From Indefinite to Positive Semi-Definite Matrices

Alberto Muñoz[1] and Isaac Martín de Diego[2]

[1] University Carlos III de Madrid, c/ Madrid 126, 28903 Getafe, Spain
alberto.munoz@uc3m.es
[2] University Rey Juan Carlos, c/ Tulipán s/n, 28933 Móstoles, Spain
isaac.martin@urjc.es

Abstract. Similarity based classification methods use positive semi-definite (PSD) similarity matrices. When several data representations (or metrics) are available, they should be combined to build a single similarity matrix. Often the resulting combination is an indefinite matrix and can not be used to train the classifier. In this paper we introduce new methods to build a PSD matrix from an indefinite matrix. The obtained matrices are used as input kernels to train Support Vector Machines (SVMs) for classification tasks. Experimental results on artificial and real data sets are reported.

1 Introduction

Classification methods generally rely on the use of a (symmetric) similarity matrix. In many situations it is convenient to consider more than one similarity measure. For instance, in Web Mining problems we have an asymmetric link matrix among Web pages, A. A_{ij} is 1 when there is a link between page i and page j and it is 0 when there is not a link. Two different matrices are defined from A: the co-citations (A^TA) and co-references (AA^T) matrices. Another matrix is defined from the terms by documents (or web pages) matrix, D. $D_{ij} = 1$ if term i appears in web page j and it is 0 when it does not appear. The 'document by document' matrix is defined by D^TD. The co-citations, co-references and 'document by document' matrices correspond to different similarity representations focusing on different data aspects. Several methods have been proposed to combine similarity matrices [11,8,10] in order to create a new single representation for which a classifier is trained. If the similarity representations are not equivalent, a better classification performance should be expected if we are able to combine them. Often, the resulting combination matrix is not positive semi-definite (PSD), that is, it has one or more positive eigenvalues and one or more negative eigenvalues. Then, it is not possible to embed the data into a Euclidean space. The combination matrix is not appropriate to train most used classifiers, and thus it must be modified.

In this paper we afford a deep review of the existing techniques to obtain a PSD matrix from an indefinite one, and propose new methods specially useful for classification tasks. The process of obtaining a PSD matrix from an indefinite

D.-Y. Yeung et al. (Eds.): SSPR&SPR 2006, LNCS 4109, pp. 764–772, 2006.

matrix will be called *Euclideanization* in the following. We will use the resulting matrix as kernel to train a Support Vector Machine (SVM) classifier.

The rest of the paper is organized as follows. In Section 2, we review the existing Euclideanization methods. In Section 3 we propose several Euclideanization methods, adapting them to the classification context. The experimental setup and results on artificial and real classification problems are described in Section 4. Section 5 concludes.

2 Classical Methods

Let K be a real $n \times n$ symmetric indefinite matrix. By the spectral decomposition theorem K can be written as $K = U_n \Lambda_n U_n^T = \sum_{i=1}^n \lambda_i u_i u_i^T$, where Λ_n is a diagonal matrix of eigenvalues of K (first, p positive eigenvalues with decreasing values, next q negative ones with decreasing magnitude, and finally, zero values), and U_n is an orthogonal matrix whose columns u_i are standarized eigenvectors.

2.1 Multidimensional Scaling

The first Euclideanization method considers the matrix $Z = U_r \Lambda_r^{\frac{1}{2}}$, where $r \leq p$ [2]. The new matrix is defined as follows:

$$K^*_{MDS} = ZZ^T = U_r \Lambda_r U_r^T \, . \tag{1}$$

This is equivalent to consider only those eigenvalues larger than a positive constant ϵ, (if $\epsilon = 0$, then $r = p$). In the case of indefinite matrices, the magnitudes of negative eigenvalues suggest the deviation from Euclideaness [13]:

$$r_{mm} = 100 \frac{|\lambda_{min}|}{\lambda_{max}}, \quad r_{neg} = 100 \frac{\sum_{\lambda_i < 0} |\lambda_i|}{\sum_{i=1}^n |\lambda_i|} \, . \tag{2}$$

Now, consider a classification problem involving a sample $x_1, \ldots, x_n$ and an indefinite kernel matrix K, where $K_{ij} = K(x_i, x_j)$. In order to use the kernel matrix K^*_{MDS} with an SVM classifier, we should be able to calculate $K^*(x, x_i)$ for a new point x, given that the SVM classifier takes the form $f(x) = b + \sum_i \alpha_i K^*_{MDS}(x, x_i)$. Let $K(x, \cdot) \in \mathbb{R}^{1 \times n}$ be the vector of original kernel values for the new point, then:

$$K^*_{MDS}(x, \cdot) = K(x, \cdot) U_r \Lambda_r^{-\frac{1}{2}} \Lambda_r^{\frac{1}{2}} U_r^T = K(x, \cdot) U_r U_r^T \, . \tag{3}$$

2.2 Pseudo-euclidean Space

An alternative solution to MDS is to use both positive and negative eigenvalues of K to represent the data set in a Pseudo-Euclidean space [3], $Z = U_k |\Lambda_k|^{\frac{1}{2}}$, where $k = p + q$. The new matrix is defined as follows:

$$K^*_{Pseudo} = ZZ^T = U_k |\Lambda_k| U_k^T \, . \tag{4}$$

In this case, the kernel expression for new points is given by [12]:

$$K^*_{Pseudo}(x,\cdot) = K(x,\cdot)U_k|\Lambda_k|^{-\frac{1}{2}}M|\Lambda_k|^{\frac{1}{2}}U_k^T = K(x,\cdot)U_kMU_k^T \,, \tag{5}$$

where $M = \begin{bmatrix} I_p & 0 \\ 0 & -I_q \end{bmatrix}$. An alternative method is to consider a positive constant ϵ and only those eigenvalues such that $|\lambda_i| \geq \epsilon$.

2.3 Adding a Quantity to the Diagonal of the Matrix

In this method, a positive constant λ is added to the diagonal of the original matrix, large enough to make positive all the eigenvalues of the kernel matrix ($\lambda > |\lambda_{min}|$ will do):

$$K^*_{Add} = K + \lambda I = U(\Lambda + \lambda I)U^T \,. \tag{6}$$

3 Alternative Methods

3.1 Square Transformation

First, we propose a very intuitive, computationally cheap and free parameter method to build a kernel matrix from a symmetric indefinite matrix K as follows:

$$K^*_{ST} = K^2 = KK = U\Lambda U^T U\Lambda U^T = U\Lambda^2 U^T \,. \tag{7}$$

For the new points the kernel values can be calculated by: $K^*_{ST}(x,\cdot) = K(x,\cdot)K$.

3.2 Bending

Hayes and Hill propose in [4] a method termed ‘bending’ for the modification of estimates of covariance matrices in the construction of genetic selection indices. Bending is an iterative process of updating a matrix when a weighting matrix is given to control the relative importance of the elements of the original matrix [6]. Let K be a symmetric indefinite matrix and let W be a weighting matrix for the elements of K. The Bending process is resumed in Algorithm 1.

```
Let $n = 0$, $K_0 = K$, $\varepsilon$ a positive constant and $\odot$ denotes the Hadamard product.
while $K_n$ is indefinite do
   Calculate the decomposition: $K_n = U_n \Lambda_n U_n^T$.
   Replace $\Lambda_n$ with $\Lambda_n^*$, where $\lambda_i^* = g(\lambda_i, \varepsilon)$.
   Calculate a new matrix: $K_{n+1} = K_n - \left[K_n - U_n \Lambda_n^* U_n^T\right] \odot W$.
   $n = n + 1$.
end while
```

Algorithm 1. Bending

In the original Bending algorithm, $g(\lambda_i, \varepsilon) = max(\lambda_i, \varepsilon)$. Alternatively, we propose to use $g(\lambda_i, \varepsilon) = max(|\lambda_i|, \varepsilon)$ as in Pseudo method, $g(\lambda_i, \varepsilon) = \lambda_i + \lambda$ as in the method of adding a constant to the eigenvalues, or $g(\lambda_i, \varepsilon) = \lambda_i^2$ as in the Square Transformation (ST) method. If the weighting matrix W is such that all its elements are equal, then the Bending method is equivalent to the MDS method. If $W_{ij} = 0$ then the value in K_{ij} does not change at this step. We propose to calculate $K^*(x, \cdot)$ for a new point x in a similar way to the MDS method. Then, to modify $K^*(x, \cdot)$, a weighting matrix for the new points should be known in advance:

$$\begin{aligned} K_{n+1}(x,\cdot) &= K_n(x,\cdot) - (K_n(x,\cdot) - K^*_{MDS}(x,\cdot)) \odot W(x,\cdot) \\ &= K_n(x,\cdot) - \left(K_n(x,\cdot) - K_n(x,\cdot)UU^T\right) \odot W(x,\cdot) \\ &= K_n(x,\cdot) - K_n(x,\cdot)\left(I - UU^T\right) \odot W(x,\cdot)\,. \end{aligned} \tag{8}$$

The Bending method can be used when we are dealing with a distance matrix but the matrix under consideration is an indefinite matrix, to guarantee that the diagonal elements of the PSD final matrix become 0.

3.3 Alternating Projections

Alternating Projections [14] is a theoretically powerful method for computing best approximations from a closed convex set K that is the intersection of a finite number of closed convex sets, $K = \cap_{m=1}^{M} K_m$. We will use this method to find the nearest matrix at the intersection of the sets of PSD matrices and the matrices which diagonal elements are fixed to a given value. This method works as an iterative algorithm that reduces the problem to find best approximations from the individual sets.

Consider the following problem:

$$\begin{array}{ll} min_A & \|K - A\|_W \\ s.t. & A = A^T\,, \\ & A \succeq 0\,, \\ & diag(A) = c\,, \end{array} \tag{9}$$

where $\|X\|_W = \|W^{1/2} X W^{1/2}\|_F$, $\|\cdot\|_F$ denotes the Frobenius norm, W is a symmetric PSD matrix, and c is a vector in $\mathrm{I\!R}^n$. This problem appears in the finance industry when given a symmetric matrix K (for example correlations between stocks), the nearest symmetric PSD matrix K^* with unit diagonal (the nearest correlation matrix) is required.

The solution to (9) is a matrix in the intersection of the set of symmetric PSD matrices (S) and the set of symmetric matrices with diagonal equals to the vector c (U), that is closest to K using a weighted Frobenius norm. Since S and U are both closed convex sets, it can be shown that the minimun in (9) is achieved and the solution is unique [7].

Let P_S and P_U be the projections onto S and U respectively. To find the nearest matrix in the intersection of the sets S and U we can iteratively project by repeating the operation:

$$A \leftarrow P_U(P_S(A)) . \tag{10}$$

It can be shown [5] that:

$$P_S(A) = W^{-1/2}\left((W^{1/2}AW^{1/2})^*_{MDS}\right)W^{-1/2} . \tag{11}$$

In practice, we suggest to use a diagonal matrix W. Then, it it easy to show that:

$$P_U(A) = A - (diag(A) - c) . \tag{12}$$

For a new point x, a matrix of weights $W(x,\cdot)$ is needed. We propose to calculate $P_S(K(x,\cdot))$ as in the MDS method in (11), and to obtain $P_U(P_S(K(x,\cdot)))$, only the values of c for the new point are needed.

Next, we present two new Euclideanization methods. In the first one, a kernel matrix is modified to be as similar as possible to the indefinite matrix, without losing the PSD property. In the second method, a linear combination of kernels as similar as possible to the original indefinite matrix is built.

3.4 Conformal Transformation

Let A be a given PSD matrix similar to an indefinite matrix K. In our context A could be the average of several kernel matrices (see Section 4 for details). Let W be a diagonal matrix in $\mathbb{R}^{n\times n}$. Consider the problem:

$$min_W \; \|K - WAW\|^2_F . \tag{13}$$

Note that if A is PSD, so is WAW. Given a PSD matrix A and an indefinite matrix K, we look for a Conformal Transformation (CT) of A such that the resulting matrix K^* is the closest to the input matrix K.

We propose an iterative method to solve problem (13). W is initialized as the identity matrix of order n. The elements of W are modified iteratively (adding or subtracting a fix constant), while a better approximation between matrices WAW and K (a lower Frobenius norm value) is being obtained. The w value for a new point x is:

$$w_x = \frac{\sum_{i=1}^n w_i K(i,x)A(i,x)}{\sum_{i=1}^n (w_i A(i,x))^2} , \tag{14}$$

where $w = diag(W) \in \mathbb{R}^{n\times 1}$.

Instead of a diagonal matrix, a more complicated expression for W could be used in (13). Note that $WAW = A \odot w * w^T$. Instead of using a matrix defined by a single column w, we extend our method by considering a matrix $V \in \mathbb{R}^{n\times r}$ of r columns of weights. The expression for the new matrix is $A \odot V * V^T$.

3.5 Conformal Linear Combination

Let K an indefinite matrix and let $K_1, \ldots, K_M$ a set of M PSD matrices (kernels). Consider the problem of finding the PSD linear combination of those matrices, closest to K:

$$\begin{array}{ll} min_{\lambda_m} & \|K - \sum_{m=1}^{M} \lambda_m K_m\|_F^2 \\ s.t. & \sum_{m=1}^{M} \lambda_m = 1\,, \\ & \lambda_m \geq 0 \quad \forall m = 1, \ldots, M\,. \end{array} \tag{15}$$

It is easy to show that this problem is equivalent to a simple quadratic programming problem. We will label this method as 'conformal linear combination' (CLC). In the particular case of $M = 2$ kernels, the solution is:

$$\lambda_1 = \frac{\langle K_1 - K_2, K - K_2 \rangle}{\|K_1 - K_2\|_F^2}, \quad \lambda_2 = 1 - \lambda_1\,. \tag{16}$$

4 Experiments

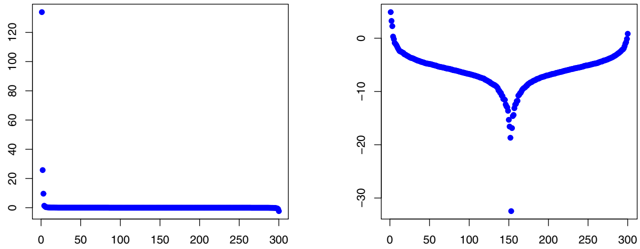
To test the performance of the proposed methods, SVMs have been trained on artificial and real data sets using the kernel matrices K^* previously constructed. To evaluate the accuracy of the classifiers, the classification error, the sensitivity: (True '+' recovered/Total true '+') and the specificity: (True '-' recovered)/(Total true '-') measures are used. In all cases, the results have been averaged over 10 runs.

4.1 Artificial Data Set

This data set consists of 400 two-dimensional points (200 per class). Each group corresponds to a normal cloud with mean vector μ_i and diagonal covariance matrix $\sigma_i^2 I$. Here $\mu_1 = (3, 3)$, $\mu_2 = (5, 5)$, $\sigma_1 = 0.7$ and $\sigma_2 = -0.9$. We have defined two kernels from the projections of the data set onto the coordinate axes. We have used 75% of the data for training and 25% for testing. The interest of this example lies in the fact that, separately, both kernels achieve a poor result (a test error higher than 15%).

We have used the *pick-out* method [11] to combine the two kernels involved. For a pair of elements in the sample, the pick-out method chooses the maximun of the kernels involved if the two elements belong to the same class and the minimun of the kernels under consideration if the two elements belong to different classes. The output matrix obtained is not necessarily PSD. The eigenvalues of the output matrix for the artificial data set are represented on Figure 1. Although the first three eigenvalues are clearly higher than the rest, half of the eigenvalues are negative. The deviation from Euclideaness can be measured using (2): $r_{mm} = 1.6 \pm 0.3$ and $r_{neg} = 2.8 \pm 0.6$ ($mean \pm s.d.$), which suggests a moderate deviation from Euclideaness.

Table 1 shows the classification results. The MDS and Pseudo subscripts represent the value of the positive constant ϵ. The MDS, Bending and AP methods achieve the lowest test error. The support vectors obtained with the MDS method were used to define the weighting matrices needed in the Bending and AP methods. The MDS and Pseudo classification results strongly depend on the value of the parameter ϵ. The best results were achieved using $\epsilon = 5$, which implies the



(a) *Eigenvalues* (b) *Log*|*Eigenvalues*|

Fig. 1. Eigenvalues of the pick-out output matrix for the artificial data set

Table 1. Percentage of misclassified data, sensitivity (Sens.), specificity (Spec.) and percentage of support vectors (S.V.) for the kernels with complementary information

Method	Train	Test	Sens.	Spec.	S.V.
MDS$_0$	3.0	10.0	0.956	0.845	19.4
MDS$_1$	3.2	7.1	0.952	0.911	19.7
MDS$_5$	4.4	4.3	0.935	0.978	21.5
Pseudo$_0$	5.3	10.8	0.966	0.821	19.4
Pseudo$_1$	3.6	7.5	0.939	0.913	18.8
Adding λI	5.2	5.2	0.899	0.996	60.1
ST	5.2	7.0	0.951	0.911	3.9
Bending	4.4	4.3	0.935	0.978	21.5
AP	4.4	4.3	0.935	0.978	21.5
AKM	6.4	6.5	0.868	1.000	35.4
CT$_{AKM}$	6.3	6.1	0.876	1.000	35.1
CLC$_{AKM}$	6.4	6.7	0.864	1.000	35.9

selection of the two highest eigenvalues. The ST method involves significantly less support vectors than the other methods. On the other hand, adding a quantity to the diagonal of the eigenvalue matrix increases the percentage of support vectors. The conformal transformation method outperforms the average of the kernels method (AKM [10]) when the AKM method was used to initialize the transformation. The starting matrices of the CLC method are the two original kernels. The classification results were similar to that obtained from the AKM. Both kernels, individually, achieve poor classification results, and thus, given the definition of the kernel, it is not possible to define a linear combination of the kernels able to significantly improve the AKM results.

4.2 A Real Data Set Classification Problem

In this section we have dealt with a database from the UCI Machine Learning Repository: the Johns Hopkins University Ionosphere database [1]. The data

Table 2. Percentage of misclassified data, sensitivity (Sens.), specificity (Spec.) and percentage of support vectors (S.V.) for the ionosphere data set

Method	Train	Test	Sens.	Spec.	S.V.
MDS$_0$	1.9	6.4	0.973	0.874	43.0
MDS$_1$	3.3	6.5	0.969	0.878	34.4
MDS$_5$	5.4	7.7	0.964	0.851	34.1
Pseudo$_0$	1.9	6.7	0.952	0.901	44.7
Pseudo$_1$	3.3	6.5	0.969	0.878	34.4
Adding λI	1.6	6.4	0.983	0.855	65.9
ST	2.1	5.9	0.965	0.901	21.5
Bending	2.0	6.0	0.966	0.895	44.5
AP	1.7	5.9	0.977	0.881	43.5
CT$_{RBF}$	4.0	6.0	0.987	0.859	53.0
AKM	2.3	6.7	0.982	0.851	45.0
CLC$_{AKM}$	2.2	5.9	0.980	0.875	45.5

set consists of 351 observations with 34 continous predictor attributes variables each. We have used 60% of the data for training and 40% for testing.

For this data set we have combined several RBF kernels $K_m(x,z)=e^{-||x-z||^2/c_m}$ with $c_m = 10 + 5 * (m - 1)$ and $m = 1, \dots, 10$. We have used a linear kernel $K(x, z) = x^T z$, and a polynomial kernel $K(x, z) = (1 + x^T z)^2$ as well. We have considered the following transformation: $K(x, z) = \frac{K(x,z)}{\sqrt{K(x,x)}\sqrt{K(z,z)}}$ to make comparable the different kernels values. The KWS method [9] (Kernel Weighting Scheme) has been used to combine these kernels. In this method we use the kernel value, the neighbourhood of the elements and the label information to assign different weights to each element into the kernel matrix. The output matrix is not necessarily PSD. The percentage of negative eigenvalues is 19.0%, and their relative importance is significant: $r_{mm} = 3.2 \pm 0.3$ and $r_{neg} = 4.6 \pm 0.4$.

The classification results are shown on Table 2. The ST, AP and CLC methods achieve the best results. Similar results were obtained with the CT method which improves the a priori kernel matrix used (RBF with parameter $c = 55$: 7.0% in test error). The CLC method clearly outperforms the AKM method. The performance of MDS and Pseudo methods is related to the choice of the parameter ϵ. The support vectors obtained with the MDS method with $\epsilon = 0$ were used to define the weighting matrices needed in the Bending method. When the diagonal elements of the output matrix were fix to be 1, the AP method outperforms the MDS method.

5 Conclusions

In this paper, we propose new techniques to build a PSD matrix from an indefinite one. The obtained PSD matrix is used as input kernel to train a SVM classifier. The classification results strongly depend on the method used to build the kernel. The Square Transformation method implies the lowest number of support vectors. The Alternating Projections and Bending methods have been shown to be good alternatives to the classical techniques. The Conformal Transformation method clearly improves the results obtained from an a priori kernel. The Conformal Linear Combination method has been shown to be an alternative to the average of the kernels method.

Acknowledgments

This work was partially supported by Spanish grant SEJ2004-03303.

References

1. C.L. Blake and C.J. Merz, C.J. UCI *repository of Machine Learning databases. University of Carolina, Irvine, Department of Information and Computer Sciences. http://www.ics.uci.edu/∼mlearn/MLRepository.html,* 1998.
2. T.F. Cox and M.A.A. Cox. *Multidimensional Scaling.* Chapman-Hall, 1994.
3. L. Goldfarb. *A new approach to Pattern Recognition.* Progress in Pattern Recognition, 2, 241-402, 1985.
4. J.F. Hayes and W.G. Hill. *Modification of estimates of parameters in the construction of genetic selection indices ("Bending").* Biometrics, 37 (1981) 483-493.
5. N. Highman. *Computing the nearest correlation matrix- a problem from finance.* IMA Journal of Numerical Analysis, 22 (2002) 329-343.
6. H. Jorjani, L. Klei and U. Emanuelson. *A simple method fot weighted bending of genetic (co)variance matrices.* J. Dairy Sci, 86 (2003) 677-679.
7. D.G. Luenberger. *Optimization by Vector Space Methods.* New York: Wiley, 1969.
8. I. Martín de Diego, J.M. Moguerza and A. Muñoz. *Combining Kernel Information for Support Vector Classificacion.* Proc. MCS (2004), LNCS 3077, Springer, 102-111.
9. I. Martín de Diego, A. Muñoz, and J.M. Moguerza, *Methods for the Combination of Kernel Matrices within a Support Vector Framework.* Submitted.
10. J.M. Moguerza, A. Muñoz and I. Martín de Diego. *Improving Support Vector Classificacion via the Combination of Multiple Sources of Information.* Proc. SSPR and SPR (2004), LNCS 3138, Springer, 592-600.
11. A. Muñoz, I. Martín de Diego and J.M. Moguerza. *Support Vector Machine Classifiers for Assymetric Proximities.* Proc. ICANN (2003), LNCS 2714, Springer, 217-224.
12. E. Pekalska, P. Paclík and R.P.W. Duin. *A Generalized Kernel Approach to Dissimilarity-based Classification.* JMLR, Special Issue on Kernel Methods 2 (2) (2002) 175-211.
13. E. Pekalska, R.P.W. Duin, S. Günter and H. Bunke. *On Not Making Dissimilarities Euclidean.* Proc. SSPR and SPR (2004), LNCS 3138, Springer, 1145-1154.
14. J. von Neumann. *The Geometry of Orthogonal Spaces.* Functional operators-vol.II. Annals of Math. Studies, no. 22. Princeton University Press. 1950.