Does the affinity matrix influence the performance of the Locality Preserving Projection algorithm?

Elias R. Silva Júnior, George D. C. Cavalcanti and Tsang Ing Ren
Federal University of Pernambuco
Center of Informatics
Recife - PE - Brazil
www.cin.ufpe.br/~viisar
{ersj2,gdcc,tir}@cin.ufpe.br

***Abstract*—Classical feature extraction techniques, like PCA and LDA, do not deal properly with multimodal problems. Such techniques create projections that do not preserve the multimodal structure of the original data distribution. Locality Preserving Projection (LPP) is a feature extraction technique which looks for a transformation matrix that minimizes the changes into the structure of the data after the transformation. This local structure is captured by the affinity matrix. However, there many ways to calculate this affinity matrix. The main aim of this paper is to evaluate the influence of different affinity matrices over the LPP accuracy. The experiments showed that the correct choice of the affinity matrix can lead to a performance gain. Among the analyzed affinity matrices, Local Scaling and Nearest Neighbor reached the best results.**

***Index Terms*—feature extraction; multimodality; affinity metrics.**

I. Introduction

It is difficult to deal with high dimensionality comprising many input variables because the demand for a large number of samples increases exponentially with the dimensionality of the input space - the curse of dimensionality [1]. One alternative to handle this problem is to reduce the dimensionality of the data before classification.

Feature extraction techniques are often used to dimensionality reduction. They perform a transformation over the original feature space resulting in a lower dimensional one. Besides, this transformation must keep or improve the class discriminative power of the data when compared to the original space. Once this new feature space is found, the whole datapoints are projected on it.

However, many feature extraction techniques do not consider that the data can be represented as a multimodal distribution (which has multiples maximum in a statistical distribution). This fact can lead to undesired results, because these techniques can lost the multimodal structure after the projection, creating undesired overlap between modes of different classes. Multimodal data can be observed in several applications, for instance: handwritten digit classification and biometric identification problems (1:N matching).

In order to transpose the lack of robustness of the classical feature extraction techniques when dealing with multimodal data, He and Niyogi [2] presented the *Locality Preserving Projections* (LPP) algorithm. LPP is an unsupervised technique which aims to preserve the local structure of the data neighborhood after the transformation. In contrast with traditional approaches, like Principal Component Analysis (PCA) and Fisher Discriminant Analysis (FDA), LPP is prepared to deal with multimodal problems, because it seeks for a transformation that minimizes the changes in the structural relation of the data.

The concept of affinity between the datapoints is used to express their structure in the feature space. Then, the affinity information is a crucial element in the development of LPP. There are many ways to calculate the affinity between datapoints: cosine, euclidean neighbor, heat kernel, local scaling and nearest neighbor. The goal of this paper is twofold: i) evaluate different metrics of affinity aiming to verify if they influence the performance of the LPP algorithm; and, ii) give directions for the use of the metrics, knowing that they influence the performance of LPP. This paper does not aim to compare the performance of LPP with other feature extraction techniques. For details about the performance of the LPP against other feature extraction techniques see [2] and [3].

This paper is organized as follows. Section II shows the theoretical basement of the LPP technique for dimensionality reduction and also shows different ways to calculate the affinity matrix. In Section III, the experimental setup is described. The experimental results are exposed in Section IV and the final considerations about this work is described in Section V.

II. Locality Preserving Projection

The Locality Preserving Projection is a linear feature extraction technique which aims to represent the datapoints from the original feature space to a lower dimensional space minimizing the changes into the structure of the datapoint's neighborhood after the transformation.

Based on a set of patterns $X = \{x_k | 1 \leq k \leq N\}$ with dimensionality d, where x_k is a $d \times 1$ matrix, the process of linear dimensionality reduction creates a set of patterns $Z = \{z_k | 1 \leq k \leq N\}$ with dimensionality r ($1 \leq r \leq d$), where z_k is a $r \times 1$ matrix such that $Z = T^t \times X$. Thus, T is the linear transformation matrix (T is a $d \times r$ matrix and $(.)^t$ is the matrix transpose operation). The parameter z_i is a representation of x_i in a lower dimensionality.

The LPP transformation matrix is showed in Equation 1. Equation 2 shows how to calculated D, a diagonal matrix of

978-1-4244-6588-0/10/$25.00 ©2010 IEEE

order N which is used to construct L, the graph-Laplacian matrix [2], [4] (Equation 3).

$$T_{LPP} = \underset{\mathbf{T}\in\mathbf{R}^{d\times r}}{\arg\min} \left(\sum_{i,j=1}^{N} \frac{A_{i,j}}{2} \left\| T^t x_i - T^t x_j \right\|^2 \right) \quad (1)$$

$$D_{i,i} = \sum_{j=1}^{N} A_{i,j} \quad (2)$$

$$L = D - A \quad (3)$$

with $L_{i,j} = D_{i,j} - A_{i,j}\ \forall i,j \in \{1,\ldots,N\}$.

The T_{LPP} transformation matrix consists of the eigenvectors corresponding to the r shortest eigenvalues obtained from Equation 4.

$$\frac{XLX^t}{XDX^t}\psi = \gamma\psi \quad (4)$$

Figure 1 shows the behavior of PCA and LPP in a multimodal problem. In this figure, the PCA does not obtain a satisfactory projection, while the LPP achieves a good projection.

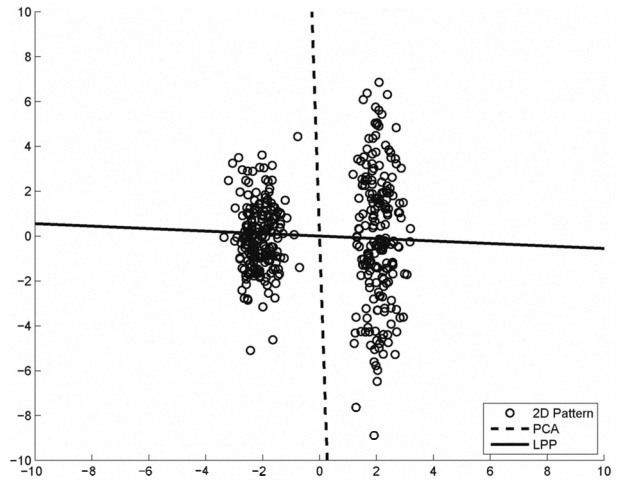

Fig. 1. A comparison between LPP and PCA in a multimodal example.

The affinity matrix is used in order to preserve the local structure of the patterns. This matrix expresses the degree of similarity between the patterns. Given the affinity matrix A of order N, the affinity between the patterns x_i and x_j is $A_{i,j}$ ($A_{i,j} \in [0,1]$). If $A_{i,j} \to 1$ then x_i and x_j are near, if $A_{i,j} \to 0$ then x_i and x_j are far away. The affinity matrix, also named as similarity matrix, must have the following properties: i) non-negative; ii) symmetric; iii) invertible.

Sugiyama [3] showed four metrics to calculate the affinity between patterns: Euclidean Neighbor (Equation 5), Heat Kernel (Equation 6), Local Scaling (Equations 7 and 8) and Nearest Neighbor (Equation 9).

$$A_{i,j} = \|x_i - x_j\| \leq \epsilon \quad (5)$$

where $\epsilon > 0$ is a tuning parameter and $A_{i,j} = 1$ if the distance between x_i and x_j is less or equal to ϵ, otherwise $A_{i,j} = 0$.

$$A_{i,j} = \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma^2} \right) \quad (6)$$

where $\sigma > 0$ is a tuning parameter that controls the decay of the affinity.

$$A_{i,j} = \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma_i \sigma_j} \right) \quad (7)$$

$$\sigma_i = \left\| x_i - x_i^K \right\| \quad (8)$$

where x_i^K represents the K^{th} nearest neighbor of x_i and σ_i is the distance between x_i and its K^{th} nearest neighbor (x_i^K).

$$A_{i,j} = \left(x_i \in NN_j^{(K)} \right) \vee \left(x_j \in NN_i^{(K)} \right) \quad (9)$$

where $NN_i^{(K)}$ is the K nearest neighbors set of x_i. As the euclidean neighbor metric, the nearest neighbor takes only binary values.

$$A_{i,j} = \frac{x_i^t x_j}{\|x_i\| \, \|x_j\|} \quad (10)$$

Li et al. [5] reported a cosine-based metric to calculate the affinity matrix (Equation 10). This equation does not have free parameters and its affinity measure is calculated by the cosine rule between the patterns x_i and x_j. In other word, pairs of distinct patterns, which keep the same angle between them, have the same affinity value, independently of the distance between the patterns. Additionally, this affinity calculation accepts negative values, which does not match with the given affinity definition.

III. Experimental Settings

The main goal of this paper is to evaluate different metrics to compute the affinity matrix and show their influence over the LPP performance. Figure 2 displays the architecture of these experiments.

The training set has its affinity matrix computed by the five metrics used here: (i) cosine, (ii) euclidean neighbor, (iii) heat kernel, (iv) local scaling and (v) nearest neighbor.

The euclidean neighbor and the heat kernel metrics have their free parameter: ϵ and σ, respectively. These parameters are tuned by a Genetic Algorithm (GA) in order to minimize the negative influence of them over the metric performance. The local scaling and the nearest neighbor metrics have the same free parameter $K = \{1, 3, 5, 7, 9, 11, 13\}$. Then, each one of these metrics generate seven affinity matrices (one affinity matrix for each value of K). The metric cosine does not have free parameter, then it is directly applied over the training set.

Once the affinity matrix was obtained, LPP is performed over this matrix and the resulting transformation matrix is evaluated using the test set.

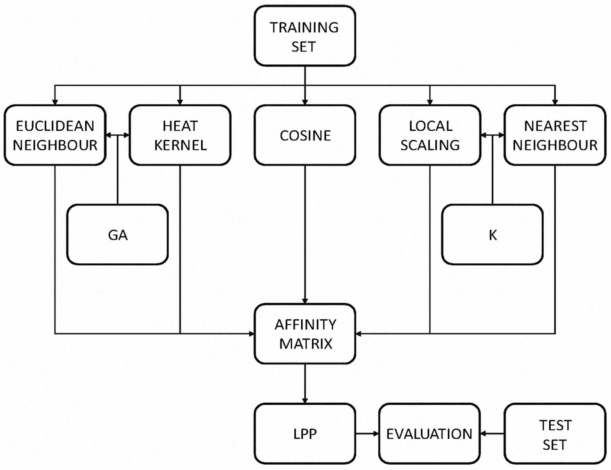

Fig. 2. Diagram of the experimental architecture.

A. Datasets

The datasets used in this paper can be originally found in the UCI Machine Learning repository [6]. However, we use a version that can be obtained in [7]. This version stores random samples from each dataset (20 or 100 rounds of sampling depending from the dataset). This datasets was also used by Sugiyama [3].

Table I shows each used dataset with their number of: training patterns, test patterns, executions (sampling) performed, attributes and classes.

TABLE I
SELECTED DATASETS

Dataset	Training	Test	Executions	Attributes	Classes
banana	400	4900	100	2	2
breast-cancer	200	77	100	9	2
diabetis	468	300	100	8	2
german	700	300	100	20	2
heart	170	100	100	13	2
image	1300	1010	20	18	2
ringnorm	400	7000	100	20	2
splice	1000	2175	20	60	2
twonorm	400	7000	100	20	2
waveform	400	4600	100	21	2

B. GA Settings

The GA architecture used to tuning the parameter ϵ of the euclidean neighbor metric is the same one used to tuning σ of the heat kernel metric.

The population size was set to 20 chromosomes and the maximum number of generations to 100. The chromosomes are binary bitstrings that represent the value of the parameter to be tuned. The conversion from the bitstring to a decimal value (phenotype) is given by Equation 11. $p(d)$ represents the phenotype of the chromosome, min_p and max_p are the minimum and maximum values, L is the length of the bitstring and d is the decimal representation of the bitstring (for example: $B_2(1101) = B_{10}(13)$).

$$p(d) = min_p + \frac{max_p - min_p}{2^L - 1} \times d \tag{11}$$

where $min_p = 1 \times 10^{-4}$, $max_p = 1 \times 10^4$ and $L = 20$.

The crossover fraction was set to 0.7 with two cut points (two segments of each chromosome are changed). The mutation fraction was set to 0.01 with uniform gene selection.

In order to preserve the best solution of each generation, *elitism* is used. It sends the most adapted chromosome of the current generation to the next generation. The selection policy used was *roulette wheel* [8] which gives to the best adapted chromosomes more chances to be present in the next generation.

Figure 3 displays the *fitness function* diagram of the GA. Each chromosome phenotype represents a tuned value of the free parameter of the affinity metric. This value is applied to one of the metrics (Euclidean Neighbor – EuN or Heat Kernel – HK) which enables the calculation of the affinity matrix using the GA training set. After that, the LPP transformation matrix (T) is computed. The fitness value of the chromosome is given as the accuracy of a bayesian classifier over the projected test set.

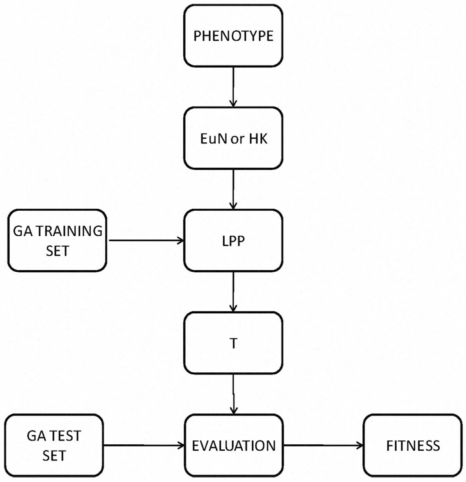

Fig. 3. Diagram of the GA fitness function.

The GA algorithm stops if the overall best solution found does not improve for 10 generations or if the maximum number of generations is reached.

C. Bayesian Classifier

The Bayesian classifier was implemented as described in the algorithm below. From lines 1 to 3, the algorithm calculates the mean (μ_l), the covariance (Σ_l) and the *a priori* probability ($P(w_l)$) of each class l – the training dataset is used here. In line 4, N_{ts} represents the number of patterns in the test set. Each test pattern x_i is projected onto the feature space given by the LPP algorithm – T is the transformation matrix and $\widetilde{x}_i$ is the representation of the test pattern into the new feature space. After that, in line 7, $P(w_l|\widetilde{x}_i)$ represents the probability of $\widetilde{x}_i$ given the class w_l. $\mathcal{N}(l, \widetilde{x}_i) = \mathcal{N}(T^t\mu_l,\ T^t\Sigma_l T,\ \widetilde{x}_i)$ which

is the estimate of $f(w_l|\widetilde{x}_i)$ by the normal distribution. In line 9, it is stored the predicted label of the test pattern.

Algorithm*[Bayesian Classifier]*

(1) **For:** $l \longleftarrow 1:c$
(2) **Compute:** $\mu_l,\ \Sigma_l,\ P(w_l)$
(3) **End For**
(4) **For:** $i \longleftarrow 1:N_{ts}$
(5) **Project:** $\widetilde{x}_i = T \times x_i$
(6) **For:** $l \longleftarrow 1:c$
(7) **Compute:** $P(w_l|\widetilde{x}_i) = \frac{\mathcal{N}(l,\widetilde{x}_i)}{\sum_{l=1}^{c}\mathcal{N}(l,\widetilde{x}_i)P(w_l)}$
(8) **End For**
(9) **Compute:** $Pred(i) = \arg\max_{1\leq l\leq c} P(w_l|\widetilde{x}_i)$
(10) **End For**
(11) **Return:** $Pred$

TABLE II
LPP FEATURE EXTRACTION TIME IN SECONDS PER AFFINITY METRIC

Affinity	$\overline{x}$ (σ)
Cos	0.5563 (0.1348)
EuN	0.5465 (0.1728)
HK	0.6020 (0.1655)
LS1	4.0046 (0.6135)
LS3	3.9532 (0.6039)
LS5	4.0966 (0.5923)
LS7	4.2501 (0.6775)
LS9	4.1131 (0.6727)
LS11	4.1776 (0.7030)
LS13	4.0786 (0.6446)
NN1	2.5779 (0.0249)
NN3	2.5658 (0.0137)
NN5	2.5554 (0.0136)
NN7	2.5450 (0.0128)
NN9	2.5354 (0.0137)
NN11	2.5229 (0.0129)
NN13	2.5136 (0.0127)

IV. RESULTS

The affinity metrics used in the experiments were: (i) Cosine – *Cos*, (ii) Euclidean Neighbor – *EuN* (tuned via GA), (iii) Heat Kernel – *HK* (tuned via GA), (iv) Local Scaling – *LSK* (where K assumes one of the values $\{1, 3, 5, 7, 9, 11, 13\}$, e.g., for $K = 3$ we have *LS3*) and (v) Nearest Neighbor – *NNK* (where K assumes one of the values {1, 3, 5, 7, 9, 11, 13}, e.g., for $K = 5$ we have *NN5*).

The error rate of the Bayesian classifier for each one of the 17 variations of the affinity metrics applied to each one of the 10 datasets was calculated. This calculation was performed in each one of the executions of each dataset. Thus, the mean error and its standard deviation are shown.

Table II shows the computational time required to perform the LPP feature extraction with each affinity metric used. This table reports the time in seconds obtained for the dataset *banana* (the relation between the times of the metric can be extrapolated to other datasets). Only the *banana* time results are presented because all metrics applied over this dataset achieved the best solution with 2 dimensions. This behavior does not happen with the other datasets. These times refer only to the feature extraction effort, the time spent to tuning the free parameters are not computed.

The local scaling (LSK) and the nearest neighbor (NNK) were the slowest metrics (for any K used), although the nearest neighbor was ever faster than the local scaling. The other metrics (cosine, euclidean neighbor and heat kernel) were the fastest ones, being more than four times faster than local scaling and nearest neighbor. However, the feature extraction times obtained through the cosine, euclidean neighbor and heat kernel metrics were statistically equivalent and it is not possible to distinguish between the fastest one of them.

A paired *t*-test analysis (significance at 5%) was performed to find the best dimension for the pair (dataset, affinity metric). This ideal dimension is the one which better discriminate the patterns in their classes. This analysis was performed comparing the relative distribution of the dimension that achieved the shortest mean error against the distribution of the other dimensions in the same combination (dataset *vs.* affinity metric). Thus, this best dimension is the one with the slowest dimensionality (resulting from the paired *t*-test) in which the null hypothesis was not rejected.

Table III displays the best dimensions obtained for each pair (dataset, affinity metric) and the values inside the parenthesis are the original dimensionality of each dataset. For some datasets, the variation of the best dimensionalities, using different metrics, were not expressive. Usually the cosine metric reached the highest dimensionality followed by the euclidean neighbor metric. In almost all the analyzed datasets, the local scaling metric achieved the smallest dimensions. As the local scaling, the nearest neighbor metric presented few or none variation of the best dimensionality achieved with the variation of K. Although, this best dimensionality tend to be great or equal the local scaling one (with few exceptions).

Table IV shows the mean error rate and the standard deviation (in parentheses) calculated using the best dimensions presented in Table III . It is possible to see that, different features extraction techniques achieved statistically equivalent solutions (for most cases). This makes hard the comparing between them.

In order to provide a better visualization of the results in Table IV, a plot was constructed: Figure 4. These figures reinforce the statistically equivalence of the feature extraction techniques for each dataset (with few exceptions), which can be seen as the intersection between the confidence interval of the techniques, for each dataset. Although, observing Figure 4, it is easy to note that the metrics local scaling and nearest neighbor achieved the smallest mean error rate for almost all datasets. Even if one of them do not reach the smallest mean error rate, at least, this error is statistically equivalent to the smallest one (the splice dataset is the only exception).

Based on the strong results equivalence, it is not possible to determine what is the best K for the metrics local scaling and nearest neighbor. However, the local scaling was more stable than the nearest neighbor.

TABLE III
THE BEST DIMENSIONS OBTAINED USING THE PAIRED t-TEST

	banana (2)	breast-cancer (9)	diabetis (8)	german (20)	heart (13)	image (18)	ringnorm (20)	splice (60)	twonorm (20)	waveform (21)
Cos	2	1	8	19	13	5	20	12	20	21
EuN	2	5	7	10	9	6	17	38	11	2
HK	2	1	6	7	2	3	19	11	2	3
LS1	2	1	6	9	1	11	18	8	1	2
LS3	2	1	6	9	1	11	18	8	1	2
LS5	2	1	6	9	1	12	18	8	1	2
LS7	2	1	6	8	1	12	18	8	1	2
LS9	2	1	6	8	1	12	18	8	1	2
LS11	2	1	6	7	1	12	18	8	1	2
LS13	2	1	6	7	1	12	18	8	1	2
NN1	2	1	6	14	3	12	19	57	2	2
NN3	2	1	6	14	2	10	19	60	1	2
NN5	2	1	6	14	1	10	19	59	1	2
NN7	2	1	6	14	1	10	19	59	1	2
NN9	2	1	6	13	1	11	19	59	1	2
NN11	2	1	6	13	1	11	19	60	1	2
NN13	2	1	6	14	1	11	19	60	1	2

TABLE IV
MEAN ERRORS: AFFINITY METRIC PER DATASET (%)

	banana	breast-cancer	diabetis	german	heart	image	ringnorm	splice	twonorm	waveform
Cos	39.072	28.909	26.327	27.483	19.440	27.619	2.579	15.834	3.457	18.066
	(3.455)	(4.339)	(1.928)	(2.882)	(3.379)	(9.816)	(0.351)	(0.964)	(0.300)	(1.049)
EuN	39.072	26.584	28.970	29.083	25.310	32.916	4.262	12.152	2.665	12.189
	(3.455)	(4.287)	(6.342)	(3.533)	(7.171)	(1.806)	(2.635)	(0.582)	(0.186)	(0.359)
HK	39.072	25.714	26.010	25.967	18.740	38.307	2.589	15.522	2.388	12.193
	(3.455)	(4.487)	(2.361)	(2.089)	(3.871)	(9.301)	(0.310)	(0.823)	(0.127)	(0.394)
LS1	39.072	25.753	26.557	27.537	18.230	10.480	2.647	15.078	2.429	12.156
	(3.455)	(4.570)	(2.007)	(3.216)	(3.690)	(1.910)	(0.265)	(0.789)	(0.125)	(0.358)
LS3	39.072	25.675	26.610	27.287	18.020	10.470	2.640	15.080	2.443	12.157
	(3.455)	(4.483)	(1.922)	(3.327)	(3.553)	(1.412)	(0.273)	(0.911)	(0.128)	(0.337)
LS5	39.072	25.662	26.603	27.023	18.120	9.787	2.643	15.152	2.443	12.150
	(3.455)	(4.604)	(1.901)	(3.257)	(3.616)	(1.096)	(0.268)	(0.947)	(0.129)	(0.337)
LS7	39.072	25.844	26.550	27.757	18.120	9.842	2.640	15.113	2.451	12.163
	(3.455)	(4.591)	(1.882)	(3.537)	(3.691)	(1.094)	(0.273)	(0.952)	(0.134)	(0.340)
LS9	39.072	25.948	26.517	27.333	18.150	9.822	2.635	15.122	2.455	12.156
	(3.455)	(4.548)	(1.964)	(3.276)	(3.644)	(1.081)	(0.270)	(0.977)	(0.136)	(0.348)
LS11	39.072	26.013	26.493	27.333	18.190	9.856	2.641	15.092	2.456	12.163
	(3.455)	(4.642)	(1.973)	(3.080)	(3.648)	(1.075)	(0.277)	(0.966)	(0.138)	(0.341)
LS13	39.072	26.065	26.473	26.880	18.130	9.861	2.642	15.087	2.456	12.163
	(3.455)	(4.649)	(1.950)	(2.948)	(3.670)	(1.092)	(0.277)	(0.953)	(0.140)	(0.352)
NN1	39.072	29.636	25.560	27.783	19.580	9.965	2.672	16.375	2.837	12.459
	(3.455)	(4.985)	(2.061)	(2.707)	(4.283)	(1.154)	(0.349)	(1.224)	(0.254)	(0.424)
NN3	39.072	27.727	25.910	27.697	18.950	12.411	2.654	15.726	2.580	12.126
	(3.455)	(4.646)	(2.033)	(2.670)	(4.152)	(2.757)	(0.342)	(0.938)	(0.191)	(0.393)
NN5	39.072	26.442	25.543	27.617	18.130	11.950	2.625	16.025	2.513	12.088
	(3.455)	(4.421)	(1.924)	(2.594)	(3.662)	(1.674)	(0.314)	(1.191)	(0.168)	(0.365)
NN7	39.072	26.442	25.537	27.563	17.500	12.248	2.592	16.218	2.495	12.121
	(3.455)	(4.428)	(1.921)	(2.612)	(3.622)	(1.640)	(0.285)	(1.013)	(0.163)	(0.337)
NN9	39.072	26.545	25.563	27.943	16.940	11.777	2.573	16.455	2.485	12.112
	(3.455)	(4.968)	(1.946)	(3.195)	(3.601)	(1.249)	(0.289)	(1.319)	(0.157)	(0.345)
NN11	39.072	27.091	25.690	28.043	16.430	11.965	2.580	15.726	2.470	12.110
	(3.455)	(5.102)	(1.958)	(3.188)	(3.468)	(1.278)	(0.298)	(0.938)	(0.161)	(0.349)
NN13	39.072	27.519	25.790	27.637	16.190	11.960	2.599	15.726	2.482	12.104
	(3.455)	(4.897)	(1.974)	(2.767)	(3.547)	(1.272)	(0.305)	(0.938)	(0.155)	(0.349)

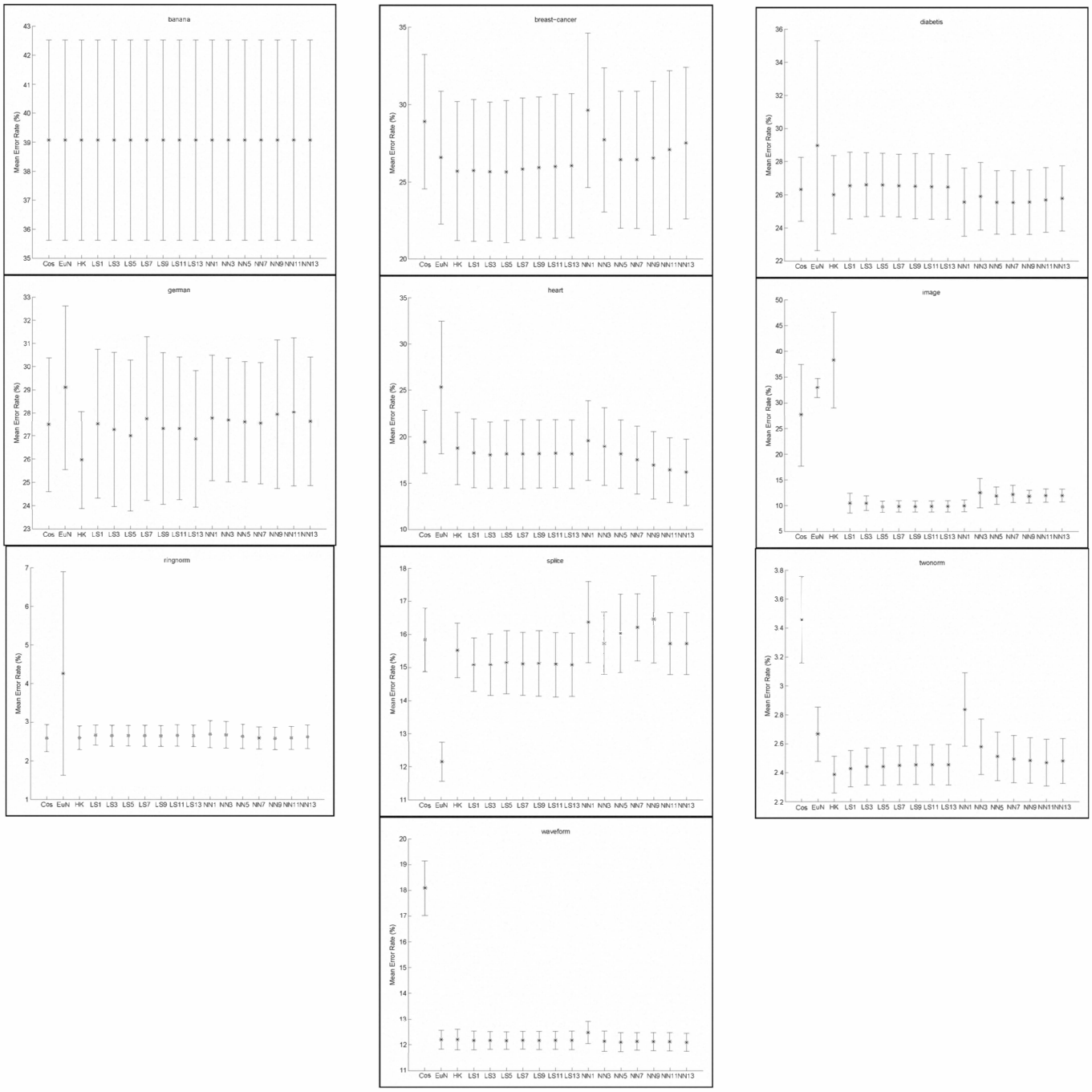

Fig. 4. Graphics of the error rates of each affinity metric to a specific dataset (see Table IV for data details).

V. Conclusion

This paper evaluates the influence of different affinity metrics applied to the LPP feature extraction algorithm. The experiments show that the affinity metrics really influence the performance of the LPP.

The affinity metrics analyzed here were: cosine, which does not have free parameter; euclidean neighbor and heat kernel, that have only one decimal free parameter; local scaling and nearest neighbor, that have only one natural free parameter.

In order to adjust the free parameters of the euclidean neighbor and the heat kernel metrics, a GA was used. The tuning of the natural free parameters was performed based on a set of predetermined values, which represent the number of nearest neighbors of a pattern.

Among the evaluated affinity matrices, the most stable metrics were local scaling and nearest neighbor. These two metrics were statistically equivalent, so it is not possible to point out the best one. However, we can indicate one advantage for each one. In terms of computational time, the local scaling metric was, approximately, 1.5 times slower than the time spent by the nearest neighbor metric. Analyzing the number of dimension, the local scaling metric was better. Once it required less dimensions than the nearest neighbor metric. Besides, the local scaling is more stable to the free parameters variation than the nearest neighbor. Thus, these two metrics are indicated for general applications. If computational cost is the parameter to be optimized, nearest neighbor is the correct choice. On the other hand, local scaling requires less storage space.

References

[1] R. O. Duda, P. E. Hart and D. G. Stork "Pattern classification," in (Chapter 4), Edited by John Wiley and Sons Inc, New York, NY: Wiley-Interscience, 2001.

[2] P. He and P. Niyogi "Locality preserving projections," *Advances in Neural Information Processing Systems*, pp. 585–591, 2003.

[3] M. Sugiyama, "Dimensionality reduction of multimodal labeled data by local fisher discriminant analysis," *Journal of Machine Learning Research*, vol. 8, pp. 1027–1061, 2007.

[4] F. R. K. Chung, *Spectral Graph Theory*, (CBMS Regional Conference Series in Mathematics, 92), American Mathematical Society, 1997.

[5] Jun-Bao Li, Jeng-Shyang Pan and Shu-Chuan Chu, "Kernel class-wise locality preserving projection," *Information Sciences*, vol. 178, pp. 1825–1835, 2008.

[6] A. Asuncion and D. J. Newman, "UCI Machine Learning Repository," in http://www.ics.uci.edu/∼mlearn/MLRepository.html, January, 2010.

[7] "Benchmark Repository of the Intelligent Data Analysis Group," in http://ida.first.fhg.de/projects/bench/benchmarks.htm, January, 2010.

[8] A. E. Eiben and J. E. Smith "Introduction to evolutionary computing," in (Chapter 3), Edited by Springer, 2003.