Metric Learning in Dissimilarity Space for Improved Nearest Neighbor Performance

Robert P.W. Duin[1], Manuele Bicego[2], Mauricio Orozco-Alzate[3], Sang-Woon Kim[4], and Marco Loog[1]

[1] PRLab, Delft University of Technology, The Netherlands
{r.p.w.duin,m.loog}@tudelft.nl
[2] Department of Computer Science,
University of Verona, 37134, Verona, Italy
manuele.bicego@univr.it
[3] Departamento de Informática y Computación,
Universidad Nacional de Colombia, Sede Manizales, Colombia
morozcoa@unal.edu.co
[4] Dept. of Computer Science and Engineering,
Myongji University, Yongin,
449-728 South Korea
kimsw@mju.ac.kr

Abstract. Showing the nearest neighbor is a useful explanation for the result of an automatic classification. Given, expert defined, distance measures may be improved on the basis of a training set. We study several proposals to optimize such measures for nearest neighbor classification, explicitly including non-Euclidean measures. Some of them may directly improve the distance measure, others may construct a dissimilarity space for which the Euclidean distances show significantly better performances. Results are application dependent and raise the question what characteristics of the original distance measures influence the possibilities of metric learning.

1 Introduction

The Nearest Neighbor (NN) rule is a classical and very natural classifier. It does not need density estimation or function optimization as it entirely relies on the user defined distance measure. An important advantage is that it gives an intuitive motivation of the assigned class label by showing the nearest neighbor(s) to the user. A second advantage is that the distance measure fully determines the classification performance as there is no learning involved. All is based on the collection of training examples.

The second advantage is also a disadvantage as it shows that there is room for improvement by using a training set. In case the original objects are represented in a vector space, e.g. by features, the performance may be improved by selecting or rescaling features. Such methods can also be considered as procedures for metric learning. In general, metric learning aims to find a better distance measure between objects on the basis of a training set.

P. Fränti et al. (Eds.): S+SSPR 2014, LNCS 8621, pp. 183–192, 2014.

Studies on metric learning either focus on adaptations of the vector space, preserving the original Euclidean distance, or optimize the metric, preserving the given vector representation, or combine a set of given distance measures. Examples are the Large Margin NN Classifier [11] and the Direct Minimization of the NN Error [3].

We will primarily deal with given, possible non-Euclidean, dissimilarities. New dissimilarity measures defined on the given ones will be proposed and evaluated. This may also yield a non-Euclidean result. We will use the word dissimilarity to emphasize that we allow ill-defined measures that even may violate the triangle inequality. This is in line with many applications based on images, shapes or sequences. It will not harm the use of the NN rule as long as there is a monotonic relation between measured dissimilarities and object differences.

An important possibility that we include in our considerations is that dissimilarities may be used to define a dissimilarity space [4],[6] and that in this space a distance measure is defined that combines the dissimilarities to the objects in the representation set that constitutes the dissimilarity space.

The vector space defined by the dissimilarity representation differs from the feature representation by the mentioned monotonic relation, as well as by the natural correlations arising from using similar objects for representation. Three proposals using these characteristics will be evaluated for some public domain real-world datasets. For evaluation, the performance of the NN rule will be used.

In Section 2 the three proposals will be presented. They are evaluated with the direct NN performance on the given distances as well as with the NN performance in the dissimilarity space. In Section 3 the datasets and some of their properties are reported. Results are presented in Section 4 and conclusions are summarized in the final section.

2 Methods

Let X be a set of labeled training objects $X = \{x_i, i = 1, ..., n\}$ and let x be an arbitrary object inside or outside X. The objects are initially only represented by their dissimilarities $\mathbf{d}(x) = [d(x, x_i), i = 1, ..., n]$. These dissimilarities are defined by some expert (e.g. as function of raw measurements on x and x_i) in such a way that if $d(x, x_1) < d(x, x_2)$ it is more likely that x belongs to the same class as x_1 than that it belongs to the class of x_2. For that reason the NN rule using $\mathbf{d}(x)$ is an appropriate classifier.

We are searching for a modified dissimilarity measure $d_{mod}(x, x_i)$ being a function of all distances to the training set $\mathbf{d}(x)$ such that the performance of the NN rule improves. Any such procedure can be used directly by classifying new objects on the basis of their modified dissimilarities. Below we discuss one existing and three new procedures that will be evaluated in Section 4.

The training set used for metric learning is a square dissimilarity matrix

$$D = [\mathbf{d}(x_1), \mathbf{d}(x_2), ..., \mathbf{d}(x_n)] \tag{1}$$

It is not always symmetric and some procedures allow even non-zero diagonals. When needed we make it symmetric by averaging and force a zero-diagonal.

Such a matrix can be embedded in a $(n-1)$-dimensional pseudo-Euclidean space (PE-Space) [6] that consists of two Euclidean subspaces. These are built by an eigenvalue decomposition of a Gram matrix derived from (1). The eigenvectors corresponding to the positive eigenvalues constitute the positive space, the other ones constitute the negative space. For Euclidean dissimilarity matrices the dimensionality of the latter is zero as in that case all eigenvalues are positive. In this paper the PE-Space will only be used to characterize the dissimilarities.

2.1 Dissimilarity Space, DS

A straightforward way to derive new dissimilarities to a given set of representative objects (the representation set) by combining the available ones is the dissimilarity space, [6]. This is the vector space constructed by the vector of distances as mentioned in the previous subsection: $\mathbf{d}(x) = [d(x, x_i), i = 1, n]$. Here we will use the training set for representation as well. If we use Euclidean distances in the dissimilarity space the modified dissimilarity can be written as:

$$d_{DS}(x, x_i) = \|\mathbf{d}(x) - \mathbf{d}(x_i)\|$$

It has been found in the past [6] that the NN performance may improve as well as deteriorate by this modification. It is still an open issue to find the conditions when one or the other may happen.

2.2 Locally Adaptive Nearest Neighbor Distances, LANN

The locally adaptive distance measure was originally proposed by Wang et al. [10], claiming that it significantly improves the performance of the kNN rule when used with a metric distance measure. The rationale behind their local adaptation approach is simple and elegant: dividing a conventional distance measure —the authors restricted themselves to the Euclidean and Manhattan metrics for five feature-based data sets— by the smallest distances from the corresponding training examples to training examples of different classes. We study the application of the procedure, referred as LANN, to given and unconstrained dissimilarity measures. More formally, LANN can be described as follows.

Let d be a dissimilarity measure and x and x_i be a test object and a training object, respectively. Let r_i be the radius of the largest topological ball[1] around x_i that excludes —in the corresponding PE-space— all training objects from other classes. This radius is given by

$$r_i = \min_{j:\theta_j \neq \theta_i} d(x_i, x_j)$$

where θ_i is the class label associated to the i-th training object.

The locally adaptive dissimilarity measure $d_{LANN}(x, x_i)$ is then defined as:

$$d_{LANN}(x, x_i) = \frac{d(x, x_i)}{r_i} \tag{2}$$

[1] Notice that depending on the dissimilarity measure, the neighborhoods defined by objects with dissimilarity to x_i less than r_i may not be a proper ball.

LANN can be understood as a columnwise scaling of the test dissimilarity matrix, where the scaling factors correspond to the radii associated to the training objects. Dissimilarities to training objects with large radii are diminished/rewarded since they are considered more trustable (a large neighborhood of the same class); conversely, dissimilarities to objects with small radii are, comparatively, emphasized/penalized (less trustable due to a small neighborhood of the same class). Two potential drawbacks associated to LANN are noise sensitivity and dependency on the sample size: notice that (i) outliers, even though not trustable, are associated to large radii and (ii) small training sample sizes will produce large but empty neighborhoods where unseen objects of different classes might lie in.

2.3 Non-linear Scaling of Dissimilarities

Here we explore the possibility of transforming the input dissimilarities by employing a non linear function: in particular we explore the effect of applying the power transformation to each pairwise dissimilarity:

$$d_{NLScale}(x, x_i) = d(x, x_i)^{\rho} \qquad \rho > 0 \tag{3}$$

Clearly, this operation does not have an impact on the NN rule based on the original dissimilarities[1], since a monotonic transformation does not change the ordering of objects. On the contrary, this operation may change the behavior of the NN rule in the dissimilarity space, as it represents a *non-linear scaling* of it.

In general, scaling feature spaces is often very useful, especially for classifiers based on the Euclidean distance or inner products (like NN or SVM). The typical choice in this context is to perform a *linear scaling*, like the well known z-score standardization (every feature is centered and divided by the standard deviation). Nevertheless, there can be situations where the linearity assumption is too restrictive, and a benefit may be obtained from a non-linear scaling, which acts in different ways in different parts of the feature space. One clear example of non-linear transformation, which has nevertheless scarcely applied in the classification context, is the well known Box-Cox transformation [1], [8], introduced in the 60's, representing a parametric way to non linearly transform a set of points in order to make their distribution approximately Gaussian. More recent approaches, explicitly devoted to the classification case, appeared in [2], where kernels for HMM-based generative embeddings were successfully augmented via a non-linear transformation of the space.

Here we propose to use this non-linear scaling to enhance the performances of the NN rule in the dissimilarity space. Dissimilarities appear to be an optimal context where to apply this non-linear mapping, for different reasons: i) the power mapping does not change the rankings of the objects, so the original information on which the space is built is preserved; ii) all the directions of the dissimilarity space share the same nature (they are all dissimilarities), therefore

[1] Even if useless in the NN case, this operation can be beneficial for other classification techniques, especially if they rely on the Euclideaness of the space: actually for $\rho < 1$ the Euclideaness of the dissimilarity matrix is increased by this non linear mapping.

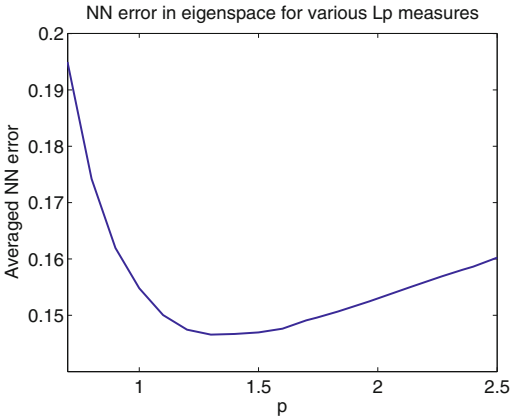

Fig. 1. The NN error in a set of 2-fold cross validation experiments (repeated 10 times) averaged over the 44 Chickenpieces datasets

it may be simpler to find a common good parameter for all the directions; iii) all directions are positives, avoiding strange effects for negative values.

In our implementation, the scaling factor ρ is optimized by a grid search between 0.03 and 30 by a leave-one-out cross-validation. This is still fast up to a few thousand objects (the dissimilarity matrix should fit in fast memory).

2.4 Distance in Eigenspace, ESL1.5

For various applications like histograms and images, other distance measures may be more appropriate than the Euclidean distance based on bins or pixels. In [5] it was suggested to use the L^1 metric. As other metrics than the Euclidean one (L^2) are rotation sensitive, it was suggested in that study to perform an eigenspace rotation first, thereby removing all correlation. (It is admitted that this part of the procedure uses L^2).

Dissimilarity spaces suffer, like pixel based representation, heavily from correlations. We wondered whether other distance measures than L^2 would make sense in the dissimilarity space.

The distance transformation can thereby be written as follows. First the total training set is considered in the dissimilarity space derived from the dissimilarity matrix (1). It coincides with the training set represented in the dissimilarity space. We compute the set of eigenvectors E, so $ED = \Lambda D$ with Λ a diagonal matrix. A vector $\mathbf{d}$(x) in the dissimilarity space is transformed to the eigenspace by

$$\mathbf{e}(x) = E\mathbf{d}(x)$$

The L^p distance in this space of an object x and a training object x_i is:

$$d_{ESLp}(x, x_i) = (\sum_j |e_j(x) - e_j(x_i)|^p)^{1/p} \tag{4}$$

in which $e_j(x)$ is the j-th component of $\mathbf{e}(x)$. Fig. 1 shows a preliminary experiment based on the Chickenpieces dissimilarity dataset, see Section 3. The

NN performances in a 2-fold cross validation experiment averaged of all Chickenpieces datasets are shown for L^p as a function of p. It shows that there is a significant minimum between $p = 1$ and $p = 2$. This appeared to be true in other experiments as well. In a more extensive study p might be optimized for every application. Here we decided to use always $p = 1.5$, avoiding additional cross-validation loops, and named the procedure ESL1.5.

3 Datasets

We use a set of public domain datasets, see Table 1. More information on the datasets themselves can be found on the internet[1]. Most datasets are obtained from real objects (images, text, protein sequences). PolyDisH57 and PolyDisM57 are the only two artificial datasets, obtained by the (modified) Hausdorff distance on randomly generated pentagons and heptagons. The Chickenpieces dataset consists out of 44 dissimilarity matrices. In the table, the average characteristics are shown. The Pendigits dataset is much larger. To make our experiments feasible we used a randomly selected subset of 4000 objects.

Here are short definitions of the properties used in Table 1, see also [4].

- *size*: the total number of objects in the dataset.
- *class*: the number of classes.
- *ID*: an estimate of the the intrinsic dimensionality.
- *LOO*: the leave-one-out NN error.
- *NEF*: the negative eigenfraction, a measure for the Euclideaness.
- *NMF*: the non-metricity fraction of triplets violating the triangle inequality.
- *SignP*: the number of positive eigenvalues in pseudo-Euclidean embedding[2].
- *SignN*: the number of negative eigenvalues in pseudo-Euclidean embedding.
- *Asym*: the averaged deviation of symmetric dissimilarity measure.

4 Evaluation

The procedures described in Section 2 are applied to all datasets mentioned in Section 3. A two-fold cross-validation is repeated 25 times. The errors found by the NN rule are averaged. The mean errors and the standard deviation of the means are listed in Table 2. Results that are significantly better than those obtained for the original dissimilarities are printed in bold. (We judge a difference in means as significant if the intervals defined by the two standard deviations do not overlap). In order to save space, the errors over the Chickenpieces datasets are averaged. Below they will be summarized in some figures.

Table 2 shows the results found by a direct use of the (modified) dissimilarities in the left of every column and the results of the corresponding dissimilarity space in the right. The two procedures LANN and ESL1.5 show many significant

[1] `http://37steps.com/prdisdata`

[2] The two numbers [SignP SignN] are called the *signature* of the embedding.

Table 1. Dataset properties

Dataset	*size*	*class*	*ID*	*LOO*	*NEF*	*NMF*	*SignP*	*SignN*	*Asym*
CatCortex	65	4	18	0.12	0.208	0.002	41	23	0.000
Chickenpieces	446	5	3	0.13	0.273	0.000	242	203	0.051
CoilDelftDiff	288	4	22	0.47	0.128	0.000	163	124	0.000
CoilDelftSame	288	4	13	0.65	0.027	0.000	249	38	0.000
CoilYork	288	4	4	0.23	0.258	0.000	169	118	0.009
DelftGestures	1500	20	6	0.04	0.308	0.000	765	734	0.000
FlowCyto	612	3	2	0.38	0.230	0.004	330	281	0.000
NewsGroups	600	4	83	0.25	0.202	0.000	153	387	0.000
Pendigits	4000	10	4	0.01	0.348	0.002	1944	2055	0.000
PolyDisH57	4000	2	9	0.03	0.415	0.000	2054	1945	0.000
PolyDisM57	4000	2	11	0.02	0.356	0.000	1819	2180	0.000
ProDom	2604	4	17	0.00	0.043	0.000	1502	680	0.000
Protein	213	4	14	0.02	0.001	0.000	205	4	0.000
WoodyPlants50	791	14	5	0.10	0.229	0.000	395	395	0.000
Zongker	2000	10	14	0.44	0.419	0.002	1038	961	0.000

improvements on the original dissimilarities. Note however that the ESL1.5 procedure itself already computes distances (using the L1.5 norm) in dissimilarity space. NLScale transforms the given dissimilarities by a monotonic transformation, the same for all dissimilarities. This does not influence the NN assignments as explained in Section 2.3. Its results on the given dissimilarities are thereby identical to the original ones. The results for its dissimilarity space (right column) show many significant results. In general, it is shown that metric learning may be useful for these datasets.

All Chickenpieces datasets refer to the same set of silhouettes. Bunke and Spillmann [9] just used different parameters in the weighted edit distance measure. They constitute thereby an interesting set of slightly changing dissimilarities. All results for these datasets are summarized in Fig. 2, clearly showing the improvements that are obtained by the various methods.

Since the errors associated to the studied methods correspond to coordinates in the vertical axis, dots below the line indicate that the modified dissimilarity measures are better than their original counterparts (since the lower the error, the better the performance). The further a dot is from the line, the greater the margin of improvement.

Below the individual procedures proposed in Section 2 are discussed separately.

The *dissimilarity space*, Section 2.1 (the right part of each of the columns in Table 2) is a general procedure to combine given dissimilarities into new ones by treating them as vectors. It is not focussed on improvement, but it puts pairwise dissimilarities in the context of all other objects. Sometimes the NN rule on the distances obtained from the dissimilarity space shows an improvement, sometimes it does not. It is an open issue to get a better understanding when this happens.

Table 2. Averaged two-fold cross validation results (error × 1000) for the NN-rule based on 25 repetitions. In every column on the left the NN errors on the dissimilarities, on the right the NN error in the corresponding dissimilarity space. In between brackets the standard deviation of the estimated mean errors. In bold the results that significantly improve the original dissimilarities.

Dataset	*Original*		*LANN*		*NLScale*		*ESL1.5*	
CatCortex	138(10)	**96**(7)	**96**(11)	126(11)	138(10)	**95**(8)	**88**(8)	**106**(8)
Chickenpieces	161(3)	**150**(2)	**123**(3)	156(2)	161(3)	**122**(2)	**144**(2)	216(3)
CoilDelftDiff	513(6)	**464**(7)	**465**(7)	**464**(6)	513(6)	**456**(7)	**450**(7)	531(9)
CoilDelftSame	656(6)	**410**(8)	**540**(8)	**423**(8)	656(6)	**425**(9)	**416**(8)	**517**(10)
CoilYork	319(5)	396(7)	333(5)	411(8)	319(5)	331(7)	392(8)	546(9)
DelftGestures	50(1)	95(1)	66(2)	97(1)	50(1)	54(2)	83(2)	187(2)
FlowCytoDis	403(4)	408(5)	**338**(4)	417(5)	403(4)	404(5)	403(4)	426(6)
NewsGroups	291(5)	293(6)	**269**(4)	332(6)	291(5)	293(5)	295(6)	341(7)
Pendigits	15(1)	23(1)	17(1)	30(1)	15(1)	16(1)	18(1)	61(1)
PolyDisH57	40(1)	**31**(1)	**22**(1)	**30**(1)	40(1)	**20**(1)	**30**(1)	84(1)
PolyDisM57	23(1)	**15**(1)	**12**(0)	**16**(0)	23(1)	**17**(1)	**16**(1)	22(1)
ProDom	9(1)	19(1)	**5**(1)	20(1)	9(1)	8(1)	13(1)	143(3)
Protein	37(5)	**6**(2)	**14**(3)	**4**(1)	37(5)	**8**(2)	**5**(1)	**17**(3)
WoodyPlants50	127(3)	165(3)	**119**(3)	204(3)	127(3)	121(3)	154(3)	263(3)
Zongker	358(25)	**53**(1)	**196**(21)	**130**(7)	358(25)	**40**(2)	**50**(1)	**114**(2)

Metric learning based on the *local adaptive NN procedure*, LANN, Section 2.2 performs remarkably well. It always shows improvements except for the three cases mentioned above. We were afraid that this procedure is very noise sensitive, but apparently the noise introduced by the arbitrary distances to the nearest neighbor does not harm. It is a simple, effective procedure that does not require any optimization.

Let us try to understand the behavior of the *non-linear scaling procedure*, NLScale, Section 2.3, concentrating on the case of $\rho < 1$ (for which we almost always got the best results). When using $\rho < 1$ lower dissimilarities are raised, whereas large ones are reduced. This operation has three effects:

- points tend to have the same distance from all the other points (since the dissimilarities tend to be all equal): this potentially augments the intrinsic dimensionality of the dataset (i.e. the dimensionality of the manifold where the objects lie). The larger this dimensionality, the more Euclidean (flat) the space: techniques relying on Euclidean assumptions (as the NN in the dissimilarity space) can benefit from this. Clearly, such correction can also destroy the information contained in the dissimilarities, as shown in [7].
- the contribution to the dissimilarity space of possible outliers is possibly reduced, since high distances – namely distances from very far points, i.e. outliers – are shrinked.
- the neighborhood of every point is enlarged: small distances, i.e. distances between near points, are emphasized, therefore augmenting the importance in the dissimilarity space of nearest points.

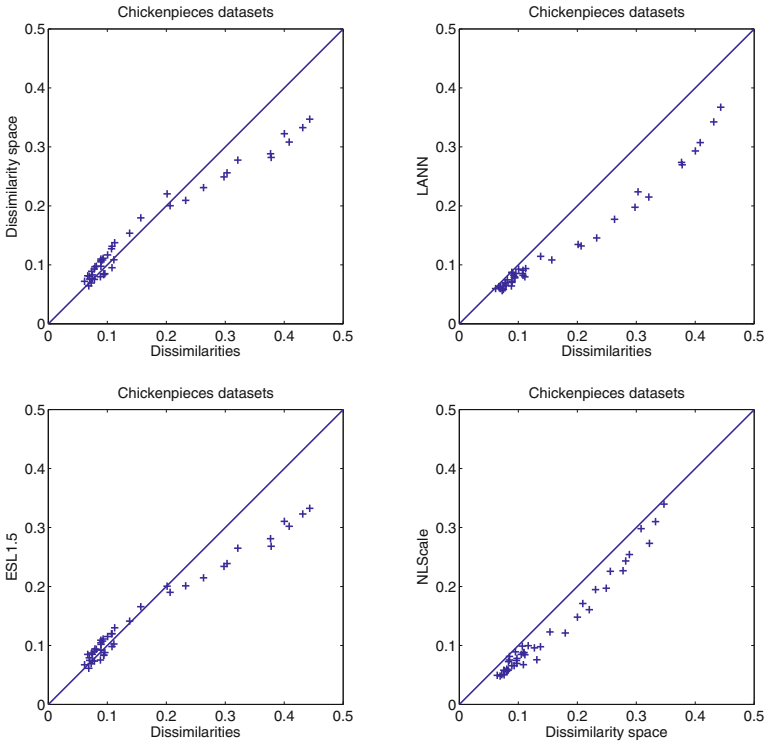

Fig. 2. Results for the 44 Chickenpieces datasets

The eigenspace procedure, ESL1.5, Section 2.4, effectively operates in the original dissimilarity space. For consistency we have printed in bold the significant differences with the original dissimilarity results themselves. Improvements in comparison with the dissimilarity space are less striking, but almost always shown. We conclude from this that the idea of using a non-Euclidean measure in the dissimilarity space (which is almost always used as an Euclidean space [6]) is effective.

5 Conclusion

This study is based on "given dissimilarities": dissimilarity datasets arising from applications, external to our study. In such applications the dissimilarity measure may have been optimized for the given objects. Thereby we might have sometimes made a second attempt to improve this measure by learning from a training set that has already been taken into account. We admit that thereby overtraining may be introduced by squeezing the data further. Nevertheless it is interesting that for 12 of the 15 datasets, one or even several significant improvements could be found. Systematic procedures for metric learning apparently make sense for NN classification.

The datasets have very diverse backgrounds and are based on entirely different dissimilarity measures. One may wonder whether from the dataset characteristics listed in Table 1 can be predicted which procedure for which dataset is promising (meta-learning). At this moment we cannot answer this in a positive way. It is, however, interesting that the datasets that could not be improved (CoilYork, DelftGestures and Pendigits) belong to the most non-Euclidean ones according to the NEF measure. PolyDisH57 and PolyDisM57 have a high NEF value as well, but their distance measures have not been optimized for the application. The ones that could not be improved are the result of studies in which the researchers tried to obtain an optimal result. This might explain both, their strong non-Euclidean behavior as well as the difficulty to improve the metric.

In conclusion, it has been shown that metric learning for a large variation of given, non-Euclidean dissimilarities is well possible and may yield significant improvements.

Acknowledgments. Support from Dirección de Investigación - Sede Manizales (DIMA), Universidad Nacional de Colombia, is acknowledged as well as the Cooperint program from University of Verona.

References

1. Box, G., Cox, D.: An analysis of transformations. Journal of the Royal Statistical Society: Series B (Methodological) 26(2), 211–252 (1964)
2. Carli, A., Bicego, M., Baldo, S., Murino, V.: Nonlinear mappings for generative kernels on latent variable models. In: ICPR, pp. 2134–2137 (2010)
3. Chernoff, K., Loog, M., Nielsen, M.: Metric learning by directly minimizing the k-NN training error. In: ICPR, pp. 1265–1268. IEEE (2012)
4. Duin, R., Pękalska, E., Loog, M.: Non-Euclidean dissimilarities: Causes, embedding and informativeness. In: Pelillo, M. (ed.) Similarity-Based Pattern Analysis and Recognition. Advances in Computer Vision and Pattern Recognition, pp. 13–44. Springer, London (2013)
5. Kim, S.-W., Duin, R.P.W.: Dissimilarity-based classifications in eigenspaces. In: San Martin, C., Kim, S.-W. (eds.) CIARP 2011. LNCS, vol. 7042, pp. 425–432. Springer, Heidelberg (2011)
6. Pękalska, E., Duin, R.: The Dissimilarity Representation for Pattern Recognition. Foundations and Applications. World Scientific, Singapore (2005)
7. Plasencia-Calaña, Y., Cheplygina, V., Duin, R.P.W., García-Reyes, E.B., Orozco-Alzate, M., Tax, D.M.J., Loog, M.: On the informativeness of asymmetric dissimilarities. In: Hancock, E., Pelillo, M. (eds.) SIMBAD 2013. LNCS, vol. 7953, pp. 75–89. Springer, Heidelberg (2013)
8. Sakia, R.: The Box-Cox transformation technique: a review. The Statistician 41, 169–178 (1992)
9. Spillmann, B.: Description of the distance matrices. Tech. rep. (2004), `http://www.iam.unibe.ch/fki/databases/string-edit-distance-matrices/dmdocu.pdf`
10. Wang, J., Neskovic, P., Cooper, L.N.: Improving nearest neighbor rule with a simple adaptive distance measure. Pattern Recognition Letters 28(2), 207–213 (2007)
11. Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. Journal of Machine Learning Research 10, 207–244 (2009)