97. Benchimol S, Fuks A, Jothy S, Beauchemin N, Shirota K, Stanners CP. Carcinoembryonic antigen, a human tumor marker, functions as an intercellular adhesion molecule. Cell. 1989;57:327–34.
98. Drummond F, Putt W, Fox M, Edwards YH. Cloning and chromosome assignment of the human CDX2 gene. Ann Hum Genet. 1997;61:393–400.
99. Bai YQ, Miyake S, Iwai T, Yuasa Y. CDX2, a homeobox transcription factor, upregulates transcription of the p21/WAF1/CIP1 gene. Oncogene. 2003;22:7942–9.
100. Fath KR, Obenauf SD, Burgess DR. Cytoskeletal protein and mRNA accumulation during brush border formation in adult chicken enterocytes. Development. 1990;109:449–59.
101. Friederich E, Vancompernolle K, Louvard D, Vandekerckhove J. Villin function in the organization of the actin cytoskeleton. Correlation of in vivo effects to its biochemical activities in vitro. J Biol Chem. 1999;274:26751–60.
102. Nadji M, Tabei SZ, Castro A, Chu TM, Morales AR. Prostatic origin of tumors. An immunohistochemical study. Am J Clin Pathol. 1980;73:735–9.
103. Nadji M, Tabei SZ, Castro A, Chu TM, Murphy GP, Wang MC, et al. Prostatic-specific antigen: an immunohistologic marker for prostatic neoplasms. Cancer. 1981;48:1229–32.
104. Diamandis EP, Yousef GM, Luo LY, Magklara A, Obiezu CV. The new human kallikrein gene family: implications in carcinogenesis. Trends Endocrinol Metab. 2000;11:54–60.
105. Stone S, Dayananth P, Jiang P, Weaver-Feldhaus JM, Tavtigian SV, Cannon-Albright L, et al. Genomic structure, expression and mutational analysis of the P15 (MTS2) gene. Oncogene. 1995;11:987–91.
106. Glendening JM, Flores JF, Walker GJ, Stone S, Albino AP, Fountain JW. Homozygous loss of the p15INK4B gene (and not the p16INK4 gene) during tumor progression in a sporadic melanoma patient. Cancer Res. 1995;55:5531–5.
107. Stone S, Jiang P, Dayananth P, Tavtigian SV, Katcher H, Parry D, et al. Complex structure and regulation of the P16 (MTS1) locus. Cancer Res. 1995;55:2988–94.
108. Lukas J, Parry D, Aagaard L, Mann DJ, Bartkova J, Strauss M, et al. Retinoblastoma-protein-dependent cell-cycle inhibition by the tumour suppressor p16. Nature. 1995;375:503–6.
109. Clarke EJ, Allan V. Intermediate filaments: vimentin moves in. Curr Biol. 2002;12:R596–8.
110. Kaku T, Ogawa S, Kawano Y, Ohishi Y, Kobayashi H, Hirakawa T, et al. Histological classification of ovarian cancer. Med Electron Microsc. 2003;36:9–17.
111. McCluggage WG. A critical appraisal of the value of immunohistochemistry in diagnosis of uterine neoplasms. Adv Anat Pathol. 2004;11:162–71.
112. Thomas AA, Stephenson AJ, Campbell SC, Jones JS, Hansel DE. Clinicopathologic features and utility of immunohistochemical markers in signet-ring cell adenocarcinoma of the bladder. Hum Pathol. 2009;40:108–16.
113. Del Sordo R, Bellezza G, Colella R, Mameli MG, Sidoni A, Cavaliere A. Primary signet-ring cell carcinoma of the urinary bladder: a clinicopathologic and immunohistochemical study of 5 cases. Appl Immunohistochem Mol Morphol. 2009;17:18–22.

Knowl Inf Syst
DOI 10.1007/s10115-012-0574-x

REGULAR PAPER

Pairwise ranking component analysis

Jean-François Pessiot · Hyeryung Kim · Wataru Fujibuchi

Received: 18 January 2011 / Revised: 21 July 2012 / Accepted: 6 October 2012

Abstract Uncovering the latent structure of the data is an active research topic in data mining. However, in the distance metric learning framework, previous studies have mainly focused on the classification performance. In this work, we consider the distance metric learning problem in the ranking setting, where predicting the order between the data vectors is more important than predicting the class labels. We focus on two problems: improving the ranking prediction accuracy and identifying the latent structure of the data. The core of our model consists of ranking the data using a Mahalanobis distance function. The additional use of non-negativity constraints and an entropy-based cost function allows us to simultaneously minimize the ranking error while identifying useful meta-features. To demonstrate its usefulness for information retrieval applications, we compare the performance of our method with four other methods on four UCI data sets, three text data sets, and four image data sets. Our approach shows good ranking accuracies, especially when few training data are available. We also use our model to extract and interpret the latent structure of the data sets. In addition, our approach is simple to implement and computationally efficient and can be used for data embedding and visualization.

Keywords Ranking · Distance metric learning · Feature clustering · Dimensionality reduction · Information retrieval

1 Introduction

Distance metrics are essential components in many machine learning algorithms [34,42]. In the classification setting, good distance metrics should improve the accuracy of such

J.-F. Pessiot (✉) · W. Fujibuchi
Computational Biology Research Center (CBRC),
Advanced Industrial Science and Technology (AIST), Tokyo 135-0064, Japan
e-mail: jfk.pessiot@aist.go.jp

H. Kim
Graduate School of Biomedical Sciences, Tokyo Medical and Dental University,
Tokyo 113-8510, Japan

Published online: 26 October 2012

algorithms as the nearest neighbor [19,41]. In the unsupervised setting, such algorithms as K-Means or multi-dimensional scaling critically rely on the choice of the distance function. Distance metrics can also be used for visualization and interpretation purposes and, therefore, are also important for understanding data. However, choosing the appropriate distance function manually is a difficult problem that requires prior knowledge. This underscores the need to automatically learn good distance metrics from available data.

A common approach to distance metric learning takes place in the classification setting. The learned distance metric should help improve classification accuracy. Many distance metric learning methods have been proposed for classification, and most of them learn a Mahalanobis distance metric to optimize the classification error directly or indirectly. Among them, linear discriminant analysis (LDA) learns a Mahalanobis distance metric such that the classes are compactly grouped around their centers, while being far from each other [14]. The intuition is that the classification task will become easier if the classes are well separated. More recently, neighborhood components analysis (NCA) directly optimizes the classification error of the nearest neighbor classifier [19]. Other related studies optimize the margin [41] or the class compactness [18], learn the local structure of the data [36], or work in the online learning setting [22,33]. All these methods share a common objective: They learn a Mahalanobis distance metric so that the classification task would become easier.

However, ranking the data vectors is sometimes more important than predicting the labels themselves [9,27]. In the information retrieval (IR) setting, for example, a user submits a query to an IR system. The IR system should then return a list of documents ranked with respect to their relevance to the query [31]. The most relevant documents should appear at the top of the list and the less relevant ones should appear at the bottom of the list. Online search engines, such as Google,[1] are typical examples of this setting. The quality of such systems relies on their ability to correctly rank the web pages rather than their ability to classify them as relevant or not. This problem has attracted considerable attention in the last few years (see, e.g., [1,5,12,17,20]).

We are interested in learning a distance metric for the ranking setting. Our primary goal is to improve the ranking performance in IR systems that return ranked lists of documents[2] to users. Previous works have proposed solutions to this problem [11,32]. However, those works mainly focus on the prediction accuracy and the scalability. Although those criteria are important, there are other desirable properties for a distance metric learning method: interpretability and simplicity. In this paper, we will focus on these two properties. Interpretability is crucial to discover the latent structure of a data set. The latent structure can then be used to visualize, understand, and summarize the data. This is especially useful for large-scale data sets, for which the analysis of each item would be prohibitively time-consuming. Simplicity is also an important property of our methodology, because databases are becoming increasingly common tools to store and query data. We want our distance metric learning method to be easy to understand and to implement by non-specialists.

The rest of the paper is organized as follows. In Sect. 2, we present state-of-the-art methods in the field of distance metric learning. In Sect. 3, we demonstrate our approach to distance metric learning in the ranking setting. We explain our model and the details of the implementation. We also present an extension of our initial model, which can be used for interpretability purpose. In Sect. 4, we evaluate our approach. We explain our experimental

[1] http://www.google.com.

[2] Although we use text documents for illustrative purposes, our approach can be applied to any kind of data represented by vectors (e.g., image data, biological data, etc.).

protocol, evaluate our work against state-of-the-art methods, and analyze the results. We formulate our conclusions and give suggestions to improve our approach in Sect. 5.

2 Related work

Many studies have been published in the distance metric learning setting. In this section, we present state-of-the-art distance metric learning methods, including linear discriminant analysis, neighborhood components analysis, and information-theoretic metric learning. We also present the popular principal component analysis, which will be used as an unsupervised baseline method in Sect. 4.

2.1 Linear discriminant analysis

Linear discriminant analysis (LDA) learns a linear projection $\mathbf{A}$ such that the classes are well separated in the projected space (see, e.g., [14]). Let N be the total number of examples in the data set, $\boldsymbol{\mu} = \sum_{i=1}^{N} \mathbf{x}_i / N$ be the mean of the data set, N_c be the number of examples in class c, and $\boldsymbol{\mu}_c = \sum_{i \in C_c} \mathbf{x}_i / N_c$ be the mean of class c. Maximizing the variance between the classes while minimizing the variance within the classes amounts to maximizing the following quantity:

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}}$$

where $\mathbf{S}_B = \sum_{c=1}^{C} (\boldsymbol{\mu}_c - \boldsymbol{\mu})(\boldsymbol{\mu}_c - \boldsymbol{\mu})^T$ is the between-class variance matrix, and $\mathbf{S}_W = \sum_{c=1}^{C} \sum_{i \in C_c} (\mathbf{x}_i - \boldsymbol{\mu}_c)(\mathbf{x}_i - \boldsymbol{\mu}_c)^T$ is the within-class variance matrix. Ultimately, maximizing $J(\mathbf{w})$ leads to the resolution of the following eigenvalue problem:

$$\mathbf{S}_W{}^{-1} \mathbf{S}_B \mathbf{w} = \lambda \mathbf{w}$$

where the sought projection matrix $\mathbf{A}$ is the concatenation of the different eigenvectors $\mathbf{w}$. However, if the data set is sparse or if there are very few examples, the matrix $\mathbf{S}_W$ may not be invertible. In this case, we regularized LDA by considering $\mathbf{S}_W + \varepsilon \mathbf{I}_d$ instead of $\mathbf{S}_W$, where $\varepsilon > 0$ is a small number and $\mathbf{I}_d$ is the identity matrix. The computational complexity of LDA is dominated by the computation of the inverse matrix and the resolution of the eigenvalue problem, resulting in $O(D^3 + D^2 N)$. The memory complexity of LDA is dominated by the variance matrices of size $(D \times D)$.

In summary, LDA may be difficult to apply when there are few training data or when the number of dimensions is large. LDA can be used to visualize data in a subspace that has two or three dimensions, where class separation is optimal. However, the lack of non-negativity constraints makes model interpretation difficult [25]. Therefore, it cannot be used to uncover the latent structure of the data.

2.2 Neighborhood components analysis

Neighborhood components analysis (NCA) (Goldberger et al 2004) learns a linear projection $\mathbf{A}$ that optimizes the following error:

$$H(\mathbf{A}) = \sum_{i=1}^{N} \sum_{j \in C_i} \frac{\exp(-\|\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_j\|^2)}{\sum_{k \neq i} \exp(-\|\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_k\|^2)}$$

This cost function is interpreted as the sum of probabilities that each example will be correctly classified under a stochastic nearest neighbor scheme. Hence, the projection learned by NCA optimizes a stochastic variant of the classification error. The computational complexity of NCA is $O(N^2K + NDK)$. In [19], experimental evaluation shows that NCA compares favorably to LDA and PCA with respect to the classification accuracy.

In summary, NCA is a distance metric learning method that has been specifically designed for the nearest neighbor classifier in the classification setting. However, the performance of NCA is not clear in the ranking setting. Like LDA, NCA can be used to visualize the data in a low-dimensional feature space and is unable to uncover the latent structure of the data because of difficulty of model interpretation [25].

2.3 Information-theoretic metric learning

In Information-theoretic metric learning (ITML), [11] build upon [24] and cast the distance metric learning problem in the framework of the information theory [10]. Their goal is to identify a distance function d_Q (parameterized by the Mahalanobis matrix $\mathbf{Q}$) that respects a set S of similarity constraints (i.e., $d_Q(\mathbf{x}_i, \mathbf{x}_j) \leq u$ for a small u) and a set E of dissimilarity constraints (i.e., $d_Q(\mathbf{x}_i, \mathbf{x}_j) \geq \ell$ for a large ℓ). In order to avoid overfitting, they also require the Mahalanobis matrix $\mathbf{Q}$ to be close to a given Mahalanobis matrix $\mathbf{Q}_0$. The resulting optimization problem writes:

$$\begin{aligned} \min_{\mathbf{Q}\succeq 0, \xi} \; & D_{\ell d}\left(\mathbf{Q}, \mathbf{Q}_0\right) + \gamma \, D_{\ell d}\left(\operatorname{diag}(\xi), \operatorname{diag}(\xi_0)\right) \\ \text{subject to} \quad & \operatorname{tr}\left(\mathbf{Q}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T\right) \leq \xi_{c(i,j)} \qquad (i,j) \in S \\ & \operatorname{tr}\left(\mathbf{Q}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T\right) \geq \xi_{c(i,j)} \qquad (i,j) \in E \end{aligned}$$

where $D_{\ell d}$ is the LogDet divergence between two positive-definite matrices, $c(i, j)$ denotes the index of the (i, j)th constraint, and $\boldsymbol{xi}$ is a vector of slack variables initialized to ξ_0. The parameter γ controls the trade-off between satisfying the constraints and minimizing $D_{\ell d}\left(\mathbf{Q}, \mathbf{Q}_0\right)$ and has to be set by the user. The authors present an efficient algorithm to solve the previous optimization problem, with a time complexity of $O(CD^2)$, where C is the number of constraints and D the number of dimensions. However, it should be noted that the matrix $\mathbf{Q}$ is of size $(D \times D)$ and can be difficult to fit in memory when D is very large. In addition, the parameter γ has to be set by the user (e.g., using cross-validation techniques), which significantly increases the computational time.

In summary, ITML is a complex and memory-demanding distance metric learning method that can be difficult to use when the number of dimensions is large. Besides, ITML does not explicitly compute a projection matrix and, therefore, cannot be used for either visualization or interpretability purpose.

2.4 Principal component analysis

Principal component analysis (PCA) is a popular dimensionality reduction method [23] that seeks to maximize the variance of the projected data. The computation of PCA involves solving an eigenvalue decomposition of the data covariance matrix. The first eigenvectors (with respect to the corresponding eigenvalues) define the principal components of the data, that is, the directions along which the variance is maximized. The sought projection matrix is the concatenation of the eigenvectors. In contrast with LDA and NCA, PCA is unsupervised, that is, the projection is computed without considering the class labels. The computational complexity of PCA is $O(D^3 + D^2N)$.

In summary, PCA is an unsupervised method and, therefore, may not be appropriate for the analysis of labeled data. Nevertheless, PCA can be used to visualize the data in a low-dimensional subspace defined by the principal components. However, the lack of non-negativity constraints makes model interpretation difficult [25]. Therefore, PCA cannot be used to uncover the latent structure because of the data set. We used the implementation provided by MDP,[3] a freely available library for data processing.

2.5 Other work

Many other studies have been published in the distance metric learning setting. Similarly to NCA, the method in [41] seeks a distance metric optimal for the nearest neighbor classifier. The projection is optimal when the nearest neighbors of different classes are separated by a large margin. The optimization problem is solved in the semidefinite programming framework. On the other hand, similarly to LDA, the method in [18] seeks a distance metric that collapses each class around its center and away from each other. To this end, the authors minimize the Kullback–Leibler divergence between probability distributions characterizing the data in the initial feature space and the low-dimensional feature space and show that their formulation leads to a convex optimization problem. In [36], the author presents LFDA, a distance metric learning method that focuses on multimodal data, where each class may consist of several clusters. Here, the author shows that his approach takes the local structure of the data into account and also presents non-linear extensions of the proposed method. Meanwhile, in [22,33], the authors focus on the online learning setting, where the algorithm does not have access to the full training data and only receives training examples one at a time. In this setting, the approach in [33] maximizes the margin, whereas the approach in [22] optimizes a LogDet divergence. In [21], the authors present a general distance metric learning framework to analyze the connections between various learning methods.

One notable characteristic of the above-described studies [18,22,33,36,41] is that they focus on the classification setting and do not uncover the latent structure of the data. In [32], the authors focus on the ranking setting. However, the method they proposed only optimizes the weights of a diagonal matrix. Therefore, their approach is not as flexible as those that optimize the weights of a projection matrix. Besides, the optimization problem is solved by modifying the SVM algorithm, which may be difficult to implement by non-specialists. Lastly, their method has limited visualization ability and thus cannot be used to uncover the latent structure of the data set.

In summary, none of the previously described methods simultaneously achieves high ranking prediction accuracy, interpretability, and simplicity of implementation. In the next section, we will introduce ways to address this issue.

3 Pairwise ranking component analysis

3.1 The ranking problem

3.1.1 Motivation

We consider an IR setting [7,27,28,31]. A user submits a query document to the IR system. The goal of the IR system is to present to the user a list of the database's documents, sorted

[3] http://mdp-toolkit.sourceforge.net/.

with respect to their similarity to the query: The most similar documents appear at the top of the list and the less similar ones appear at the bottom.

One simple way to do this is to compare the query with the documents using a distance function (e.g., Euclidean distance). Given the vector representation of the query and the documents, we simply compute the distances between the query vector and the document vectors. If document A is closer to the query than document B, then A is considered more relevant and should be ranked higher than B. The result is an ordered list of ranked documents, from the most relevant documents to the least relevant ones. This ordered list of documents can then be presented to the user.

In order to design efficient IR systems, we need to measure the quality of such ordered lists. In other words, we need to define a performance measure for the evaluation of the ranking accuracy. Many criteria and performance measures have been studied in the IR literature (see, e.g., [2]). In this work, we will focus on the pairwise ranking error.

3.1.2 Pairwise ranking error

First, let us consider a set of examples $S = \{(\mathbf{x}_i, y_i)\}_{i=1\ldots N}$, where each example is composed of a vector $\mathbf{x}_i \in \mathbb{R}^D$ and a class label $y_i \in [1, \ldots, C]$. In practice, S may be a document database where $\mathbf{x}_i$ represents the content of the document i, and y_i represents the document's thematic (e.g., sport, politics, religion, etc.).

To measure the quality of the IR system, we consider the pairwise ranking error [20]. First, let us consider the document $(\mathbf{x}_i, y_i) \in S$ as the query. The user is only interested in other documents similar to the query, that is, documents with the same class label y_i. Now, let us consider two documents $(\mathbf{x}_j, y_j) \in S$ and $(\mathbf{x}_k, y_k) \in S$, such that $\mathbf{x}_j$ belongs to the same class as $\mathbf{x}_i$ (i.e., $y_j = y_i$), and $\mathbf{x}_k$ belongs to any other class (i.e., $y_k \neq y_i$). As $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to same class, we expect $\mathbf{x}_i$ to be closer to $\mathbf{x}_j$ than to $\mathbf{x}_k$. If we consider the Euclidean distance, this means that $\|\mathbf{x}_i - \mathbf{x}_j\|^2$ should be smaller than $\|\mathbf{x}_i - \mathbf{x}_k\|^2$. When this is not the case, there is a ranking error. We can now consider the following quantity:

$$[\![\|\mathbf{x}_i - \mathbf{x}_j\|^2 > \|\mathbf{x}_i - \mathbf{x}_k\|^2]\!] = \begin{cases} 1 & \text{if } \mathbf{x}_i \text{ is closer to } \mathbf{x}_k \text{ than } \mathbf{x}_j \\ 0 & \text{otherwise} \end{cases}$$

where $[\![p]\!] = 1$ if the predicate p is true, 0 otherwise. In other words, this quantity equals 1 when there is a ranking error, 0 otherwise. By doing this, we are essentially casting the ranking problem in the classification setting (see, e.g., [20,38]). Considering $(\mathbf{x}_i, y_i)$ as the query, we can now compute the ranking error E_i:

$$E_i = \frac{1}{\alpha_i} \sum_{\substack{j \in C_i \\ j \neq i}} \sum_{k \notin C_i} [\![\|\mathbf{x}_i - \mathbf{x}_j\|^2 > \|\mathbf{x}_i - \mathbf{x}_k\|^2]\!]$$

where $C_i = \{j = 1 \ldots N | y_j = y_i\}$ is the set of indices corresponding to class y_i, and α_i is the number of pairs (j, k) such that $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same class, while $\mathbf{x}_i$ and $\mathbf{x}_k$ belong to different classes: $\alpha_i = \left|\{(y_j, y_k) | j \in C_i, j \neq i, k \notin C_i\}\right|$. With such a normalization, we have $E_i \in [0, 1]$, with $E_i = 0$ if all the pairs are correctly ordered and $E_i = 1$ if they are all incorrectly ordered. The total ranking error E is the sum of the normalized ranking errors over the whole data set:

$$E = \sum_{(\mathbf{x}_i, y_i) \in S} \frac{1}{\alpha_i} \sum_{\substack{j \in C_i \\ j \neq i}} \sum_{k \notin C_i} [\![\|\mathbf{x}_i - \mathbf{x}_j\|^2 > \|\mathbf{x}_i - \mathbf{x}_k\|^2]\!] \tag{1}$$

In this work, our goal is to learn a distance metric that minimizes the pairwise ranking error.

3.1.3 Distance metric learning for ranking

It is easy to see that the distance metric learning problem can be treated as a feature extraction problem. In fact, any feature extraction algorithm can be viewed as a distance metric learning algorithm. Let F be a set of functions that map initial feature space X to new feature space Z. Let d be a distance function over Z (e.g., the Euclidean distance). Then, any function $f \in F$ can be used to define a new distance function d_f in the feature space X: $d_f(\mathbf{x}_1, \mathbf{x}_2) = d(f(\mathbf{x}_1), f(\mathbf{x}_2))$. Therefore, the task of selecting a distance function can be reduced to the task of choosing a feature extraction function. In our setting, the problem of selecting a good distance function for ranking can then be viewed as selecting an appropriate feature space for ranking.

Then, what is a good feature space for the ranking setting? We define it as a feature space in which the Euclidean distance induces few ranking errors. We illustrate this idea in Fig. 1, which shows two different feature spaces (or equivalently, two different distance functions) used to represent the query and the documents. For example, we consider a database containing Internet web pages. Each web page belongs to a class defined by a general topic: Blue circles represent web pages about politics, red squares are web pages about sport, and green triangles are web pages about religion. In feature space $\mathcal{F}$ (top left), the query is a blue circle, that is, a web page about politics. The goal of the IR system is to fetch the web pages in the database and to order them into a ranked list according to their distances to the query (top right).

In this example, a perfect IR system would display all the political web pages of the database on the top of the ranked list of pages. To evaluate how good the induced ranking is, we compute the previously defined pairwise ranking error. First, we consider all possible pairs formed by a political page (i.e., a blue circle) and a non-political page (i.e., a red square or a green triangle). Then, we count the number of ranking mistakes, that is, the number of pairs where the political page is below a non-political page. In feature space $\mathcal{F}$, we count five such ranking mistakes, as shown by the five gray curves. Now, we compare the ranking approach with the classification approach, where each page is to be classified as 'relevant" or 'irrelevant" to the query. We consider a binary classifier, represented by its decision function in dashed lines. In feature space $\mathcal{F}$, there are two classification mistakes, as shown by purple arrows. In contrast, the feature space $\mathcal{G}$ (bottom left) induces two classification errors and only one ranking error.

Therefore, these two distance functions are equivalent with respect to classification accuracy, but have different performance with respect to ranking accuracy. This example underscores the need to learn a distance function specific to the ranking setting. In this work, our goal is to devise a new distance metric learning method that transforms feature space $\mathcal{F}$ into $\mathcal{G}$, that is, into a new feature space where the pairwise ranking error is minimal.

3.2 Model

3.2.1 Learning a Mahalanobis distance

In Eq. 1, the ranking error E uses the standard Euclidean distance function for ranking the data vectors: $d(\mathbf{x}_i, \mathbf{x}_j)^2 = (\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|^2$. However, as

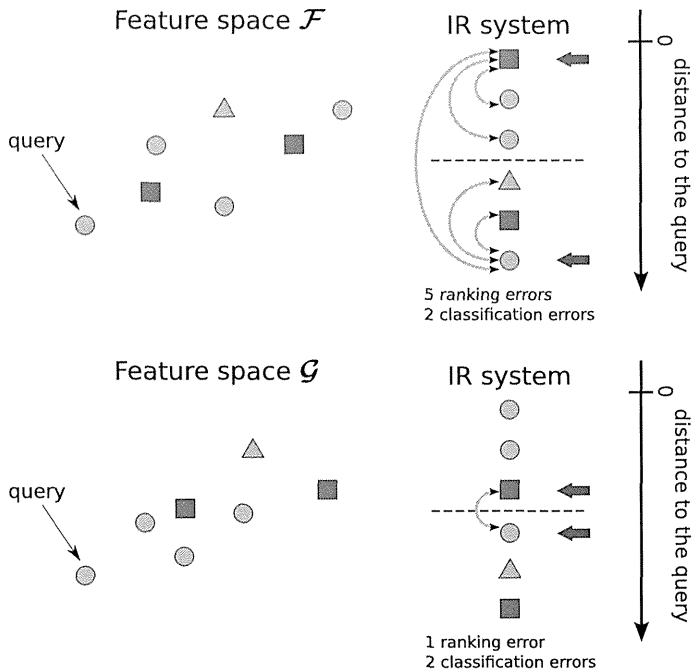

Fig. 1 The query and the documents are represented in two different feature spaces. The IR system orders the documents into a ranked list according to their distances to the query. The classification errors are shown by *purple arrows*, while the ranking errors are shown by *gray curves*. *Top*: with the feature space $\mathcal{F}$, the IR system makes two classification errors and five ranking errors. *Bottom*: with the feature space $\mathcal{G}$, the IR system makes two classification errors and one ranking error. The goal of distance metric learning is to find a transformation from $\mathcal{F}$ to $\mathcal{G}$ (color figure online)

illustrated in Fig. 1, the initial feature space may not be optimal for the ranking task, thus leading to a large ranking error. Therefore, our goal is to learn an appropriate distance function from the available data. To this end, we will consider the Mahalanobis distance: $d(\mathbf{x}_i, \mathbf{x}_j)^2 = (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{A}^T \mathbf{A}(\mathbf{x}_i - \mathbf{x}_j) = \|\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_j\|^2$ parameterized by the matrix $\mathbf{A} \in \mathbb{R}^{K \times D}$, where D is the number of dimensions of the initial feature space, and $K \leq D$ is the number of dimensions of the new feature space. It is clear that the Mahalanobis distance is equivalent to a linear projection from $\mathbb{R}^D$ to $\mathbb{R}^K$ using $\mathbf{A}$, followed by the standard Euclidean distance.

Replacing the Euclidean distance with the Mahalanobis distance in Eq. 1, the ranking error now writes:

$$E(\mathbf{A}) = \sum_{(\mathbf{x}_i, y_i) \in S} \frac{1}{\alpha_i} \sum_{\substack{j \in C_i \\ j \neq i}} \sum_{k \notin C_i} [\![\|\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_j\|^2 > \|\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_k\|^2]\!] \tag{2}$$

A good projection matrix should have a low ranking error. Therefore, the projection matrix that minimizes the ranking error is the solution of:

$$\min_{\mathbf{A} \in \mathbb{R}^{K \times D}} E(\mathbf{A})$$

3.2.2 Exponential upper bound

Equation 2 contains Boolean predicates that are not differentiable. Therefore, the cost function $E(\mathbf{A})$ is difficult to minimize. As Freund and Schapire [16] suggested, we will use $[\![x < 0]\!] < \exp(-x)$ and consider the following upper bound:

$$F(\mathbf{A}) = \sum_{(\mathbf{x}_i, y_i) \in S} \frac{1}{\alpha_i} \sum_{\substack{j \in C_i \\ j \neq i}} \sum_{k \notin C_i} \exp\left(\|\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_j\|^2 - \|\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_k\|^2\right) \tag{3}$$

It is clear that we have $E(A) < F(A)$. The new cost function 3 is now differentiable and can be minimized with standard gradient descent strategies. However, the computation of $F(\mathbf{A})$ is expensive as it requires consideration of triplets of data vectors $(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k)$. A naive implementation would have a computational complexity of $O(N^3K + NDK)$, where N is the total number of examples, D is the number of dimensions, and K is the distance metric's rank. As we want to use PARCA with large-scale data sets, a more efficient computation is required. Denoting $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$, we write $F(\mathbf{A})$ as:

$$\begin{aligned} F(\mathbf{A}) &= \sum_{(\mathbf{x}_i, y_i) \in S} \frac{1}{\alpha_i} \sum_{\substack{j \in C_i \\ j \neq i}} \sum_{k \notin C_i} \exp\left(\|\mathbf{A}\mathbf{x}_{ij}\|^2 - \|\mathbf{A}\mathbf{x}_{ik}\|^2\right) \\ &= \sum_{(\mathbf{x}_i, y_i) \in S} \frac{1}{\alpha_i} \sum_{\substack{j \in C_i \\ j \neq i}} \exp\left(\|\mathbf{A}\mathbf{x}_{ij}\|^2\right) \sum_{k \notin C_i} \exp\left(-\|\mathbf{A}\mathbf{x}_{ik}\|^2\right) \end{aligned} \tag{4}$$

With this form, the computational complexity of $F(\mathbf{A})$ is now $O(N^2K + NDK)$.

3.2.3 Derivatives

Let us differentiate the cost function F with respect to the projection matrix $\mathbf{A}$:

$$\begin{aligned} \frac{\partial F}{\partial \mathbf{A}} = -2\mathbf{A} \sum_{(\mathbf{x}_i, y_i) \in S} \frac{1}{\alpha_i} \Bigg(& \sum_{\substack{j \in C_i \\ j \neq i}} \exp\left(\|\mathbf{A}\mathbf{x}_{ij}\|^2\right) \times \sum_{k \notin C_i} \mathbf{x}_{ik}\mathbf{x}_{ik}^T \exp\left(-\|\mathbf{A}\mathbf{x}_{ik}\|^2\right) \\ & + \sum_{\substack{j \in C_i \\ j \neq i}} \mathbf{x}_{ij}\mathbf{x}_{ij}^T \exp\left(\|\mathbf{A}\mathbf{x}_{ij}\|^2\right) \times \sum_{k \notin C_i} \exp\left(-\|\mathbf{A}\mathbf{x}_{ik}\|^2\right) \Bigg) \end{aligned} \tag{5}$$

The computational complexity of the derivative $\partial F / \partial \mathbf{A}$ is $O(N^2DK)$. We can now apply standard gradient descent strategies to minimize F. Such strategies are implemented in most scientific computing libraries, for example, scipy[4] for Python or GSL[5] for C/C++. However, as the function is not convex, the optimization only will result in local minima. In practice, we run several gradient descents with random initializations and keep the best result.

[4] http://www.scipy.org/.

[5] http://www.gnu.org/s/gsl/.

3.3 Logistic PARCA and entropy-based regularization

3.3.1 Feature clustering and interpretability

As we explained in the previous section, the primary goal of distance metric learning is to improve the ranking prediction accuracy. However, we also want to uncover the latent structure of the data set. To achieve this, we will group the initial features into clusters so that the ranking error is low in the new feature space induced by the clusters. As these clusters are relevant to the ranking problem, we will interpret them as new latent, hidden features of the data, where each cluster contains semantically related features.

In the case of text data, for example, the documents are initially represented in the word space, which is not optimal for ranking. Therefore, we want to group the words into clusters, such that the ranking accuracy will improve in the new feature space induced by the clustering. The resulting clusters contain semantically related words and, therefore, can be interpreted as the latent topics of the document collection. The result is a decomposition of each document into different topics. In the case of image data, a topic is represented by a pixel cluster that forms a visual entity. The result is a decomposition of each image into different sets of pixels.

The idea of grouping the features and using the clusters as new features is related to the word clustering approaches in the text analysis community. In [35], the authors propose an information-theoretic approach to identify the word clusters that best preserve the information on the documents. In [3,4,30], the desired word clusters are the ones that best preserve the class label information of the documents. Experimental results show that the use of word clusters can improve the classification accuracy of documents [3,4].

Our approach is also related to non-negative matrix factorization (NMF) [25,26]. NMF is a popular matrix factorization method using non-negativity constraints on the learned matrices. The authors show that those non-negativity constraints enable decomposition of each matrix row into different parts. NMF has been successfully applied to image modeling [25], document clustering [43] and understanding [40], and large-scale databases [37]. As we have explained earlier, our goal is to learn a distance metric from available training data in order to minimize the ranking error in the future, yet unseen test data. However, NMF is fundamentally an unsupervised matrix factorization method, and although it has been extended to the semi-supervised setting (see, e.g., [8]), there is no straightforward way to use NMF in a supervised manner. Therefore, in contrast to PARCA and Logistic PARCA, we cannot learn an NMF model on a training set and evaluate its performance on a test set. In addition, unsupervised methods ignore class labels that are crucial to compute an appropriate distance metric, as we will show in Sect. 4. Therefore, NMF is not appropriate for the distance metric learning problem.

3.3.2 Logistic PARCA

Logistic PARCA is an extension of PARCA that groups the features into clusters for interpretability purpose. One way to group the features into clusters is to use a binary projection matrix that projects the initial high-dimensional feature space into a low-dimensional topic space. In this section, our goal is to adapt our previously described PARCA methodology to learn such a projection matrix. Formally, let $\mathbf{A}$ be a binary projection matrix of size $(K \times D)$, where D is the dimensionality of the initial feature space, and $K \leq D$ is the dimensionality of the desired low-dimensional subspace, set by the user. Each row $\mathbf{A}_k$ is a binary vector representing a topic, such that $\mathbf{A}_{kd} = 1$ if the feature d belongs to the topic k, 0 otherwise.

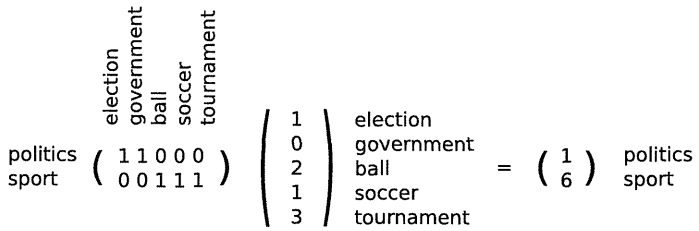

Fig. 2 *Left-hand side*: a projection matrix and a word occurrence count vector. *Right-hand side*: the corresponding topic occurrence count vector. In the projection matrix, the *first row* defines a topic related to politics, whereas the *second row* defines a topic related to sport. The document, initially represented in the word space, is projected into the topic space. In this topic space, the semantic content of the document becomes clear

The projected vector $\mathbf{Ax}$ represents the document in the topic space, where $(\mathbf{Ax})_k = \sum_{k=1}^{K} \mathbf{A}_{kd}\mathbf{x}_d$ is the occurrence count of topic k. A toy example for text data is shown in Fig. 2.

Grouping the initial features into clusters can be achieved with a slight modification of our initial PARCA methodology. First, we want the projection matrix $\mathbf{A}$ to be binary. However, this constraint requires consideration of an exponential number of matrices and, therefore, is computationally intractable. We will relax the binary constraint by allowing the elements of $\mathbf{A}$ to lie in $[0, 1]$. In order to consider matrices in $[0, 1]^{K\times D}$, we will use the logistic function $h(x) = 1/\left(1 + e^{-x}\right)$. Let $\mathbf{B} \in \mathbb{R}^{K\times D}$ be an arbitrary matrix. We denote $h(\mathbf{B})$ as the $(K \times D)$ matrix, where $[h(\mathbf{B})]_{kd} = h(\mathbf{B}_{kd})$. By construction, each element of $h(\mathbf{B})$ lies in $[0, 1]$. We now want to group the features into clusters. To do this, we will extend Eq. 4 in the following way:

$$G(\mathbf{B}, \beta) = \sum_{(\mathbf{x}_i, y_i)\in S} \frac{1}{\alpha_i} \sum_{\substack{j\in C_i \\ j\neq i}} \exp\left(\|h(\mathbf{B})\mathbf{x}_{ij}\|^2\right) \times \sum_{k\notin C_i} \exp\left(-\|h(\mathbf{B})\mathbf{x}_{ik}\|^2\right) + \beta T(h(\mathbf{B})) \tag{6}$$

where $\beta > 0$ is a user set parameter. The entropy function T is defined as:

$$T(h(\mathbf{B})) = -\sum_{d=1}^{D}\sum_{k=1}^{K} \frac{h(\mathbf{B}_{kd})}{\sum_{\ell=1}^{K} h(\mathbf{B}_{\ell d})} \log \frac{h(\mathbf{B}_{kd})}{\sum_{\ell=1}^{K} h(\mathbf{B}_{\ell d})} \tag{7}$$

To understand Eq. 7, let $p_{kd} = h(\mathbf{B}_{kd})/\sum_{\ell=1}^{K} h(\mathbf{B}_{\ell d})$. It is easy to see that $\forall(k, d)$, $p_{kd} \in [0, 1]$ and $\forall d$, $\sum_k p_{kd} = 1$. Therefore, the entropy of $\{p_{kd}\}_{k=1\ldots K}$ can be computed. Let $h(\mathbf{B}_{.d})$ be the dth column of $h(\mathbf{B})$. By definition of the entropy, the function reaches its maximum when all the elements of $h(\mathbf{B}_{.d})$ are equal, that is, when $\forall k, h(\mathbf{B}_{kd}) = 1/K$. It reaches its minimum when all the elements of $h(\mathbf{B}_{.d})$ are equal to 0 except one. If we interpret the elements of $h(\mathbf{B}_{.d})$ as clustering coefficients, the maximum entropy means that the feature d simultaneously belongs to K clusters, whereas the minimum entropy means that it belongs to one single cluster.

Hence, Eq. 6 offers a trade-off between ranking of the data vectors and clustering of the features. The trade-off is parameterized by the coefficient β. When $\beta = 0$, we only rank the data and do not try to cluster the features. For a high value of β, we focus more on clustering the features. However, if the value of β is too high, the learned matrix will ignore the ranking problem and the resulting clusters will be meaningless. Therefore, we have to empirically set β such that the features are clustered while maintaining good ranking performance. This can be performed using cross-validation techniques.

Finally, the new optimization problem we want to solve writes the following:

$$\min_{\mathbf{B}\in\mathbb{R}^{K\times D}} G(\mathbf{B},\beta)$$

To solve it, we will minimize the function G using standard gradient descent approaches. The derivative of G writes the following:

$$\frac{\partial G}{\partial \mathbf{B}} = -2h'(\mathbf{B}) \odot h(\mathbf{B}) \sum_{(\mathbf{x}_i,y_i)\in S} \frac{1}{\alpha_i} \Bigg(\sum_{\substack{j\in C_i \\ j\neq i}} \exp\left(\|h(\mathbf{B})\mathbf{x}_{ij}\|^2\right) \times \sum_{k\notin C_i} \mathbf{x}_{ik}\mathbf{x}_{ik}^T \exp\left(-\|h(\mathbf{B})\mathbf{x}_{ik}\|^2\right)$$

$$+ \sum_{\substack{j\in C_i \\ j\neq i}} \mathbf{x}_{ij}\mathbf{x}_{ij}^T \exp\left(\|h(\mathbf{B})\mathbf{x}_{ij}\|^2\right) \times \sum_{k\notin C_i} \exp\left(-\|h(\mathbf{B})\mathbf{x}_{ik}\|^2\right) \Bigg) + \beta\, T'(\mathbf{B})$$

where $\odot$ is the elementwise matrix product, $h'(\mathbf{B})$ is a $(K \times D)$ matrix of derivatives (i.e., $[h'(\mathbf{B})]_{kd} = h'(\mathbf{B}_{kd})$), h' is the derivative of the logistic function: $h'(x) = e^{-x}/\left(1+e^{-x}\right)^2$, and $T'(\mathbf{B})$ is a $(K \times D)$ matrix where the element (m, d) is the derivative $\partial T/\partial \mathbf{B}_{md}$:

$$\frac{\partial T}{\partial \mathbf{B}_{md}} = h'(\mathbf{B}_{md}) \sum_{k=1}^{K} \frac{h(\mathbf{B}_{kd})}{\sum_{\ell=1}^{K} h(\mathbf{B}_{\ell d})^2} \left(1 + \log \frac{h(\mathbf{B}_{kd})}{\sum_{\ell=1}^{K} h(\mathbf{B}_{\ell d})}\right)$$
$$- \frac{h'(\mathbf{B}_{md})}{\sum_{\ell=1}^{K} h(\mathbf{B}_{\ell d})} \left(1 + \log \frac{h(\mathbf{B}_{md})}{\sum_{\ell=1}^{K} h(\mathbf{B}_{\ell d})}\right)$$

Similarly to F, we use gradient descent to minimize the function G. As G is not convex, we run several gradient descents with random initializations to find a good local minimum.

4 Experiments

4.1 Data set description

We used eleven data sets in our experiments: four data sets from the UC Irvine repository,[6] three text data sets,[7] and four image data sets.[8] These data sets differ in the numbers of examples, features, and classes. They show balanced or unbalanced classes and different sparsity levels. A summary of the data set characteristics is shown in Table 1.

4.1.1 UCI data sets

The Balance data set was generated to model psychological experimental results. The data set contains 625 data vectors labeled in three classes. The number of features is four. In the Breast cancer diagnostic data set, images of breast cell nuclei have been digitized and converted into feature vectors. The data set contains 569 data vectors labeled in two classes (malignant and benign). The number of features is 30. In the Ionosphere data set, high-frequency radar signals are associated with different structures of the Ionosphere. The data set contains 351

[6] http://archive.ics.uci.edu/ml/index.html.

[7] http://www.cs.technion.ac.il/~ronb/thesis.html.

[8] http://www.cs.nyu.edu/~roweis/data.html.

Table 1 Characteristics of UCI data sets, text data sets, and image data sets used in our experiments

Data set	Examples	Features	Classes
Balance	625	4	3
Breast	569	30	2
Ionosphere	351	34	2
Wine	178	13	3
20 Newsgroups	18,595	1,000	20
Reuters	12,887	1,000	13
WebKB	4,199	1,000	4
Binary	1,404	320	36
MNIST	2,000	784	10
UMIST	575	10,304	20
USPS	2,000	256	10

data vectors labeled in two classes. The number of features is 34. In the Wine data set, three types of Wines are chemically analyzed. The data set contains 178 data vectors labeled in three classes. The number of features is 13.

4.1.2 Text data sets

We used the same preprocessing for the text data sets. First, we removed non-alphanumerical words as well as common words using a stop words list. Then, we removed duplicate documents and empty documents. Lastly, we removed non-informative words using information gain and kept only the top 1,000 words as the final vocabulary (see, e.g., [15,44]). Text data were then represented using the classical bag-of-words representation, where each document is a vector of word counts over the vocabulary space [31].

The 20 Newsgroups data set contains text messages posted on 20 Usenet newsgroups (e.g., "comp.sys.ibm.pc.hardware," "rec.sport.hockey," "talk.politics.guns," etc.). After preprocessing, the total number of documents is 18,595 and each class contains between 448 and 1,000 documents. The Reuters data set contains articles from the Reuters newswire, classified into different categories (e.g., "trade," "housing," "jobs," etc.). We removed categories containing less than 100 documents. After preprocessing, the data set contains 12,887 documents and 13 categories. Each category contains between 105 and 6,386 documents. The WebKB data set contains web pages collected from computer science departments of four universities. We kept four out of the seven initial categories ("course," "faculty," "project," and "student"). The resulting data set contains 4,199 web pages. Each category contains between 504 and 1,641 web pages.

Text data sets are known to be very sparse, that is, only a small fraction of the vocabulary appears in each document, resulting in data matrices mostly filled with zeros (see, e.g., [13]). The 20 Newsgroups data set is 97.9 % sparse, Reuters is 97.5 % sparse, and WebKB is 93.8 % sparse.

4.1.3 Image data sets

The Binary data set contains images of 36 handwritten letters and digits (capital letters "A" through "Z" and digits "0" through "9"). Each class contains 39 images, resulting in a total

of 1,404 images. The image size is 20×16, resulting in binary vectors of dimension 320. The MNIST data set contains 8-bit grayscale images of 10 handwritten digits ("0" through "9"). Each class contains approximately 7,000 images, from which we randomly selected 200 images. The resulting data set contains 2,000 images. The image size is 28×28, resulting in vectors of dimension 784.

The UMIST data set contains 8-bit grayscale views of 20 human faces. Each class contains between 19 and 48 images, and the total number of images is 575. The image size is 112×92, resulting in vectors of dimension 10,304. The USPS data set contains 8-bit grayscale images of 10 handwritten digits ("0" through "9"). Each class contains 1,100 images from which we randomly chose 200 images, resulting in a total of 2,000 images. The image size is 16×16, resulting in vectors of dimension 256.

4.2 Evaluation measure and experimental protocol

4.2.1 Ranking error

Let S be the test set and T be the training set. We define the normalized ranking error on the test set as:

$$E(\mathbf{A}) = \sum_{(\mathbf{x}_i, y_i) \in S} \frac{1}{\alpha_i} \sum_{\substack{j \in C_i \\ (\mathbf{x}_j, y_j) \in T}} \sum_{\substack{k \notin C_i \\ (\mathbf{x}_k, y_k) \in T}} [\![\|\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_j\|^2 > \|\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_k\|^2]\!] * \frac{1}{0.5}$$

We divided the ranking error $[\![\|\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_j\|^2 > \|\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_k\|^2]\!]$ by 0.5, which corresponds to the expected error of a random predictor. Hence, a perfect ranking algorithm will achieve $E = 0$ on the test set, while a random predictor will achieve $E = 1$.

4.2.2 Experimental protocol

We preprocessed each data set with Euclidean normalization. Then, we evaluated the ranking performance for different values of the training ratio. For each ratio, we performed a 20-fold cross-validation and computed the average ranking error on the test sets.

4.2.3 Parameter setting

In our experiments, PARCA, Logistic PARCA, LDA, and NCA are parameterized with $K = 2$ dimensions , that is, the corresponding projection matrix $\mathbf{A}$ is of size $(2 \times D)$, where D is the dimension of the initial feature space. In the case of Logistic PARCA, we always set $\beta = 0$, as in this section we only focus on the ranking performance (in Sect. 4.4, we will tune β in order to extract the latent structure of the data set). In the case of ITML, the prior Q_0 matrix is the identity matrix. We tried $\gamma \in [0.1, 1, 10]$. For each experiment, the shown ITML results always correspond to the best value of γ.

4.3 Results

4.3.1 UCI data sets

In Fig. 3, we show the ranking performance of seven methods (including the initial Euclidean distance) on four UCI data sets: Balance, Breast cancer, Ionosphere, and Wine. On the four

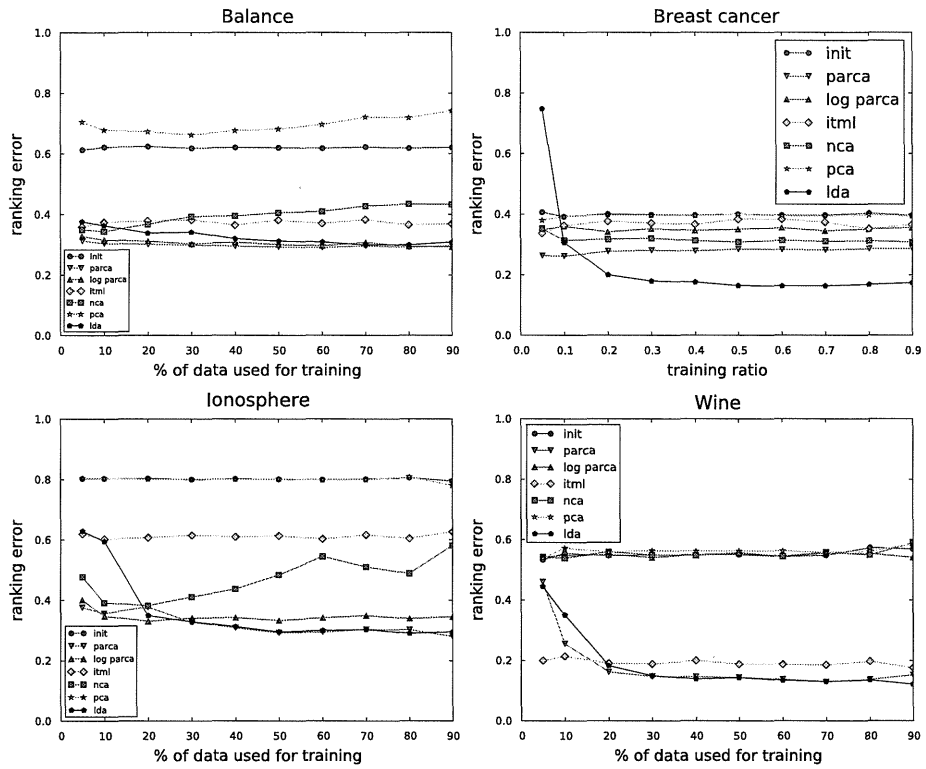

Fig. 3 Ranking performance on four UCI data sets: Balance, Breast cancer, Ionosphere, and Wine

data sets, the training ratio has minimal influence on the ranking performance of PCA. This was expected, as an unsupervised method does not have any parameter and hence does not learn from the data. Except for the Balance data set, PCA and the Euclidean distance achieve almost identical ranking errors. As we explained in Sect. 2.4, PCA only keeps the two principal components of the data set. The results for Breast cancer, Ionosphere, and Wine suggest that the two principal components identified by PCA contain sufficient information to achieve a similar ranking performance as in the initial feature space. This contrasts with the Balance data set, where the bad performance of PCA compared to that of Euclidean distance indicates that PCA loses important information about the data set.

NCA shows a surprising behavior on these UCI data sets, as its ranking error does not improve as more training data become available. On the Breast cancer and Wine data sets, the ranking error of NCA does not depend much on the training ratio. On the Balance and Ionosphere data sets, the ranking error of NCA noticeably degrades as more training data become available. This is surprising, as we expect the ranking error to improve as more training data become available. However, NCA optimizes a classification error rather than a ranking error. We suspect that a distance function relevant to the classification task (as defined by NCA) is sometimes irrelevant to the ranking task. Hence, the optimization of a classification error may drive NCA away from learning a good distance function for the ranking task.

In the case of ITML, the training ratio seems to have little influence on the ranking accuracy. This suggests that this method only needs few training data to achieve optimal performance.

ITML shows good ranking performance on the Balance and Wine data sets. For the Wine data set, it is interesting to note that ITML outperforms PARCA and LDA for low values of the training ratio. As we explained in Sect. 2.3, ITML is a regularized distance metric learning method. It seems that for this data set, the regularization efficiently prevents overfitting the training data. However, the regularization parameter γ has to be set by cross-validation. This makes ITML more difficult to use than the other methods.

Overall, PARCA and LDA achieve the best ranking performance for high training ratios. However, on the Balance, Breast cancer, and Ionosphere data sets, LDA suffers from overfitting problems when few training data are available. As we explained in Sect. 2.1, the LDA projection collapses each class to its center, while keeping the class centers away from each other. However, when a few training data are available, the training examples might be poor representatives of their class, and hence, LDA will collapse them away from the true class centers. When a new test vector is projected using LDA, it will not be close to its true class center, and therefore, it will be misclassified. As a result, on the Balance, Breast cancer, and Ionosphere data sets, PARCA performs better than LDA when few training data are available. In addition, the ranking error of PARCA does not change much with the training ratio. These results suggest that PARCA only needs a small amount of training data to achieve good performance, which is a highly desirable property.

We also noticed that PARCA always performs similarly to or better than Logistic PARCA. Recall that when $\beta = 0$ in Logistic PARCA, the only difference between the two methods lies in the constraints over the projection matrix: $\mathbf{A} \in \mathbb{R}^{K \times D}$ for PARCA and $\mathbf{A} \in [0, 1]^{K \times D}$ for Logistic PARCA. Therefore, the class of distance functions considered by Logistic PARCA is more limited, and we expect PARCA to achieve better prediction performance when the data are complex. The results on the four UCI data sets seem to confirm this hypothesis.

In Fig. 4, we show the visualization results on the UCI Ionosphere test set for 5 % of training data. Of course, we expected that a method good at ranking will also show good class separation. Accordingly, we found that the data projected by PARCA and NCA, both of which achieve low ranking errors, show similar shapes. This contrasts with LDA, where one of the two classes (the blue squares) is less scattered and shows a more Gaussian shape. This is not surprising, as LDA seeks to minimize the intraclass variance and to maximize the interclass variance. Finally, PCA maximizes the total variance of the data set and ignores the class labels, resulting in a suboptimal class separation for ranking. As we explained in Sect. 2.3, ITML learns a Mahalanobis matrix and thus cannot be used for visualization purpose.

4.3.2 Text data sets

In Fig. 5, we show the ranking performance on three text data sets: 20 Newsgroups, Reuters, and WebKB. Due to the size of these data sets, we only considered training ratios between 2 and 10 %. For all three data sets and for all training ratios, the initial Euclidean distance performs better than PCA. The poor performance of PCA stems from its unsupervised nature. Because PCA ignores the class labels, the first two principal components are not relevant for the ranking task.

Despite being a supervised method, LDA shows poor performance. For all the three data sets, LDA performs worse than the Euclidean distance for most values of the training ratio. As it was experimentally observed on the UCI data sets and in other works [29], LDA has a severe tendency to overfit when few training data are available. The results of LDA on the text data sets suggest that this is also the case here. Hence, it seems that when only few

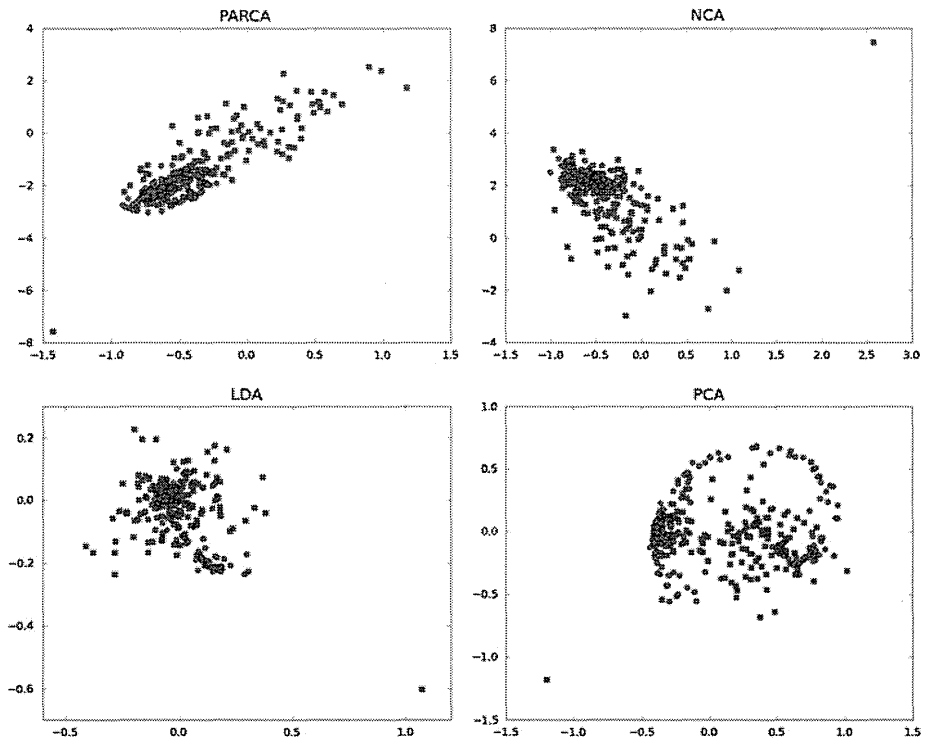

Fig. 4 Visualization results of the UCI Ionosphere test set, for 5 % of training data. *Each colored symbol* represents one of the two classes (color figure online)

training data are available, LDA should not be used for the ranking task. On 20 Newsgroups and WebKB, ITML performs only slightly better than the Euclidean distance. However, it shows good ranking accuracy on the Reuters data set.

PARCA achieves the best ranking error for all the three data sets and all the training ratios, while NCA and Logistic PARCA show similar performance. As we previously noticed with the UCI data sets, PARCA always performs slightly better than Logistic PARCA. The performance of NCA improves as more data become available for training, except on the Reuters data set. This confirms what we previously noted on the UCI data sets and suggest that the classification error optimized by NCA is sometimes irrelevant for ranking. The performance of PARCA always improves as more data become available for training. As we noted previously, PARCA only needs a few training data to achieve a low ranking error.

In Figs. 6 and 7, we show the visualization results on the 20 Newsgroups and Reuters test sets for 2 % of training data. We can see that PARCA and NCA are the two best methods at separating the classes on both data sets. The separation quality of PARCA is noticeably better than that of NCA on the Reuters data set. This is not surprising, as PARCA achieves the lowest ranking error. As we analyzed previously, LDA shows severe overfitting problems on both data sets. In the resulting projections, the classes tend to collapse together. Finally, PCA focuses on maximizing the variance of the projected data and ignores the class labels. As a result, the projected data are widely spread and the class information is lost.

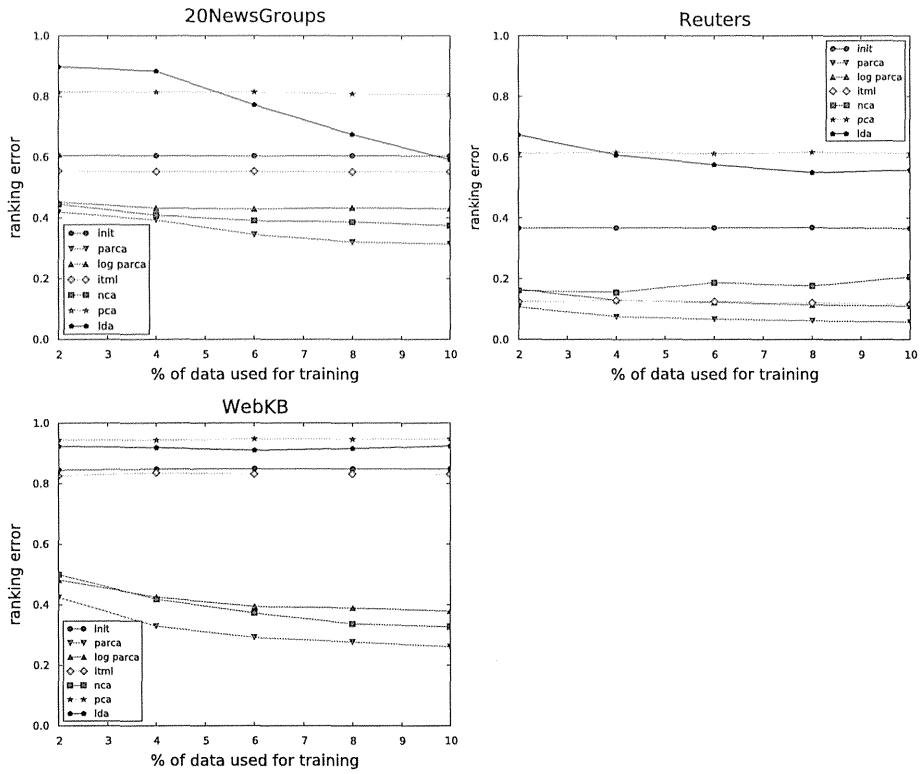

Fig. 5 Ranking performance on three text data sets: 20 Newsgroups, Reuters, and WebKB

4.3.3 Image data sets

In Fig. 8, we show the ranking performance on four images data sets: Binary alphadigits, MNIST handwritten digits, UMIST faces, and USPS handwritten digits. The results generally confirm what we noted previously on the UCI and text data sets. The Euclidean distance and PCA achieve similar performance on the Binary and MNIST data sets. However, the Euclidean distance performs better than PCA on the UMIST and USPS data sets. As we observed previously, PCA is not expected to perform better than the Euclidean distance because it ignores the class labels.

As noted on the UCI and text data sets, LDA shows severe overfitting problems for low training ratios on the Binary, MNIST, and USPS data sets. Then, the performance of LDA improves as more training data become available. As we noticed on the UCI and text data sets, the performance of ITML does not depend much on the training ratio. ITML shows good ranking performance on MNIST and USPS and significantly outperforms the other methods on the Binary alphadigits data set. In the latter data set, we observed that ITML achieves the best ranking accuracy for all values of the regularization parameter γ. This suggests that for this particular data set, the information-theoretic cost function optimized by ITML is relevant for the ranking problem. As we explained in Sect. 2, LDA and ITML are difficult to use with high-dimensional data. Therefore, the performance of LDA and ITML could not be assessed in the UMIST data set ($D = 10{,}304$).

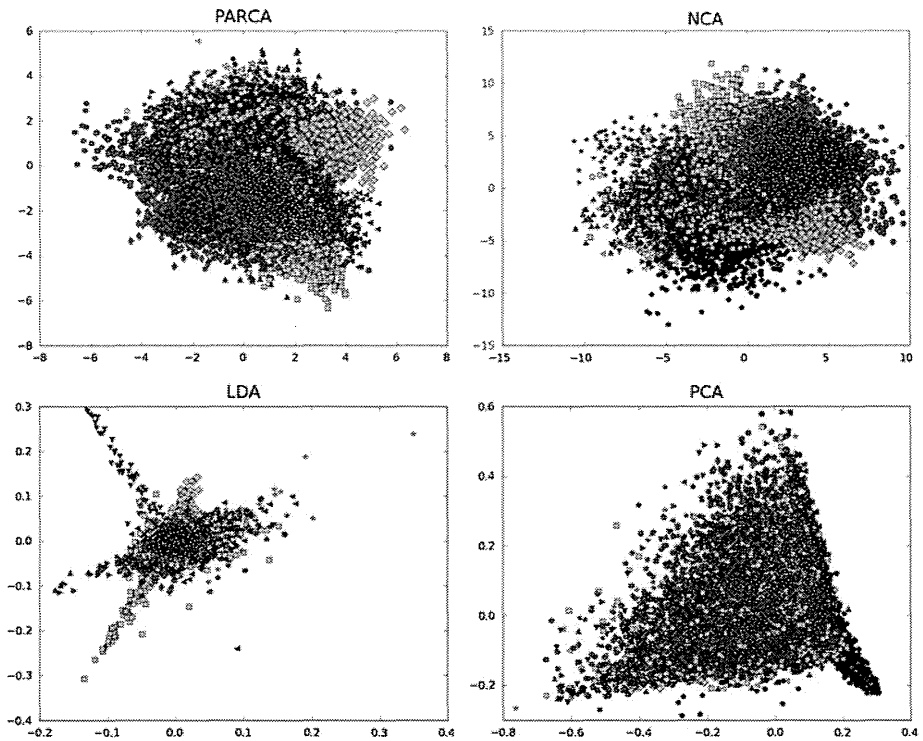

Fig. 6 Visualization results on the 20 Newsgroups test set, for 2 % of training data. *Each colored symbol* represents one of the 20 classes (color figure online)

Except for Binary alphadigits, PARCA achieves the best ranking performance for most values of the ranking ratio. Once again, we noticed that unlike LDA, PARCA needs a few training data to learn a good ranking model. The performance of Logistic PARCA and PARCA is similar to the four data sets. They are almost identical to the UMIST faces data set, suggesting that even though Logistic PARCA is less expressive than PARCA, the data set is sufficiently simple that both methods achieve similar performance.

In Fig. 9, we show the visualization results on the USPS handwritten digits for 10 % of training data. According to their ranking performance, PARCA and NCA achieve the best class separation. As noticed previously on the 20 Newsgroups and Reuters data sets, LDA overfits the data and collapses all the classes together. As expected with PCA, the projected data achieve poor class separation and high variance.

4.3.4 Time and memory performance

As we have explained in Sect. 3.2, the computational complexity of PARCA is dominated by $O(N^2K + NDK)$ for $F(\mathbf{A})$ (Eq. 4) and $O(N^2DK)$ for $\partial F/\partial \mathbf{A}$ (Eq. 5), where N is the number of examples, D is the dimensionality of the initial feature space, and K is the dimensionality of the final feature space. As a result, the number of samples N has a stronger influence than D and K on the running time. Table 2 shows the average running times of the five best performing methods: LDA, ITML, NCA, PARCA, and Logistic PARCA. The running times have been averaged over 20 experiments. For all data sets except the three