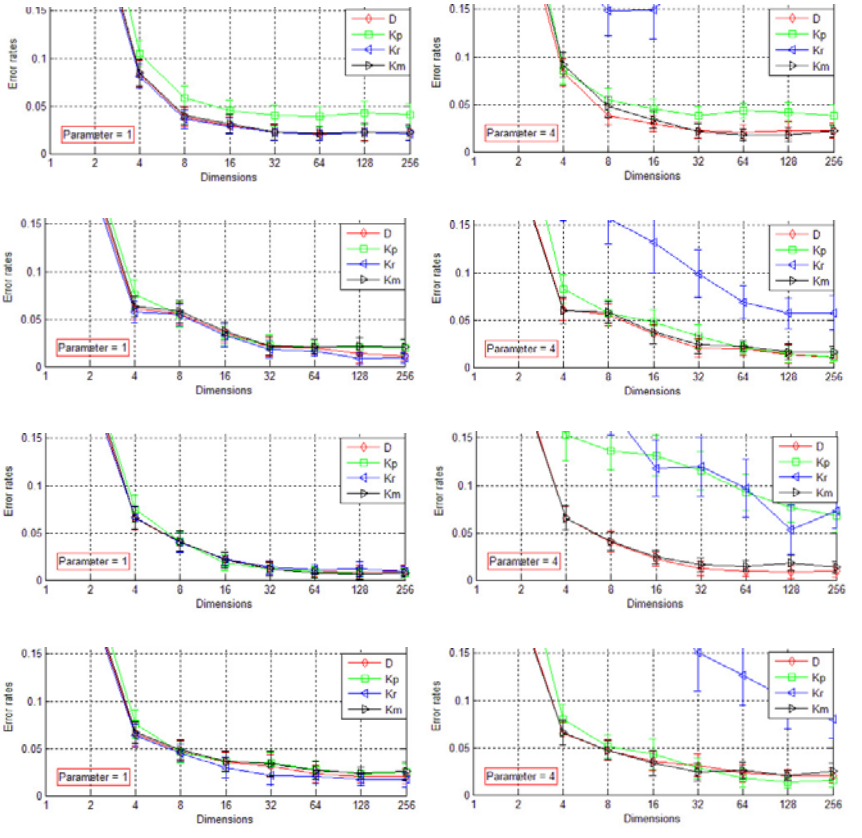

Fig. 3. A comparison of the error rates of *knnc*, *ldc*, *qdc*, and *svc* built in D and $3K$'s spaces with the kernel parameters of 1 and 4 for Nist38: (a) top left, (b) top right, $\cdots$, (g) bottom left, and (h) bottom right; (a) - (b) are of *knnc*, (c) - (d) are of *ldc*, (e) - (f) are of *qdc*, and (g) - (h) are of *svc*

$3K$'s have *almost* the same classification accuracies. (3) Specifically, the classification accuracy of *svc* is the best one obtained in K_r space. However, the classifier does not work satisfactorily in the kernel-based feature space with a wrong parameter, i.e., $s = 4$. (4) When the chosen parameters are far from optimal, the ranking of the discriminative power of the kernel-based feature space is K_m, K_p, and K_r. That is, the best discriminative power is that of K_m, while the worst one is that of K_r. The same characteristic could also be observed in the other databases, such as RoadSign and Kimia2. The details for the results of these databases are omitted here again in the interest of compactness.

Finally, it is an interesting issue to observe how robust to noise the classifiers trained in D and $3K$'s spaces are. To find reason for this phenomenon, we assume that the sample x_i is obtained by a noisy perturbation on the sample. This perturbation can be perceived as the inclusion of some additional noise θ[4], and, thus, we write: $x_i \leftarrow x_i + \theta$.

[4] $\theta(\cdot)$ refers to the noise generation random variables.

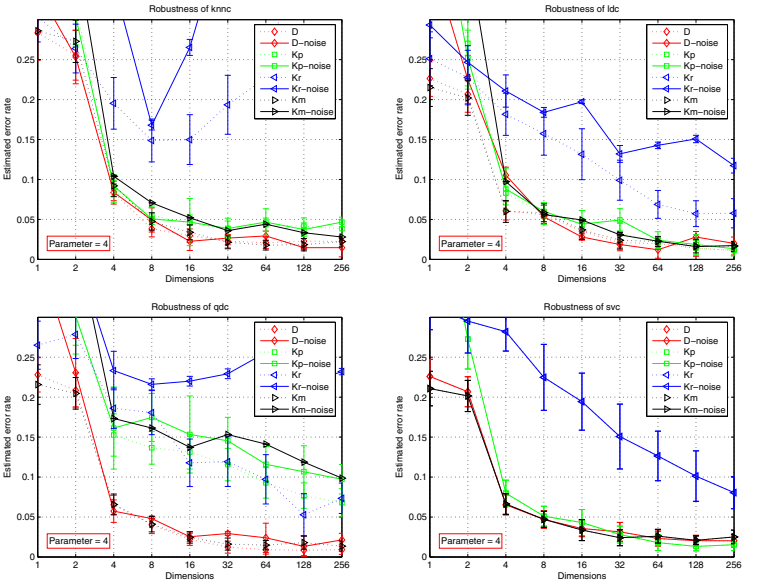

Fig. 4. A comparison of the error rates of *knnc*, *ldc*, *qdc*, and *svc* for the noisy Nist38: (a) top left, (b) top right, (c) bottom left, and (d) bottom right; (a) - (d) are obtained in D and $3K$'s spaces with kernel parameter "4"

For example, the noisy data can be obtained as: $x_i \leftarrow x_i * (1 + \epsilon * rand)$; Here, the function *rand* is to generate an array of random numbers whose elements are normally distributed with mean 0 and variance 1; ϵ is an experimental constant. Fig. 4 shows a comparison of the error rates of *knnc*, *ldc*, *qdc*, and *svc* trained in D and $3K$'s for the noisy Nist38. Here, $\epsilon = 0.3$.

From the figure, it should be observed that the differences in the estimated error rates of DBCs and KBCs obtained from the originally transformed feature space and their noisy perturbation spaces are different. This is clearly shown in the error rates of *qdc* represented with two red lines (dashed and solid lines of $\diamond$ marker) and two blue lines (dashed and solid lines of $\triangleleft$ marker) in Fig. 2(c). From this consideration, the reader should observe that the robustness of DBCs is higher than that of KBCs when there is a badly chosen parameter.

4 Conclusions

In this paper, we performed an empirical comparison of kernel-based classifications (KBCs) and dissimilarity-based classifications (DBCs). A number of classifiers designed in the two feature spaces were tested on well-known benchmark databases, and the classification accuracies obtained were compared. Our experimental results demonstrated that the classification accuracies obtained with KBCs and DBCs were almost the same when there was an appropriate kernel parameter. However, when the parameter

was not chosen appropriately, it seemed that the accuracies of DBCs were better than those of the KBCs. Especially, the results demonstrated that support vector classifiers of KBCs were vulnerable to function parameters. Despite this success, problems remain to be addressed. First, in this comparison we employed only three real life databases, in which each feature component of all objects was uniformly distributed in a fixed manner. Thus, evaluating the dissimilarity relations represented in a relative way is an avenue for future work. Next, to improve the internal consistency of the representation matrices, we could correct the matrices using pseudo-Euclidean embedding algorithms. Therefore, the problem of investigating the embedding algorithms developed for KBCs and DBCs remains to be done. Future research will address these concerns.

References

1. Balachander, T., Kothari, R.: Kernel based subspace pattern recognition. In: Proc. of Int'l Joint Conference on Neural Networks, Washington DC, USA, vol. 5, pp. 3119–3122 (1999)
2. Balcan, M.-F., Blum, A., Vempala, S.: Kernels as features: On kernels, margins, and low-dimensional mappings. Machine Learning 65, 79–94 (2006)
3. Baudat, G., Anouar, F.: Generalized discriminant analysis using a kernel approach. Neural Comput. 12(10), 2385–2404 (2000)
4. Chen, B., Yuan, L., Liu, H., Bao, Z.: Kernel subclass discriminant analysis. Neurocomputing 71, 455–458 (2007)
5. Goldfarb, L.: A unified approach to pattern recognition. Pattern Recogni. 17, 575–582 (1984)
6. Haasdonk, B.: Feature space interpretation of SVMs with indefinite kernels. IEEE Trans. Pattern Anal. and Machine Intell. 25(5), 482–492 (2005)
7. Hardoon, D.R., Szedmak, S., Shawe-Taylor, J.: Canonical correlation analysis: An overview with application to learning methods. Neural Comput. 16, 2639–2664 (2004)
8. Kim, S.-W., Oommen, B.J.: On using prototype reduction schemes to optimize dissimilarity-based classification. Pattern Recognition 40, 2946–2957 (2007)
9. Neuhaus, M., Bunke, H.: Edit distance-based kernel functions for structural pattern classification. Pattern Recognition 39, 1852–1863 (2006)
10. Paclik, P., Novovicova, J., Somol, P., Pudil, P.: Road sign classification using Laplace kernel classifier. Pattern Recognition Lett. 21(13-14), 1165–1173 (2000)
11. Pekalska, E., Duin, R.P.W.: The Dissimilarity Representation for Pattern Recognition: Foundations and Applications. World Scientific Publishing, Singapore (2005)
12. Pekalska, E., Duin, R.P.W.: Beyond traditional kernels: Classification in two dissimilarity-based representation spaces. IEEE Trans. Sys. Man, and Cybern(C) 38(6), 727–744 (2008)
13. Schölkopf, B., Smola, A.J., Müller, K.-R.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput. 10, 1299–1319 (1998)
14. Sebastian, T.B., Klein, P.N., Kimia, B.B.: Recognition of shapes by editing shock graphs. In: Proc. of 8th IEEE Int'l Conf. on Computer Vision, Vancouver, Canada, pp. 755–762 (2001)
15. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press, Cambridge (2004)
16. Tsagaroulis, T., Hamza, A.B.: Kernel locally linear embedding algorithm for quality control. In: Sobh, T., Elleithy, K., Mahmood, A., Karim, M.A. (eds.) Novel Algorithms and Techniques in Telecommunications, Automation and Industrial Electronics, pp. 1–6. Springer, Heidelberg (2008)
17. Wang, J., Lee, J., Zhang, C.: Kernel Trick Embedded Gaussian Mixture Model. In: Gavaldá, R., Jantke, K.P., Takimoto, E. (eds.) ALT 2003. LNCS (LNAI), vol. 2842, pp. 159–174. Springer, Heidelberg (2003)
18. Wilson, C.L., Garris, M.D.: Handprinted Character Database 3, Technical report, National Institute of Standards and Technology, Gaithersburg, Maryland (1992)