Extracting out Top-N Formal Concepts in dynamic order

Aixiang Li
Hokkaido University

Makoto Haraguchi
Hokkaido University

Abstract

In this paper, we propose an algorithm for extracting Top N formal concepts based on their evaluation values computed from a document-term co-occurrence table. The basic strategy is to use a branch bound depth-first algorithm for closures of formal concepts, and in this paper, we adopt the eigenvector corresponding to the least positive eigenvalue of the laplacian Matrix for the document-term data table. The eigenvector is then used to define a dynamic ordering of candidate documents from which formal concepts are constructed. We will give a performance evaluation.

1. Background

Given a document and term co-occurrence table, one of our tasks is trying to find out the groups of similar documents. For this purpose, various soft and/or hard clustering methods are considered as standard ones. However, it is hard to give an adequate number of clusters or their sizes beforehand. In addition, it is also not an easy task to find the semantic meaning of clusters thus obtained.

According to the previous studies, each document is supposed to be represented as a vector, and then a cluster of documents is determined by the distance from its gravity point. As long as the documents are processed in this way, the meaning of documents and clusters as well is not fully taken to be considered. On the other hand, as a formal concept consists of extent (documents) and intent (common terms), it is much easier to grasp the meaning of clusters.

Such a formal concept (FC) is placed in a space called a FC lattice. More general FCs with more number of documents and with fewer terms are placed at upper part. Conversely, more specific ones are located at lower part in the lattice. For those FCs, it is relatively easy to find them by using top-down and/or bottom-up mining algorithms. To the contrary, an efficient method for FCs at the middle part is not yet fully developed, as the number of those FCs is so huge.

In [1] [2], an efficient approach has been invented as a kind of clique search in static order, and it has performed much better than others; but it still takes longer time. So, in this paper, we try to design a new algorithm that is basically a closure enumerator based on some dynamic ordering of documents. Then, how about the result? We will show it in the next.

2. Spectrum Analysis

Given a set of data points $x_1,...x_n$ and some notion of similarity $s_{ij} \geq 0$ between all pairs of data points x_i and x_j [3], a nice way of representing the data is in the form of the undirected similarity graph $G=(V,E)$, the vertices v_i represent the data points x_i, and the edge is weighted by s_{ij}. The problem of clustering can be expressed as: we want to find a partition of the graph such that the edges between groups have a very low weight and the edges within a group have high weight [3]. From the graph G, we can get an adjacency matrix $W=(w_{ij})_{i,j=1,...n}$ (with $w_{ij}=w_{ji}=s_{ij}$). The degree of a vertex $v_i \in V$ is defined as

$$d_i = \sum_{j=1}^{n} w_{ij}$$

The degree matrix D is defined as the diagonal matrix with the degrees $d_1,...,d_n$ on the diagonal. The unnormalized graph Laplacian matrix is defined as

$$L = D - W$$

It is shown in [3] that the coordinates in the eigenvector corresponding to the second smallest eigenvalue (the least positive eigenvalue) of L indicates some indices useful to obtain clusters of similar nodes. In this paper, we call the coordinate *score*. That is the score is a measure of their similarity. A cluster of similar nodes will consist of nodes with closer scores.

In our strategy, we consider a document-term co-occurrence table as a matrix $C(n \times m)$, n is the number of documents and m is the number of terms. W is defined as

$W(n \times n) = C \times C^t$ ($C^t$is the transpose matrix of C)

By JAMA package [4], we get the eigenvector corresponding to the 2nd smallest eigenvalue, and then we assign *scores* to every document respectively.

3. Problem definition

Input: Document-term co-occurrence table;
The constraint δ (temporarily the size of intent);
N (Top N)
Output: Top N Formal Concepts in their qualities

with respect to the evaluation of extents.

4. Algorithm

Basically, we still adopt the branch bound depth-first algorithm;
The main procedure is closure calculation:

$$\psi(\varphi(X \cup x_i))$$

X : closure, the extent of current formal concept;

x_i : candidate of X;

φ : intent calculation;

ψ : extent calculation;

Evaluation formula:
$\| extent \|$ the size of extent(set of documents)
Constraint:
$\delta = \|$ intent$\|$

Pruning rule:
Same as in [1][2],
1) If $(\varphi(X \cup x_i)) < \delta$, We can never obtain the δ valid formal concepts involving X as well as x_i. So we delete the document x_i from the candidate list of documents to be added to X.
2) If $\| extent \| + \| canditates \| <=$ the minimum evaluation value of extents in Top N list.
In this case, the size (evaluation value) of a possible extension of the present extent is at most the value in left hand side that cannot exceed any of the current top N FCs. Therefore, we can safely prune the present extent and its any extension.

Dynamic order:
Before generating next FC (or closure), we calculate the score of X ; then for the candidates (satisfying the constraint) of X , we sort them by their distances of scores to X ; In order to increase the size of extent more quickly, we select candidates in descending order of distances. Anyway we attempt to obtain the Top N FCs earlier by changing the original fixed order.

Updating the Top N list:
When the candidate set of a FC becomes empty, then the FC is checked, if its evaluation value is more than the minimum one in the list, it will update the element of Top N list, otherwise will not.

5. Experiment

Temporarily we only used a *tf* data formed by 1000 documents and 500 terms, Of course the Boolean data is also applicable, from this point, we can consider our method as a co-clustering one; and we believe that to the sparse data it may be more efficient.

We made experiments in three kinds of orders under the same δ and N. The 1st is in static degree order that is computed as a pre-processing. We calculated the degrees of documents, and reordered them in descending order and kept the order fixed in the whole branch expanding process. The 2nd is in static score order. We also need some pre-processing to calculate scores of documents and order them in its ascending order; and the 3rd is in dynamic order.

In this paper, we only show one result to illustrate the experimental fact:
Environment: *OS:windows XP, CPU: pentium4 2.8G, RAM: 1.0G, Language: java*

N=2 $\delta = 25$	degree-static	score -static	dynamic
Time(s)	98	21	22

6. Discussion and conclusion

From the simple result in the above, we assert that the ordering candidates by scores from the eigenvector of Laplacian both in static order and in dynamic order all improved the speed to about 20% of the degree order. For other values of δ and N we have experimented, the time decreased to even below. That is just the effect of spectrum analysis.

There exist some cases for which FC finding based on the static ordering is faster than the dynamic ones. One of the major reasons is that the information supplied by the least positive eigenvalue is not enough to evaluate plausible search direction. For such cases, it is an idea to use eigenvectors of higher eigenvalues, just as Spectrum Clustering.
Even though the score- static performs some faster than dynamic in some case, the current experimental result increases our confidence toward dynamic order.

Reference

[1] *M.Haraguchi, Y.Okubo, "An Extended Branch and Bound Search Algorithm for Finding Top-N Formal Concepts of documents", Springer-LNCS 4384, pp.276-288, 2007*

[2] *Y.Okubo, M.Haraguchi, "Finding Conceptual Documents Clusters with Improved Top-N Formal Concept Search", Proceeding of IEEE/WIC/ACM WI-2006*

[3] *Ulrike von Luxburg, "A Tutorial on Spectral Clustering", Technical Report No. TR-149, Max Planck Institute for Biological Cybernetics*

[4] *http://math.nist.gov/*