Pattern Recognition 42 (2009) 2392 – 2402

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Enhanced graph-based dimensionality reduction with repulsion Laplaceans ☆

E. Kokiopoulou [a,*], Y. Saad [b]

[a] Seminar for Applied Mathematics, ETH, Zurich, Switzerland
[b] Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455, USA

ARTICLE INFO

Article history:
Received 28 October 2008
Received in revised form 31 March 2009
Accepted 6 April 2009

Keywords:
Linear dimensionality reduction
Orthogonal projections
Supervised learning
Face recognition
Graph Laplacean

ABSTRACT

Graph-based methods for linear dimensionality reduction have recently attracted much attention and research efforts. The main goal of these methods is to preserve the properties of a graph representing the affinity between data points in local neighborhoods of the high-dimensional space. It has been observed that, in general, supervised graph-methods outperform their unsupervised peers in various classification tasks. Supervised graphs are typically constructed by allowing two nodes to be adjacent only if they are of the same class. However, such graphs are oblivious to the proximity of data from different classes. In this paper, we propose a novel methodology which builds on 'repulsion graphs', i.e., graphs that model undesirable proximity between points. The main idea is to repel points from different classes that are close by in the input high-dimensional space. The proposed methodology is generic and can be applied to any graph-based method for linear dimensionality reduction. We provide ample experimental evidence in the context of face recognition, which shows that the proposed methodology (i) offers significant performance improvement to various graph-based methods and (ii) outperforms existing solutions relying on repulsion forces.

1. Introduction

The goal of dimensionality reduction [3] is to map high-dimensional data samples to a lower dimensional space such that certain properties are preserved. Graph-based methods (e.g., [5,10,18,14]) have attracted much research interest over the past few years. These methods typically rely on some graph to capture the salient geometric relations of the data in the high-dimensional space. This graph is usually called an affinity graph [14], since its edge set conveys some information about the proximity of the data in the input space. Once the affinity graph has been constructed, these methods derive the low-dimensional samples by imposing that certain graph properties be preserved in the reduced space. This typically results in an optimization problem, whose solution provides the reduced data, or a mechanism to project data from the original space to low-dimensional space.

In general, it has been observed that supervised graph-based methods outperform significantly their unsupervised peers in various recognition tasks. It is common practice to construct a supervised graph by only setting adjacent the nodes from the same class (see, e.g., [14]). The intuition is that during projection, when the data locality (or local geometry) is preserved, points from the same class will be mapped to points that are close by. This, however, has one particular weakness; namely that points from different classes but nearby in some other measure (e.g., Euclidean distance) may be projected to points that are close-by in the low-dimensional space. This may lead to potential misclassification.

To remedy this weakness, we propose a methodology based on repulsion graphs. A repulsion graph is a graph whose edge set captures pairs of points that belong to different classes, but are close by in the input space. Maximizing the pairwise distances between these points will tend to repel these points from one another when they are projected. The proposed framework based on repulsion graphs is generic and can be applied to any graph-based method to improve its classification performance. The idea of repulsion forces, or negative energies, has been previously used in another context in graph-drawing techniques [15,17] and in dimensionality reduction under different formulations (see, e.g., [24,25]). The latter approaches use the k-NN graph to define attraction and repulsion forces. However, they both fail to exploit the full supervision information as the attraction forces are only defined among nearest neighbors.

In contrast, our methodology applies attraction forces to all points of the same class, in addition to using repulsion forces, and this in turn brings significant discriminant information. We provide experimental evidence which shows that (i) including repulsion forces in

☆ Work supported by NSF under Grants DMS 0810938 and DMS 0528492, and by the Minnesota Supercomputing Institute.

* Corresponding author. Tel.: +41 44 632 7643.
E-mail addresses: effrosyni.kokiopoulou@epfl.ch (E. Kokiopoulou), saad@cs.umn.edu (Y. Saad).

0031-3203/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2009.04.005

various graph-based methods can significantly boost their performance and (ii) the proposed framework outperforms other competing solutions based on related repulsion ideas. In short, the contribution of our paper is twofold: (i) we introduce a new methodology for defining attraction and repulsion forces and (ii) we provide a theoretical analysis of the spectral properties of the resulting repulsion matrix as well as a physical interpretation of the repulsion forces.

The rest of the paper is organized as follows. First, in Section 2 we establish the concept of affinity graphs that capture the data proximity and then we discuss in Section 3 how these graphs are typically employed in different graph-based dimensionality reduction methods. Then, in Section 4 we introduce the proposed methodology on repulsion graphs. Next, in Section 5.1 we analyze the spectral properties of the involved matrix and in Section 5.2 we provide a physical interpretation of the repulsion forces in our framework. Finally, Section 6 presents our experimental results.

2. Affinity graphs and their properties

Assume a set of high-dimensional data samples

$$X = [x_1, x_2, \ldots, x_n] \in R^{m\times n}. \tag{1}$$

Due to the high dimension, it is common practice to use graphs in order to model the geometry of the data and also to cope with the curse of dimensionality. Thus, we often define a graph

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}) \tag{2}$$

whose nodes $\mathcal{V}$ correspond to the data samples and the edge set $\mathcal{E}$ models the relations between them. When we build the graph, depending on whether we use the class labels or not, we distinguish between two different cases: supervised and unsupervised.

Supervised case: *the class graph*. Assume that we have c classes and that the data are given as in (1) along with their class labels $\ell : [1, \ldots, n] \to [1, \ldots, c]$. Here $\ell(i) = j$ means that the ith data sample belongs to the jth class. For the supervised case the class labels are used to build the graph. It has been observed in general that supervised methods perform better in many classification tasks relative to the unsupervised ones. The motivation here is to build the graph in a discriminant way in order to reflect the categorization of the data into different classes. One simple approach is to build the data graph in (2) such that an edge $e_{ij} = (x_i, x_j)$ exists if and only if x_i and x_j belong to the same class. In other words, we make adjacent those nodes that belong to the same class.

Consider now the structure of the induced adjacency matrix A. Let n_i be the number of samples which belong to the ith class. Observe that the data graph G consists of c cliques, since the adjacency relationship between two nodes reflects their class relationship. This implies that with an appropriate reordering of the columns and rows, the adjacency matrix A will have a block diagonal form, where the size of the ith block is equal to the size n_i of the ith class. Hence, A will be of the following form:

$$A = \mathrm{diag}(A_1, A_2, \ldots, A_c).$$

In the above, the block A_i corresponds to the ith class.

The unsupervised case: neighborhood graphs. In the unsupervised case the class labels are not used and we typically define the edge set $\mathcal{E}$ in (2) in a way that captures the proximity of the data in the input high-dimensional space. The k-NN graph is one very popular example that belongs to this category. In the k-NN graph two nodes x_i and x_j are made adjacent only if x_i is among the k nearest neighbors of x_j or vice versa. Another typical example is the ε-graph. In this case, a node x_i is made adjacent to all nodes x_j, $j \neq i$ that are within distance ε from it.

2.1. Graph weights

Edge weights are assigned in order to determine how each sample is influenced by its neighbors. This amounts to defining a weight matrix $W \in R^{n\times n}$ whose sparsity pattern is inherited by the adjacency matrix A. A few popular choices of weights are reviewed below.

Binary weights. The weight matrix W is simply set equal to the adjacency matrix A.

Gaussian weights. The weight matrix W is defined as follows:

$$W_{ij} = \begin{cases} e^{-\|x_i - x_j\|^2/t} & \text{if } A_{ij} \neq 0, \\ 0 & \text{otherwise.} \end{cases}$$

These weights are also known as heat kernel weights. Note the presence of the parameter t.

Reconstruction weights. These weights were first introduced in [18,20]. The weight matrix in this case is built by computing optimal coefficients which relate a given point to its nearest neighbors in some locally optimal way. Each data sample along with its k nearest neighbors are assumed to (approximately) lie on a locally linear manifold. Hence, each data sample x_i is (approximately) reconstructed by a linear combination of its k nearest neighbors. The reconstruction errors are measured by minimizing the objective function

$$f(W) = \sum_i \left\| x_i - \sum_j W_{ij} x_j \right\|_2^2. \tag{3}$$

The weights w_{ij} represent the linear coefficient for reconstructing the sample x_i from its neighbors $\{x_j\}$. For a fixed i, the weights w_{ij} are nonzero only when i and j are neighbors, and their sum is constrained to be equal to one. There is a simple closed-form expression for the weights. For details see [18,20].

2.2. Graph Laplaceans

Graph Laplaceans provide one of the most common and useful tools for dimensionality reduction, see, e.g. [5,10,14,13,18]. Let $\mathcal{N}(k)$ denote the adjacency list of a vertex x_k. Then, a graph Laplacean is a matrix $L \in R^{n\times n}$, which has the following property:

$$L_{ij} \begin{cases} \leqslant 0 & \text{for } j \in \mathcal{N}(i),\ j \neq i, \\ = -\sum_{k\neq i} L_{ik} & \text{if } i = j. \end{cases}$$

For instance, observe that when W is a weight matrix, and if D is a diagonal matrix with $D_{ii} = \sum_j W_{ij}$, then $L = D - W$, is a graph Laplacean. The fundamental property of graph Laplaceans is that for any vector x of n scalars x_i, $i = 1, \ldots, n$, we have: $x^\top L x = \frac{1}{2}\sum_{ij} W_{ij}|x_i - x_j|^2$. This relation can be generalized to arrays with n column-vectors $y_i \in \mathcal{R}^m$ as follows:

$$\mathrm{Tr}[YLY^\top] = \frac{1}{2}\sum_{i,j=1}^{n} W_{ij}\|y_i - y_j\|_2^2, \tag{4}$$

where $Y \in \mathcal{R}^{m\times n}$ (see [14,13]). Finally, note that graph Laplaceans have been used extensively for clustering, see, e.g., [21], and for the closely related problem of graph partitioning [1]. The paper [23] gives a good overview of graph Laplaceans and their properties.

3. Graph-based dimensionality reduction: overview

Given a data set X as in (1), the goal of dimensionality reduction is to produce a set Y which is an accurate representation of X, but whose dimension d is much less than the original dimension m. This can be achieved in different ways by selecting the *type* of producing the reduced dimension data Y as well as the desirable *properties*