Neurocomputing 144 (2014) 448–462

Contents lists available at ScienceDirect

Neurocomputing

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

Latent tree models for rounding in spectral clustering ☆

April H. Liu [a], Leonard K.M. Poon [b,*], Teng-Fei Liu [a], Nevin L. Zhang [a]

[a] Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China
[b] Department of Mathematics and Information Technology, The Hong Kong Institute of Education, 10 Lo Ping Road, Hong Kong, China

ARTICLE INFO

Article history:
Received 29 October 2013
Received in revised form
14 February 2014
Accepted 30 April 2014
Communicated by S. Choi
Available online 29 May 2014

Keywords:
Spectral clustering
Rounding
Latent tree models

ABSTRACT

In spectral clustering, one defines a similarity matrix for a collection of data points, transforms the matrix to get the so-called Laplacian matrix, finds the eigenvectors of the Laplacian matrix, and obtains a partition of the data points using the leading eigenvectors. The last step is sometimes referred to as *rounding*, where one needs to decide how many leading eigenvectors to use, to determine the number of clusters, and to partition the data points. In this paper, we propose a novel method using latent tree models for rounding. The method differs from previous rounding methods in three ways. First, we relax the assumption that the number of clusters equals the number of eigenvectors used. Second, when deciding how many leading eigenvectors to use, we not only rely on information contained in the leading eigenvectors themselves, but also make use of the subsequent eigenvectors. Third, our method is model-based and solves all the three subproblems of rounding using latent tree models. We evaluate our method on both synthetic and real-world data. The results show that our method works correctly in the ideal case where between-clusters similarity is 0, and degrades gracefully as one moves away from the ideal case.

1. Introduction

Clustering is a data analysis task where one assigns similar data points to the same cluster and dissimilar data points to different clusters. It is an important topic in machine learning, as well as related fields such as statistics, pattern recognition and data mining. The most commonly used clustering methods are probably mixture models and K-means. Those methods yield 'globular' clusters and perform poorly when the true clusters are non-convex and are shaped, for instance, like crescent moons.

Spectral clustering [2] is one way to overcome the aforementioned shortcoming. It has gained prominence in recent years. The idea is to convert clustering into a graph cut problem. More specifically, one first builds a *similarity graph* over the data points using a measure of data similarity as edge weights and then partitions the graph by cutting some of the edges. Each connected component in the resulting graph is a cluster. The cut is done so as to simultaneously minimize the cost of cut and balance the sizes of the resulting clusters [3,4].

The graph cut problem is NP-hard and is hence relaxed. In the relaxed problem, the cluster indicator functions are allowed to be real-valued. The solution is given by the leading eigenvectors of the so-called Laplacian matrix, which is a simple transformation of the original data similarity matrix. In a post-processing step, a partition of the data points is obtained from those real-valued eigenvectors. This post-processing step is called *rounding* [5,6].

In this paper we focus on rounding. Although the spectral clustering literature is abundant, there are relatively few papers on rounding. In general, rounding is considered an open problem. There are three subproblems: (1) decide how many (and more generally which) leading eigenvectors to use; (2) determine the number of clusters; and (3) determine the members of each cluster. Among the three, the first two subproblems are considered much harder.

Previous rounding methods fall into two groups depending on whether they assume the number of clusters is given. When the number of clusters is known to be k, rounding is usually done based on the first k eigenvectors. The data points are projected onto the subspace spanned by those eigenvectors and then the K-means algorithm is run on that space to get k clusters [2]. Bach and Jordan [6] approximate the subspace using a space spanned by k piecewise constant vectors and then run K-means on the latter space. This turns out to be equivalent to a weighted K-means algorithm on the original subspace. Zhang and Jordan [7] observe a link between rounding and the orthogonal Procrustes problem in Mathematics and iteratively use an analytical solution for the latter problem to build a method for rounding. Rebagliati and Verri [8] ask the user to provide a number K that is larger than k and

☆This paper extends the earlier work by Poon et al. [1].

* Corresponding author.
E-mail addresses: aprillh@cse.ust.hk (A.H. Liu), kmpoon@ied.edu.hk (L.K.M. Poon), liutf@cse.ust.hk (T.-F. Liu), lzhang@cse.ust.hk (N.L. Zhang).

http://dx.doi.org/10.1016/j.neucom.2014.04.030
0925-2312/

obtain k clusters based on the first K eigenvectors using a randomized algorithm that repeatedly calls K-means as a subroutine.

When the number of clusters is not given, one needs to estimate it. A common method is to manually examine the difference between every two consecutive eigenvalues starting from the first two. If a big gap appears for the first time between the kth and $(k+1)$th eigenvalues, then one uses k as an estimate of the number of clusters. Zelnik-Manor and Perona [9] propose an automatic method. The method considers a number of integers. For each integer k, it tries to rotate the first k eigenvectors so as to align them with the canonical coordinate system for the eigenspace spanned by those vectors. A cost function is defined in terms of how well the alignment can be achieved. The k with the lowest cost is chosen as an estimate for the number of clusters. Xiang and Gong [10] and Zhao et al. [11] question the assumption that clustering should be based on all the eigenvectors from a continuous block at the beginning of the eigenvector spectrum. They use heuristics to choose a collection of eigenvectors which do not necessarily form a continuous block, and then use Gaussian mixture models to determine the number of clusters and to partition the data points. Socher et al. [12] assume the number of leading eigenvectors to use is given. Based on those leading eigenvectors, they determine the number of clusters and the membership of each cluster using a non-parametric Bayesian clustering method.

In this paper, we propose and study a novel model-based approach to rounding. The method differs from the previous methods in three ways. First, we relax the assumption that the number of clusters equals the number of eigenvectors that one uses for rounding. In the *ideal case* where between-cluster similarity is 0, if one knows the number k_t of true clusters, one can indeed recover the k_t clusters from the first k_t eigenvectors. However, this might not be the case in non-ideal cases or when the number of clusters one tries to obtain is not k_t. Our method allows the number of clusters to differ from the number of eigenvectors. This is conducive to robust performance in non-ideal cases.

Second, we choose a continuous block of leading eigenvectors for rounding just as Zelnik-Manor and Perona [9]. The difference is that when deciding the appropriateness of the first k eigenvectors, Zelnik-Manor and Perona use only information contained in those eigenvectors, whereas we also use information contained in subsequent eigenvectors. So our method uses more information and hence the choice is expected to be more robust.

Third, we solve all the three subproblems of rounding and we do so within one class of models, namely latent tree models [13]. In contrast, most previous methods assume that the first two subproblems are solved and the solutions are equal, and focus only on the third subproblem. Xiang and Gong [10] and Zhao et al. [11] do consider all three subproblems. However, they do not solve all the subproblem within one class of models. They first choose a collection of eigenvectors based on some heuristics and then use Gaussian mixture models to solve the other two subproblems. Zelnik-Manor and Perona [9] also consider all three subproblems. However, their method is not model-based and it assumes the number of clusters equals the number of eigenvectors. An advantage of the model-based approach is that its performance degrades gracefully as we move away from the ideal case.

The remaining paper is organized as follows. In Section 2 we review the basics of spectral clustering and point out two key properties of the eigenvectors of the Laplacian matrix in the ideal case. In Section 3 we describe a straightforward method for rounding that takes advantage of the two key properties. This method is fragile and breaks down as soon as we move away from the ideal case. In Sections 4 and 5 we propose a model-based method for rounding that exploits the same two properties. The method is named LTM-Rounding. It is evaluated on synthetic data in Section 6 and is compared with other methods in Section 7. This paper concludes in Section 8.

2. Basics of spectral clustering

In this section we review the basics of spectral clusters and point out two properties that we exploit later.

2.1. Similarity measure and similarity graph

Let $X=\{\boldsymbol{x_1},\ldots,\boldsymbol{x_n}\}$ be a set of n data points in an Euclidean space $\mathbb{R}^d$. In order to partition the data, one needs to define a non-negative *similarity measure* s_{ij} for each pair $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ of data points. This can be done in a number of ways. In our work we consider two measures:

- *k-NN similarity measure*: $s_{ij}=1$ if $\boldsymbol{x_i}$ is one of the k nearest neighbors of $\boldsymbol{x_j}$, or vice versa, and $s_{ij}=0$ otherwise.
- *Gaussian similarity measure*: $s_{ij}=\exp(-||\boldsymbol{x_i}-\boldsymbol{x_j}||^2/\sigma^2)$, where σ is a parameter that controls the width of neighborhood of each data point.

The matrix $S=(s_{ij})_{i,j=1,\ldots,n}$ is called the *similarity matrix*.

Given a similarity measure, the data can be represented as an weighted undirected graph G. In the graph there is a vertex v_i representing each data point $\boldsymbol{x_i}$, and there is an edge between two vertices v_i and v_j if and only if $s_{ij}>0$. The value s_{ij} is used as the edge weight and is sometimes denoted as w_{ij}. The graph is called the *similarity graph* and its adjacency matrix $W=(w_{ij})_{i,j=1,\ldots,n}$ is the same as the similarity matrix S. Note that the similarity graph G is a complete graph when the Gaussian similarity measure is used, and it might not be so when the k-NN similarity measure is used.

2.2. Graph Laplacian

In spectral clustering one transforms the similarity matrix to get another matrix called the *graph Laplacian* matrix. There are a number of Laplacian matrices to choose from [2]. In this paper, we use the *normalized Laplacian matrix* L_{rw} given by

$$L_{rw}=I-D^{-1}S, \tag{1}$$

where $I\in\mathbb{R}^{n\times n}$ is the identity matrix, $D=(d_{ij})\in\mathbb{R}^{n\times n}$ is the diagonal *degree matrix* given by $d_{ii}=\sum_{j=1}^{n}s_{ij}$, and D^{-1} is the inverse of D. The following proposition is well-known [2].

Proposition 1. *The Laplacian matrix L_{rw} satisfies the following properties*:

1. *L_{rw} is positive semi-definite.*
2. *The eigenvalues of L_{rw} are non-negative and the smallest one is* 0.
3. *If the similarity graph is connected, then there is only one eigenvalue that equals* 0.
4. *The unit vector $\mathbf{1}\in\mathbb{R}^n$ that consists of all* 1*'s is an eigenvector for eigenvalue* 0.

The eigenvalues of L_{rw} are arranged in ascending order as $0=\lambda_1\le\lambda_2\le\cdots\le\lambda_n$ and the eigenvectors for the eigenvalues are arranged in the same order as $\boldsymbol{e}_1,\boldsymbol{e}_2,\ldots,\boldsymbol{e}_n$. The eigenvectors at the front of the list are called the *leading eigenvectors*. Note that an eigenvector of L_{rw} is a vector of n real numbers. It can also be viewed as a function over the data points. As a matter of fact, in the graph cut formulation of spectral clustering [2], an eigenvector is a cluster indicator function for a cut. Two example eigenvectors are shown in Fig. 1.