잠재적 소셜 네트워크를 이용하여 스펙트럼 분할하는 방식 기반 영화 추천 시스템

일홈존[1], 펭소니[2], 싯소포호트[1], 김대영[3], 박두순*[3]
[1]순천향대학교 소프트웨어융합학과 석사과정
[2]순천향대학교 소프트웨어융합학과 박사과정
[3]순천향대학교 컴퓨터소프트웨어학과 교수

ilkhomjon.sadriddinov@gmail.com(일홈존), dyoung.kim@sch.ac.kr(김대영), parkds@sch.ac.kr(박두순)

A Movie recommendation using method of Spectral Bipartition on Implicit Social Network

Sadriddinov Ilkhomjon[1], Sony Peng[1], Sophort Siet[1], Dae-Young Kim[2], and Doo-Soon Park[2]
[1]Dept. of Software Convergence, Soonchunhyang University
[2]Dept. of Computer Software Engineering, Soonchunhyang University

Abstract

We propose a method of movie recommendation that involves an algorithm known as spectral bipartition. The Social Network is constructed manually by considering the similar movies viewed by users in MovieLens dataset. This kind of similarity establishes implicit ties between viewers. Because we assume that there is a possibility that there might be a connection between users who share the same set of viewed movies. We cluster users by applying a community detection algorithm based on the spectral bipartition. This study helps to uncover the hidden relationships between users and recommend movies by considering that feature.

1. Introduction

Network (also known as graph, we interchangeably use these terminologies) is a discrete complex structure and a way of modeling different real-world systems [1]. Analyzing networks is a very broad field from its abstract theory to practical applications. However, not all the facets are of interest to us. Detecting communities, partitioning the graph into several groups, is the main part that we have found useful for our research experiment.

In order to detect communities, we constructed a network of users by establishing a connection between them if both of them have the same set of viewed movies in common. Aforementioned criterion is a mere assumption and a way of forming a network of users. In the context of Social Network Analysis (SNA), these sorts of relationships are called implicit ties. Because in contrast to explicit tie, the implicit tie is not a direct connection between users (or any object that is assumed as nodes), instead, the tie is mediated by representation of specific features [2]. We then classified new users to discovered clusters by a similarity metric.

The contribution of this paper is that we showed how recommendation systems can leverage from the hidden relationships among users, in particular, a built implicit social network.

The remainder of this research paper is

*Corresponding author: Doo-Soon Park, Dae-Young Kim
Acknowledgments: This research was supported by the National Research Foundation of Korea (No. NRF-2022R1A2C1005921) and BK21 FOUR (Fostering Outstanding Universities for Research) (No.5199990914048)

organized as follows. Section 2 provides important terminology, formally defines what is the graph, community and related theoretical context. Section 3 contains the experiment where we implemented spectral partitioning. In Section 4, we discussed the result and represented the evaluation outcome.

2. Preliminaries: Definitions and examples

In this section we discuss graphs, their structural properties and spectral partitioning alongside the Laplacian matrix.

In the formal context, a graph G consists of a finite nonempty set V of objects called vertices (the singular form is vertex) and a set E of 2-element subsets[2] of V called edges [3]. It is also common to represent $V(G)$ and $E(G)$ for vertex set and edge set of graph G, respectively.

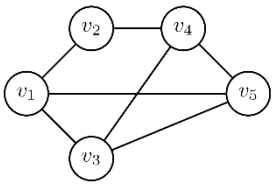


Figure 1

In the Figure 1, the vertex set of graph[3] G is $V(G) = \{v_1, v_2, v_3, v_4, v_5\}$ and the edge set of G is

$$E(G) = \{\{v_1, v_2\}, \{v_1, v_3\}, \{v_1, v_5\}, \{v_2, v_4\}, \{v_3, v_4\}, \{v_3, v_5\}, \{v_4, v_5\}\}$$

In contrast to Figure 1, there is another way of representing graph – an adjacency matrix – which is very convenient for further computation and analysis. The definition of the adjacency matrix A for an undirected graph to be the matrix with elements:

$$A_{ij} = \begin{cases} 1, \textit{ if there is an edge joining vertices } i,j, \\ 0 \quad \textit{otherwise} \end{cases} \quad (1)$$

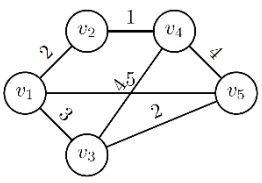
In our study, we built a weighted network, where the connections are not merely binary entities (like the definition (1)), that can be represented also mathematically by an adjacency matrix [4]:

$$A_{ij} = w \quad (2)$$

where, w is a weight on edge between i and j vertices. The weight can be also expressed as a map:

$$w\colon E \longrightarrow \mathbb{Z}^+ \quad (3)$$

Note that, in general, the weight can be any real number ($w(\{i,j\}) \in \mathbb{R}$), however, we restrict our attention in this paper to network having weights only with non-negative integers. For example:



$$\equiv \begin{array}{c} \begin{matrix} v_1 & v_2 & v_3 & v_4 & v_5 \end{matrix} \\ \begin{pmatrix} 0 & 2 & 3 & 0 & 5 \\ 2 & 0 & 0 & 1 & 0 \\ 3 & 0 & 0 & 4 & 2 \\ 0 & 1 & 4 & 0 & 4 \\ 5 & 0 & 2 & 4 & 0 \end{pmatrix} \end{array} \begin{matrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{matrix} \quad (4)$$

Figure 2

Since graph and its representation are defined, now we can move on another property – a degree of a vertex. The degree of a vertex in an undirected graph is the number of edges incident with it, except that a loop at a vertex contributes twice to the degree of that vertex [5]. The degree of the vertex i is denoted by $\deg(i)$. We can further define the $\deg(i)$ function with the summatory form using definition (1):

$$deg(i) = k_i = \sum_j A_{ij} \quad (5)$$

We turn now to the substructure of a network which is known to be a community(partition). In the context of graph theory, the field concerned with such a structure is community detection. When $G(V,E)$ and $C(W,F)$ are graphs, C is called to be a subgraph of G, written as $C \subseteq G$, if $W(C) \subseteq V(G)$ and $F(C) \subseteq E(G)$ are true [3]. At the same time a community is a subgraph. One can see other terminologies that can be used interchangeably with community detection: graph or network clustering. And yet it is not universally defined [6]. Therefore, we provide a concise and easy-to-understand definition – gathering of vertices into groups such that there is a higher density of edges within groups that between them [7]. Many community detection algorithms work with unweighted networks. And yet there are some works that generalize the algorithm by carrying over with little or no modification so that it is extended to work with weighted networks. We start our discussion with

[2] Unordered pairs of vertices.

[3] We assume here and for the rest of paper that the network is an undirected graph having bidirectional edges.

spectral partitioning of graph that is analogous to the leading eigenvector method [8]. The problem statement is that a given graph is required to be bisected into two subgraphs so that the number of edges removed must be as few as possible.

In a formal context, the number of removed edges is called a cut size:

$$R = \frac{1}{2} \sum_{\substack{i,j \ in \\ different \\ subgraphs}} A_{ij} \quad (6)$$

For our algorithm, we need to define a Laplacian matrix:

$$L_{ij} = \begin{cases} k_i, & i = j, \\ -w(i,j), & i \neq j \ and \ there \ is \ an \ edge \\ 0 & otherwise \end{cases} \quad (7)$$

(If a graph $\boldsymbol{G}$ is unweighted, then indices of $\boldsymbol{L_G}$ correspond to edges of graph $\boldsymbol{G}$ are -1)

If a graph $\boldsymbol{G(V,E)}$ is given and the $\boldsymbol{L_G}$ is a Laplacian matrix of $\boldsymbol{G}$, then the eigenvalues and corresponding eigenvectors of $\boldsymbol{L_G}$ are $\boldsymbol{\lambda_1, \lambda_2, \lambda_3, \ \ldots, \ \lambda_n}$ and $\mathbf{x_1, x_2, x_3, \ \ldots, \ x_n}$, respectively(where $\boldsymbol{n = |V|}$).

To find the bipartition, we take the second eigenvector of the Laplacian matrix, $\mathbf{x_2}$, corresponding to $\boldsymbol{\lambda_2}$(these are also known to be Fiedler value and Fiedler vector [8]).

As a result, now we can show partitions that are $\boldsymbol{P_1 = \{v_i |\ v_i \in V \ and \ x_i^{(2)} \geq 0\}}$ and $\boldsymbol{P_2 = \{v_i |\ v_i \in V \ and \ x_i^{(2)} < 0\}}$, where the $\mathbf{x_i^{(2)}}$ is $\boldsymbol{i}$th element of $\mathbf{x}_2$.

The computational complexity of spectral clustering is $\boldsymbol{O(n^3)}$, where $\boldsymbol{n}$ is the number of vertices in a given graph [9].

3 Implementation

In this section, we represent an implementation of the method described in Section 2.

3.1 Dataset

MovieLens100K contains 943 users and 1682 items(movies). There exist 100,000 ratings, where ratings are in the 0-5 range. We also used the demographic vector of users provided inside this dataset. It contains age, occupation, and gender.

In Table 1 below we expressed some important information with corresponding set of notations for our further experiment.

3.2 Constructing an Implicit Social Network

Based on the criterion, an assumption that there is a relationship between users who share the same set of movies, we built a network $\boldsymbol{G}$.

Notation	Description
U	Set of users
M	Set of movies
r_{um}	Rating for movie m by user $u(m \in M, u \in U)$
$\widehat{r_{um}}$	Predicted rating for movie m by user $u(m \in M, u \in U)$
d_u	Demographic vector of user $u(u \in U)$
M_u	Proper subset of M representing a set of movies watched by user $u(u \in U)$
U_{train}	Proper subset of U representing a set of users used for training model
U_{test}	Proper subset of U representing a set of users used for testing model

Table 1

We split user dataset into two parts: training set ($\boldsymbol{U_{train}}$) and testing set ($\boldsymbol{U_{test}}$) with the ratio 4:1. Now considering described assumptions above, network $\boldsymbol{G}$ is defined as $\boldsymbol{G(V,E)}$, where $\boldsymbol{V = U_{train}}$ and $\boldsymbol{E = \left\{\{u_i, u_j\} \middle|\ u_i, u_j \in U_{train}, M_{u_i} \cap M_{u_j} \neq \emptyset, u_i \neq u_j\right\}}$. Below the adjacency matrix $\boldsymbol{A}$ of a network $\boldsymbol{G}$ is defined.

$$A_{u_i u_j} = \begin{cases} 0, & M_{u_i} \cap M_{u_j} = \emptyset \quad or \ u_i = u_j, \\ n, & |\ M_{u_i} \cap M_{u_j}| = n \end{cases} \quad (8)$$

Note that, the newly composed graph is undirected, weighted, and does not contain self-loop[4].

Once the construction process has been completed, we can see that the network had 730 vertices and 256,482 edges. Network is very dense in terms of number of edges, almost close to its maximum limit 266,085 ($\boldsymbol{n(n-1)/2, n = |V|}$). The reason for this is easy to state - given a set $\boldsymbol{U_{train}}$ consisting of $\boldsymbol{n}$ users, a graph was formed by establishing an edge between couple of users even they share at least single movie in common. In a real-world scenario, it is natural to see that phenomenon: many viewers can have a similar view history of movies.

3.3 Partitioning a network

Since the algorithm described in Section 2

[4]An edge that connects a vertex to itself.

bisects a network into two parts, we iterated the same process of partitioning for each of those two parts to get four subgraphs. Based on the definition of a subgraph in Section 2, we can now show the following expression:

$$\bigcup_{i=1}^{4} \boldsymbol{C_i} = \boldsymbol{C_1} \cup \boldsymbol{C_2} \cup \boldsymbol{C_3} \cup \boldsymbol{C_4} \subseteq \boldsymbol{G} \quad (9)$$

Even though it is very obvious that the Eq. (10) is true, this is a very important property that shows there are no overlaps among discovered subgraphs.

$$\bigcap_{i=1}^{4} C_i = C_1 \cap C_2 \cap C_3 \cap C_4 = \emptyset \quad (10)$$

Table 2 provides reader the descriptive information about the subgraphs.

Subgraphs	Number of vertices	Number of edges
C_1	192	18209
C_2	201	19937
C_3	159	12531
C_4	178	15719

Table 2

3.4 Classification to Clusters

Now we turn to classifying users from test set ($\boldsymbol{U_{test}}$) to the clusters by considering the similarity of their demographic vectors ($\boldsymbol{d_u}$). When $\boldsymbol{d_{u_i}}, \boldsymbol{d_{u_j}} \in \mathbb{R}^{\boldsymbol{3}}$ is given, let $\boldsymbol{sim}$ to be a function takes two vectors as parameter and generates a Euclidean distance between them:

$$sim\left(d_{u_i}, d_{u_j}\right) = \sqrt{\sum_{k=1}^{3} (d_k^{(u_i)} - d_k^{(u_j)})^2} \quad (11)$$

For classification we need to find the centroids (in Eq. (12)) of each cluster(subgraph) so that then we can find the similarity with other users from the test set ($\boldsymbol{U_{test}}$). When $\boldsymbol{k}$th cluster is defined as $\boldsymbol{C_k}(\boldsymbol{V_{C_k}}, \boldsymbol{E_{C_k}})$, then a centroid of it is

$$\alpha_k = \frac{1}{n} \sum_{i=1}^{n} d_{u_i} \quad (12)$$

Where, $\boldsymbol{n} = |\boldsymbol{V_{C_k}}|$ and $\boldsymbol{u_i}$ is corresponding user for the vertex $\boldsymbol{v} \in \boldsymbol{V_{C_k}}$.

Based on Eq. (11) and (12), now it is possible to define a classification function $\boldsymbol{\delta}$. When $\boldsymbol{u_i} \in \boldsymbol{U_{test}}$ is given, we can define the classification function for user $\boldsymbol{u_i}$ as

$$\delta(u_i) = \begin{cases} 1, & sim(d_{u_i}, \alpha_1) \text{ is the smallest,} \\ 2, & sim(d_{u_i}, \alpha_2) \text{ is the smallest,} \\ 3, & sim(d_{u_i}, \alpha_3) \text{ is the smallest,} \\ 4, & sim(d_{u_i}, \alpha_4) \text{ is the smallest} \end{cases} \quad (13)$$

4 Results and Discussion

After the classification, we evaluated the outcome of our algorithm. There are plenty of metrics out there, however, we preferred ***MAE*** (Mean Absolute Error) over others. If we consider the single user $\boldsymbol{u_i} \in \boldsymbol{U_{test}}$ and want to find the ***MAE*** metric to see how well the model predicted the set of ratings for $\boldsymbol{u_i}$, then we can use Eq. (14):

$$MAE_{u_i} = \frac{1}{n} \sum_{m \in M_{u_i}} \left(r_{u_i m} - \widehat{r_{u_i m}}\right)^2 \quad (14)$$

Where, $n = |M_{u_i}|$.

Below we show how to predict the rating of a user $\boldsymbol{u_i}$ on movie $\boldsymbol{m}$.

$$\widehat{r_{u_i m}} = \frac{1}{n} \sum_{u_j \in C_{\delta(u_i)}} r_{u_j m} \quad (15)$$

Where, $\boldsymbol{n} = \left|\{\boldsymbol{u_j} | \boldsymbol{u_j} \in \boldsymbol{C_{\delta(u_i)}} \ \boldsymbol{and} \ \boldsymbol{0 \le r_{u_j m} \le 5}\}\right|$.

We calculated ***MAE*** value 10 times: each time we selected 10 random users and 10 random movies watched by the corresponding users. On average, the ***MAE*** value was 0.95. In our case, $MAE_{u_i} \in [0,5]$ is true, since $r_{um} \in \{0,1,2,3,4,5\}$ is also true for $m \in M$, $u \in U$.

We can interpret 0.95 as the error that might occur when we apply our method. More precisely, if we want to recommend a movie based on the rating calculation, shown in Eq. (14), then on average there is a possibility that the model slightly varies than the actual rating.

Reference

[1] Pedro Ribeiro, Pedro Paredes, Miguel E.P. Silva, David Aparicio, Fernando Silva. A Survey on Subgraph Counting: Concepts, Algorithms and Applications to Network Motifs and Graphlets. ACM Comput. Surv. 54, 2, Article 28, 36 pages, March 2022.

[2] Gasparetti, F., Micarelli, A., Sansonetti, G. (2018). Community Detection and Recommender Systems. In: Alhajj, R., Rokne, J. (eds) Encyclopedia of Social Network Analysis and Mining. Springer, New York, NY. June 2018.

[3] Gary Chartrand, Ping Zhang. Introduction to Graph Theory. New York, NY 10020. Mc Graw Hill. 2005.

[4] Newman, M. Analysis of Weighted Networks. Physical review. E, Statistical, nonlinear, and soft matter physics. 70. 2004.

[5] Kenneth H. Rosen. Discrete Mathematics and Its Applications. New York, NY 10020. Mc Graw Hill. 2012.

[6] Santo Fortunato, Darko Hric. Community detection in networks: A user guide. Physics Reports. Volume 659. 2016

[7] Clauset, Aaron & Newman, M & Moore, Cristopher.

Finding community structure in very large networks. Physical review. E, Statistical, nonlinear, and soft matter physics. 70. 2005.
[8] Newman, M. Finding Community Structure in Networks Using the Eigenvectors of Matrices. Physical review. E, Statistical, nonlinear, and soft matter physics. 74. 2006.
[9] Donghui Yan, Ling Huang, and Michael I. Jordan. Fast approximate spectral clustering. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '09). 907–916. 2009.