Available online at www.sciencedirect.com
SCIENCE DIRECT®

Pattern Recognition 38 (2005) 2653–2657

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Rapid and brief communication

Learning multi-kernel distance functions using relative comparisons

Eng-Jon Ong*, Richard Bowden

Centre for Vision, Speech and Signal Processing, SEPS, University of Surrey, Guildford GU2 7XH, UK

Received 12 May 2005; accepted 16 May 2005

Abstract

In this manuscript, a new form of distance function that can model spaces where a Mahalanobis distance cannot be assumed is proposed. Two novel learning algorithms are proposed to allow that distance function to be learnt, assuming only relative-comparisons training examples. This allows a distance function to be learnt in non-linear, discontinuous spaces, avoiding the need for labelled or quantitative information. The first algorithm builds a set of basic distance bases. The second algorithm improves generalisation capability by merging different distance bases together. It is shown how the learning algorithms produce a distance function for clustering multiple disjoint clusters belonging to the same class. Crucially, this is achieved despite the lack of any explicit form of class labelling on the training data.

Keywords: Distance function learning; Multi-kernel basis; Basis grouping

1. Introduction

Distance functions are an important component in many learning applications. Clustering or classification algorithms typically rely on some form of a distance function that has been a priori defined within an input space. Additionally, it is often the case that the correct function is context dependent; in fact, it is often not possible to choose a specific distance function. It would therefore be advantageous to learn a distance function using available training data. Many approaches to learning distance functions take the form of a weighted Euclidean function. A popular approach is the Mahalanobis function [1]. It was found that the use of the Mahalanobis function can sometime cause problems in cases where a discontinuous input space is present (e.g., XOR problem). This paper proposes an alternative to the Mahalanobis distance based upon a combination of multi-kernel distance bases.

This work only assumes the availability of a set of relative comparisons as in Ref. [2], where given three variables, A, B and C, A is closer to B than C. This avoids the need for having labelled or quantitative information. To learn the distance function, two algorithms are proposed. The first algorithm builds a set of basic distance bases. To improve the generalisation capability of the final distance function, a second algorithm is proposed to merge different distance bases together. It is shown how the learning algorithms produce a distance function that can correctly cluster multiple disjoint clusters belonging to the same class together.

2. Multi-kernel distance function

At the heart of the distance function is a collection of kernels. Given a new input vector x, the kernel is

* Corresponding author. Tel.: +44 1483 689842; fax: +44 1483 686031.
E-mail address: e.ong@surrey.ac.uk (E.-J. Ong).

0031-3203/$30.00 © 2005 Pattern Recognition Society. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2005.05.011

defined as

$$K(x, c) = |(x - c)^2|, \tag{1}$$

where c is the centre of the kernel. This provides a Euclidean distance from the centre of the kernel to the point x.

These kernels are grouped together into N_B number of distance bases. Each distance basis is associated with a set of kernels: $C_j = \{c_{ji}\}_{i=1}^{b_j}$, where b_j is the number of kernels and c_{ji} is the ith kernel centre for the jth basis. The set of kernels in a distance basis can now be used to provide a measure of "nearest distance" as follows:

$$B(x, y, C_j) = \min(K(x, c_{ji})) + \min(K(y, c_{ji})), \quad i = 1, \ldots, b_j. \tag{2}$$

All the distance bases kernels can be grouped into $C = \{C_j\}$, $j = 1, \ldots, N_B$. The distance function between two points (x and y) can then be defined as

$$D(x, y, C) = \min(B(x, y, C_j)), \quad j = 1, \ldots, N_B. \tag{3}$$

3. Distance function learning using relative examples

This section shows how the kernel centres are chosen and subsequently how the distance bases are formed using an algorithm inspired by the Boosting method [3] of learning. The learning framework consists of two major steps. The first step learns primitive distance bases, each containing only two kernels. It was found that this is enough to give a very low training error. However, it does not provide good generalisation capabilities. To address this, a second step, whereby these simple distance bases are merged together to form a more general distance function, is introduced.

Before the algorithm is described, a few definitions are provided. A training dataset consisting of N_T relative comparison triplets is defined by $T_j = \{t_{ji}\}_{i=1}^{3}$, $j = 1, \ldots, N_T$. These training triplets are created in such a way that t_{j1} is closer to t_{j2} than t_{j3}. The entire training dataset is defined as $T = \{T_j\}_{j=1}^{N_T}$. Additionally, each training example is associated with a weight $W = \{w_j\}_{j=1}^{N_T}$. Given a set of distance function bases C, the training examples T and their weights W, the training error function $E(T, W, C)$ is defined as follows:

$$E(T, W, C) = \sum_{j=1}^{N_T} w_j G(T_j, C), \tag{4}$$

$$G(T_j, C) = \begin{cases} 1 & (D(t_{j1}, t_{j2}, C) > D(t_{j1}, t_{j3}, C)), \\ 0 & (D(t_{j1}, t_{j2}, C) < D(t_{j1}, t_{j3}, C)), \end{cases} \tag{5}$$

where $G(T_j, C)$ is the individual error function for the jth triplet given a set of distance bases C. To obtain the set of potential primitive distance bases, the first two examples of each training triplet are used: $K_k = \{t_{k1}, t_{k2}\}$. The entire set of primitive distance bases are denoted $K = \{K_k\}_{k=1}^{N_T}$.

3.1. Learning and merging primitive distance bases

The algorithm for learning the primitive distance bases primarily revolves around a distance-basis selection loop. Within this loop, a new primitive distance basis that provides the smallest training error and is then added into the existing set is chosen. The loop terminates when the training error falls below a threshold t. An example of the result of the algorithm applied to an extended XOR dataset (see Section 4 for details) can be seen in Fig. 1(a). The primitive distance bases are shown as lines linking two kernel centres. The algorithm will result in a set of N_B distance bases C as follows:

1: Initialisation Step
   (i) $N_B = 0$, $M = 1$, $w_j = 1$, $j = 1 \ldots N_T$
   (ii) $C_0 = \{\}$ {No distance bases found yet}
2: **while** $\sum_{j=1}^{N_T} w_j > t$ **do**
3: $K_{best} = \arg\min_{K_{best} \in K} E(T, W, \{C_{M-1}, K_{best}\})$ {Find least training error distance-basis}
4: $C_M = \{C_{M-1}, K_{best}\}$
5: $w_j = G(T_j, C_M)$, $j = 1 \ldots N_T$ { Update the weights }
6: $M = M + 1$
7: **end while**
8: $N_B = M$, $C = C_M$, break

To obtain a distance function with better generalisation capabilities, it is necessary to merge the above primitive distance bases together. This is achieved by merging two distance bases together if the result of this action does not cause the existing training error to increase. An example of the merging algorithm applied to the extended XOR problem is shown in Figs. 1(b)–(e). Each figure shows a group of kernels (red squares), the result of merging various primitive distance bases into a single larger distance basis.

4. Experiments

The first experiment shows an extended XOR problem. Six boxes are defined and split into two classes arranged in a checkerboard layout (see Fig. 2(a)). Training data triplets are produced by randomly selecting two points from the same class and the third from the other class. A distance function is learnt using the training data. A test dataset is also generated by randomly selecting points from the six boxes. Fig. 3 shows the result of the learnt distance function. A random point from each class is chosen from the test dataset (black box in Figs. 3(b) and (e)). Figs. 3(a) and (c) show the distances of all other test points to the respective chosen points. The distances for the two classes are plotted independently. The distances of most of the points belonging to the class

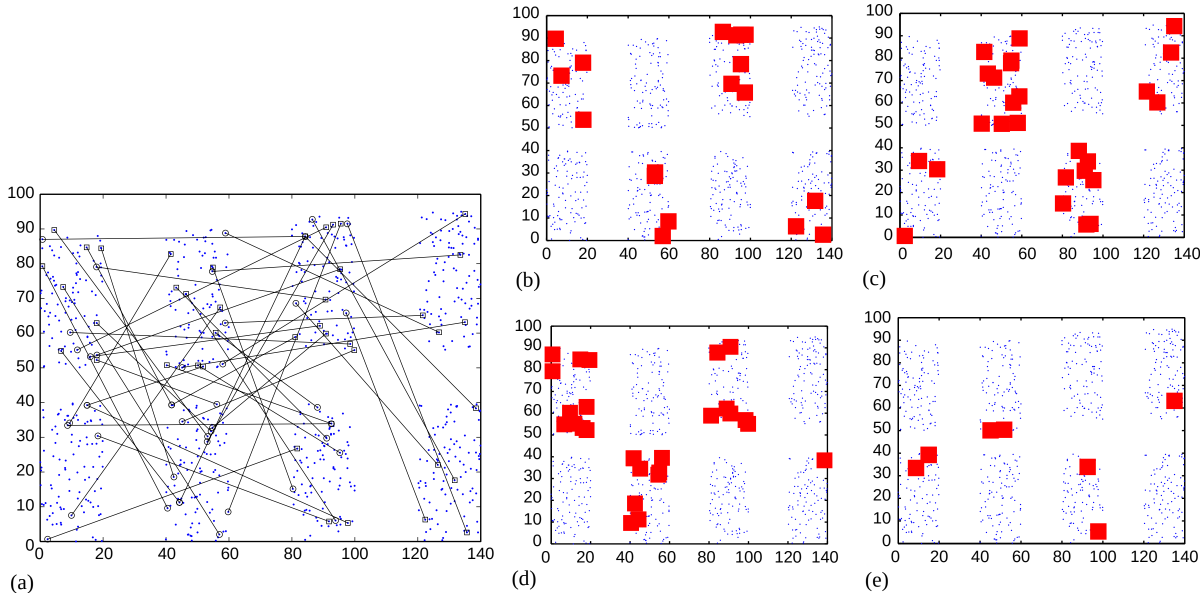

Fig. 1. Example of the learnt distance metric kernels before merging (a) and four different distance bases' kernels after merging (b)–(e).

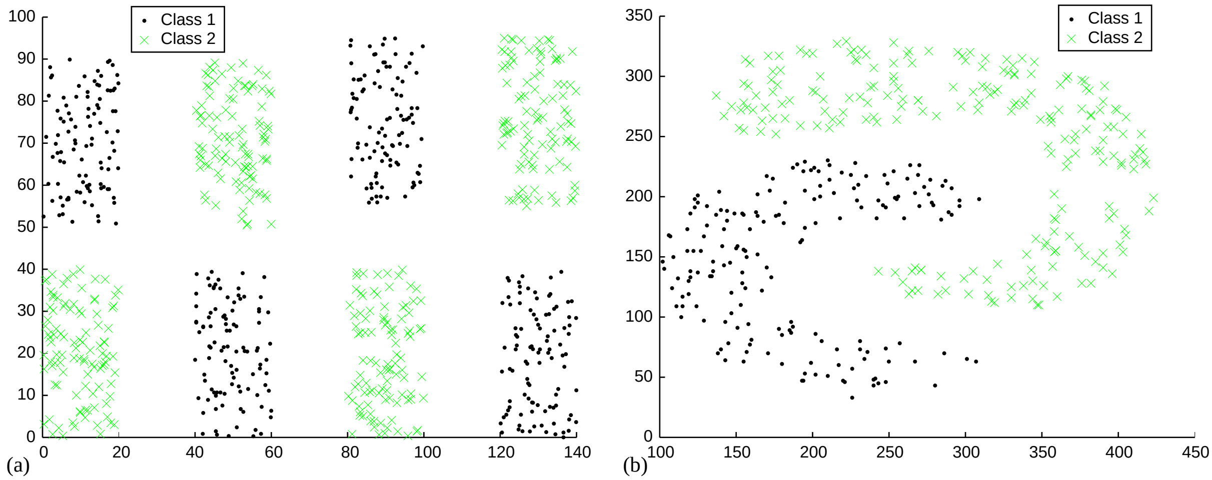

Fig. 2. Groundtruth for the two experiments.

of the selected point are smaller than the points of the other class. The points which are "close" to the selected point can be seen as filled red circles in Figs. 3(b) and (e). Additionally, a distorted space using the distance measures is shown in Figs. 3(c) and (f). Here, the selected point is made the origin, all other points are projected onto a unit circle around the origin and scaled using their respective distances. The circle defines the isocontour boundary from the origin. The second experiment involves two non-linear clusters shown in Fig. 2(b). Results equivalent to the first experiment's are shown in Fig. 4.

5. Conclusions

In this manuscript, a new form of distance function that can model spaces where the Mahalanobis distance cannot be assumed was proposed and, indeed in some cases, applied. Two novel learning algorithms were proposed to allow this distance function to be learnt. Training examples are provided in the form of relative comparisons. This allows a distance function to be learnt in non-linear, discontinuous spaces, avoiding the need for labelled or quantitative information. The first algorithm builds a set of basic distance

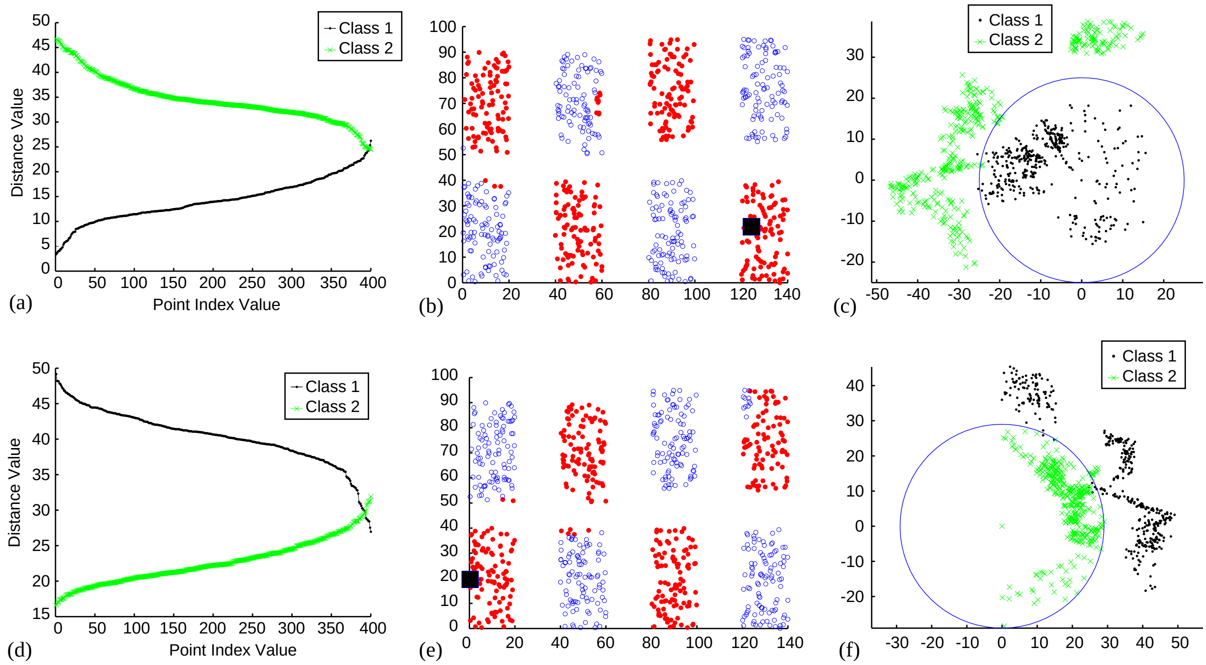

Fig. 3. Results for the extended XOR experiments.

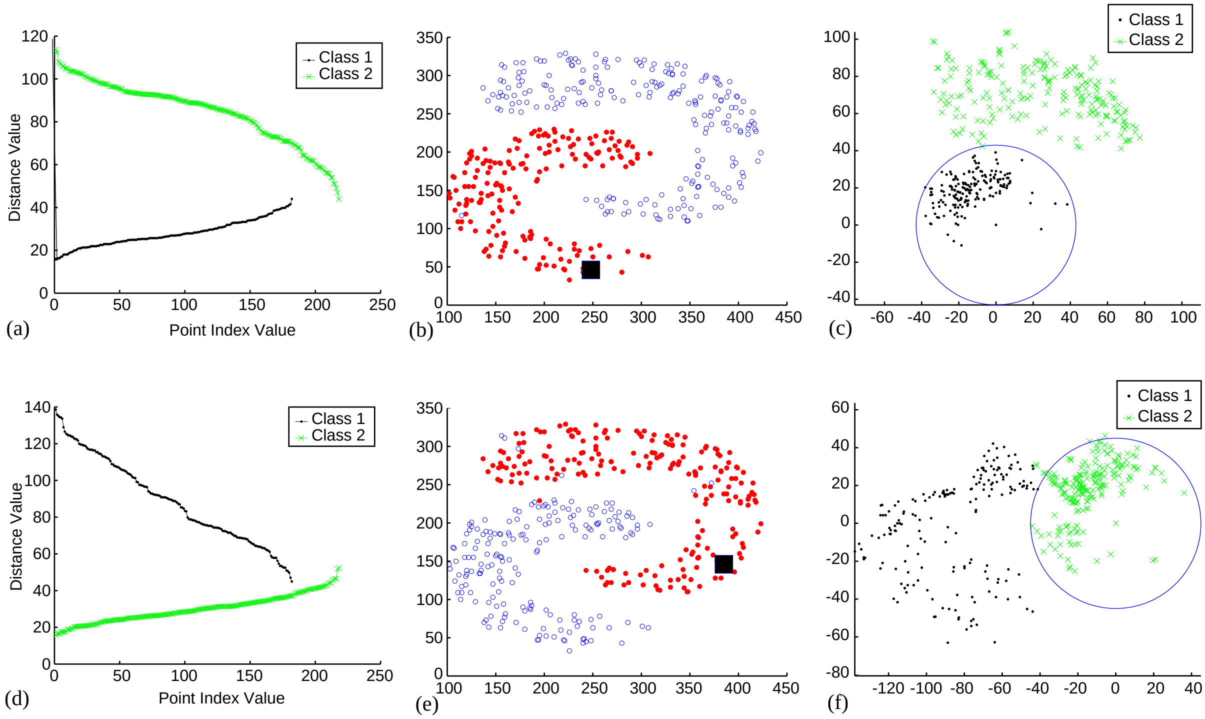

Fig. 4. Two clusters results.

bases. To improve the generalisation capability of the final distance function, a second algorithm merges different distance bases together. The method was evaluated on two cases that test how well it copes with sudden discontinuities and non-linearity in the input space. The learnt distance function was successful in segregating the data into their respective classes. Importantly, this was achieved despite the lack of any explicit form of class labelling on the training data.

Acknowledgements

The investigation reported in this contribution has been partially supported by the European Union (FP6-project 'COSPAL', IST-2003-2.3.2.4).

References

[1] I. Tsang, J. Kwok, Distance metric learning with kernels, in: Proceedings of the International Conference on Artificial Neural Networks, 2003.

[2] M. Schultz, T. Joachims, Learning a distance metric from relative comparisons, in: Proceedings of the Conference on Advance in Neural Information Processing Systems (NIPS), 2003.

[3] R. Meir, G. Rätsch, An introduction to boosting and leveraging, Advanced Lectures on Machine Learning, Springer, Berlin, 2003, pp. 119–184.