Neurocomputing 165 (2015) 75–80

Contents lists available at ScienceDirect

Neurocomputing

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

Highlighting data clusters by graph embedding

Wenye Li *

Macao Polytechnic Institute, Rua de Luís Gonzaga Gomes, Macao SAR, China

ARTICLE INFO

Article history:
Received 28 February 2014
Received in revised form
5 June 2014
Accepted 5 July 2014
Available online 23 April 2015

Keywords:
Clustering
Graph embedding
Semi-definite programming

ABSTRACT

We propose a novel method, modularity embedding, to embed high-dimensional data or graphs in a low-dimensional space. Central to our work is a model that quantifies the relationship of two data points by their *pairwise modular value*. A larger value indicates a higher chance that they should be placed near to each other, and vice versa. The objective function of the model has a simple formulation of minimizing the sum of squared distances between data points weighted by pairwise modular values. It is naturally relaxed as a semi-definite program that learns a low-rank kernel matrix with only one linear constraint, which can be solved efficiently by modern mathematical optimization solvers. Compared with traditional graph embedding algorithms, the proposed method is shown to be able to highlight cluster structures inherent in high-dimensional data and graphs, which provides a promising tool in data analysis applications.

1. Introduction

Clustering refers to the task of grouping a set of objects such that objects in the same group are more similar to each other than to those in different groups. As a common technique for statistical data analysis, it has been studied and used in many fields [24,15]. Different approaches need to be developed to reveal the underlying cluster relationship for different types of data. In this work, we are particularly interested in the cluster structures of high-dimensional data, which has attracted much recent research attention. High-dimensional data, with anywhere from tens to thousands or even millions of features, are often encountered in a variety of applications like videos, images, texts and complex networks. Being hard to think in and impossible to visualize, the high dimensionality poses significant difficulties and challenges to modern data processing research [18].

As a treatment to this "curse of dimensionality", dimensionality reduction techniques have been investigated during the past decades, particularly in areas of statistics, neural networks, machine learning and computing sciences. These techniques try to reduce the number of random variables under consideration with the assumption that high-dimensional data have an intrinsic dimension that is significantly lower than the number of features they appear to have. Common techniques include principal component analysis and metric multi-dimensional scaling [16,9]. The two classical linear methods project the data from a high-dimensional space into a low-dimensional subspace by either maximizing the projected variance or best preserving the pairwise squared distance among the data.

Besides linear methods, much research on nonlinear techniques has been devoted. Well-known methods include self-organizing map, generative topographic map and related [17,3]. These methods can be regarded as a type of neural networks that is trained using unsupervised learning to produce a low-dimensional representation of the input space of the samples, which have been successfully applied in many challenging tasks [4].

More recent work of nonlinear techniques focuses on graph embedding methods. These embedding methods build upon but go beyond the classical linear solutions. They assume that the data are from a low-dimensional manifold that is embedded in a high-dimensional space, which is more general than the assumption of subspace by linear methods. The methods often start from building a sparse connectivity graph describing local relationship between each data point and its neighbors. The graph serves as an approximation to the underlying data manifold. With the graph, a compact representation of the data in a low-dimensional space can be obtained in different ways [8,2,11,33,28–31]. These methods differ in preserving different signatures of the underlying manifold, such as the geodesic distances between inputs and the local combination angles. These distinct features make the embedding algorithms applicable in different domains.

In this paper, we develop a novel embedding method for analysis of high-dimensional data and graphs. Compared with existing approaches, the proposed method tries to find a low-dimensional depiction of graphs with the objective of highlighting inherent cluster structures by moving intra-cluster points together, and pushing inter-cluster points apart. In empirical evaluation, we found that the method

* Tel.: +853 8599 6434.
E-mail address: wyli@ipm.edu.mo

http://dx.doi.org/10.1016/j.neucom.2014.07.085
0925-2312/

often produces separation of clusters far more evident than other methods.

From a computational point of view, the objective function of the proposed model can be naturally relaxed and solved by a semi-definite program with a linear constraint, which provides an effective and efficient solution. Such a formulation is also flexible in incorporating prior knowledge which can often be expressed as linear equality or inequality constraints.

The paper is organized as follows: we briefly review the necessary background on semi-definite programming, which plays an important role to the success of the proposed model. Then we illustrate the modularity embedding model in detail. Finally we report our empirical evaluation of the method with promising results and conclude the work.

2. Background on semi-definite programming

Semi-definite programming (SDP) is a relatively new field which is of growing interest, and dramatic advances have been made recently [25,32]. SDP deals with convex optimization problems over symmetric positive semi-definite matrix variables with linear objective function and linear constraints. It may be regarded as an extension of, but much more general than, linear programming.

Denote by $\mathbf{S}^n$ the space of all $n \times n$ real symmetric matrices, equipped with the inner product $\langle X, Y\rangle = \mathrm{tr}(X^T Y) = \sum_{ij=1}^{n} x_{ij} y_{ij}$, where x_{ij} and y_{ij} denote the (i,j)th element of matrices X and Y respectively. A matrix X in $\mathbf{S}^n$ is positive semi-definite if all its eigenvalues are nonnegative; we write $X \succcurlyeq 0$. For given matrices $A, A_1, \ldots, A_m$ in $\mathbf{S}^n$, SDP optimizes problems of the type

$$\max \mathrm{tr}\left(A^T X\right) \tag{1}$$

subject to

$$\mathrm{tr}\left(A_i^T X\right) = c_i, \quad i = 1, 2, \ldots, m$$
$$X \succcurlyeq 0$$

in variables $X \in \mathbf{S}^n$.

SDP has a particular structure that makes its solution computationally tractable by interior-point and related methods [5,7], and the optimal solution can be approximated to arbitrary precision. It is now used in a host of applications, including relaxation of combinatorial optimizations and machine learning problems [13,19,34,33].

3. Modularity embedding

3.1. Model

For an undirected graph $G = (V, E)$, where $V = \{v_1, \ldots, v_n\}$ is a set of vertices and E is a set of edges connecting pairs of vertices in V. Let w_{ij} be an element of the adjacency matrix W of the graph, which gives the number of edges between vertices v_i and v_j. We further denote $m_i = \sum_j w_{ij}$ as the degree of v_i and $m = \frac{1}{2}\sum_i m_i$ as the total edge number.

Our proposed work is based on the notion of "modularity" in the study of complex networks [26]. Assume that the degree m_i associated with each vertex v_i is preserved. Therefore under uniform random selection, the expected number of edges between vertices v_i and v_j is $m_i m_j / 2m$. A value

$$b_{ij} = w_{ij} - \frac{m_i m_j}{2m}$$

gives the observed number of edges minus the expected number of edges between vertices v_i and v_j. In this paper, we refer to this value, b_{ij}, as the *pairwise modular value* between vertices v_i and v_j. It is positive if the edge weight between vertices v_i and v_j is larger than the expected weight between them. It is zero or negative otherwise. Thus the value quantifies a kind of affinity between the two vertices. A larger b_{ij} suggests a stronger connection between the two vertices and a higher chance that they are in the same cluster, and vice versa.

An $n \times n$ square matrix B with all elements $\{b_{ij}\}$ is called a modularity matrix [26]. It has a nice property. All row values and all column values of the matrix sum to zero, i.e.,

$$\sum_j b_{ij} = 0, \quad \text{for all } i \tag{2}$$

and

$$\sum_i b_{ij} = 0, \quad \text{for all } j. \tag{3}$$

Our work wishes to find a low-dimensional embedding of the graph G such that the cluster structure is highlighted. Considering the close relationship between the vertices' cluster membership and their pairwise modular values, this goal can be achieved by placing two vertices near to each other if their pairwise modular value is large, and pushing them far away otherwise. Mathematically it can be expressed as minimizing the sum of squared distances between data points weighted by their pairwise modular values, which is formulated by a *modularity embedding* model with the following objective:

$$\min_X \sum_{ij} b_{ij} \times \ell(x_i, x_j) \tag{4}$$

subject to a normalization constraint:

$$\sum_i x_i^T x_i = n. \tag{5}$$

Here $X = [x_1, \ldots, x_n]$ is a $d \times n$ matrix, where x_i gives the coordinate of the vertex v_i in a space with given dimension d and $\ell(x_i, x_j)$ is the squared Euclidean distance between x_i and x_j.

3.2. SDP relaxation

The model formulated in Eq. (4) can be relaxed and solved by a semi-definite program. Note that $\ell(x_i, x_j) = x_i^T x_i + x_j^T x_j - 2x_i^T x_j$, the optimization objective becomes

$$\begin{aligned}\sum_{ij} b_{ij} \times \left(x_i^T x_i + x_j^T x_j - 2x_i^T x_j\right) &= \sum_i x_i^T x_i \sum_j b_{ij} + \sum_j x_j^T x_j \sum_i b_{ij} - 2\sum_{ij} b_{ij} x_i^T x_j \\ &= -2\sum_{ij} b_{ij} x_i^T x_j\end{aligned}$$

The last "=" holds due to the properties in Eqs. (2) and (3).

Let $S = X^T X$, which implicitly enforces a constraint on positive semi-definiteness of S, a constraint on the trace $\mathrm{tr}(S) = n$ which comes from the normalization constraint in Eq. (5) and a constraint on the rank $rank(S) = d$. Subject to these constraints, our objective can be written equivalently as the maximization of $\mathrm{tr}(B^T S)$.

The rank constraint on S makes this problem generally hard to solve. Here we point out without proof that with the constraint a special case of our optimization problem becomes an NP-hard binary partition problem discussed in [6]. Fortunately, there is a simple relaxation heuristic that is usually found effective in practice. We go on with the optimization by neglecting the rank