이학석사 학위논문

Local-Ensemble Graph Collaborative Filtering with Spectral Co-Clustering

스펙트럴 이중 군집화를 이용한 그래프기반 협업필터링의 국지 앙상블 방법

2022년 08월

서울대학교 대학원

협동과정 계산과학

정호인

Local-Ensemble Graph Collaborative Filtering with Spectral Co-Clustering

지도교수 강 명 주

이 논문을 이학석사 학위논문으로 제출함

2022년 08월

서울대학교 대학원
협동과정 계산과학
정호인

정호인의 이학석사 학위논문을 인준함

2022년 08월

위 원 장	________________	(인)
부 위 원 장	________________	(인)
위 원	________________	(인)

Local-Ensemble Graph Collaborative Filtering with Spectral Co-Clustering

by

HOIN JUNG

A THESIS

Submitted to the faculty of the Graduate School
in partial fulfillment of the requirements
for the degree Master of Science
in the Department of Computational Science & Technology
Seoul National University
Aug 2022

Abstract

Local-Ensemble Graph Collaborative Filtering with Spectral Co-Clustering

HOIN JUNG
Computational Science & Technology
The Graduate School
Seoul National University

The importance of a personalized recommendation system is emerging as the world becomes more complex and individualized. Among various recommendation systems, Neural Graph Collaborative Filtering(NGCF) and its variants treat the user-item set as a bipartite graph and learn the interactions between user and item without using their unique features. While these approaches only using collaborative signals have achieved state-of-the-art performance, they still have the disadvantage of abandoning feature similarity among users and items. To tackle this problem, we adopt unsupervised community detection from bipartite graph structure to enhance the collaborative signal for a Graph-based recommendation system. Co-Clustering algorithms segment the user-item matrix into small groups. Each local CF captures a strong correlation among these local user-item subsets, while the original incidence matrix is also used to analyze global interaction between groups. Finally, our Local-Ensemble Graph Collaborative Filtering(LEGCF) aggregates all local and global collaborative information. As the proposed approach can utilize various Co-clustering and Collaborative Filtering flexibly, one of the most straightforward variants, Spectral Co-Clustering and NGCF, can enhance the overall performance.

Keywords : Recommendation System, Graph Convolutional Network, Collaborative Filtering, Spectral Co-Clustering
Student number : 2020–21619

Contents

Chapter 1

Introduction

The importance of personalized recommendation is emerging as social media and marketing system is getting more complex and embodied. Moreover, as it becomes easier to crawl more detailed and immense data, various types of recommendation systems have been developed. One of the most generic methods is finding a pattern for the feature of users and items such as age, gender, country, genre, price, etc[1]. Some methods utilize time stamp sequential information of given features. These content-based methods analyze the features of items that users interacted positively with and recommend items with similar features. Undoubtedly, Deep Neural Networks(DNNs) make it easier to analyze the nonlinear relationship between various types of users and items. The content-based filtering(CBF) recommends items to users based on their previous preferences, such as actions and explicit feedback. Therefore, it requires an item's features to determine whether an item is similar to the previous one. This makes it possible to recommend user-specific items which the user may prefer explicitly. Also, CBF can capture niche items that very few similar users were interested in. However, there exists a limitation of CBF because of the item's features. All the features are hand-engineered contents, and the features could be hard to distinguish each other if the feature represents generic information such as genre, year of publication, and make. Moreover, the trained model may have limited recommending ability and not be able to extend over the user's potential interests.

For these reasons, collaborative filtering(CF) utilizes similarities between users and items simultaneously; assume that user A might be interested in items that user B prefers. Thus, CF does not depend on the item's description. Instead, the CF models consider a feedback matrix of users(row) and items(column), consisting of the user's explicit preference. The item's feature embedding is made by the feedback matrix, while the size of embedding represents the complexity of the item's feature. Some fundamental CF methods can learn the embedding automatically. Among lots of collaborative filtering methods, Matrix Factorization(MF)[2] regards the feedback matrix $A \in \mathbb{R}^{m\times n}$ as a linear combination of user embedding $U \in \mathbb{R}^{m\times d}$ and item embedding $V \in \mathbb{R}^{n\times d}$, simplify the problem to approximate dot product UV^{T} where m is the number of users, n is the number of items, and d denotes the embedding size. Also, Neural Collaborative Filtering(NCF)[3] overcomes the limitation of linearity of the MF method. Deep Neural Networks(DNNs) for collaborative filtering allow interpreting high-order nonlinearity between user and item and significantly improve the recommendation ability.

However, there exist noisy data in given user and item features for various reason, such as account sharing of hidden users, out-of-date information, and wrong assigned item. Neural Graph Collaborative Filtering(NGCF)[4] shows that using the noisy data feature may corrupt recommendation ability. Instead, NGCF[4] adopted only collaborative information between the user set and the item set by constructing a user-item bipartite graph without any features. It assumes that users having similar preferred items may prefer similar items in the future. Therefore, user embedding and item embedding are derived from the only collaborative graph information containing high-order connectivity, and Graph Convolution Network(GCN) is adopted to analyze the user-item signal. LightGCN[5] is a simplified version of NGCF while still improving the recommending ability. There exist a lot of variants of NGCF such as DGCF[6], LR-GCCF[7], NIA-GCN[8], and UltraGCN[9].

On the other hand, relying on only such collaborative information may not represent a tendency of groups of users and types of items. We can easily imagine that users within a particular group have the same preference for specific items. In this work, we assume that each large dataset can be divided into several user-item groups via Bipartite Co-Clustering, and the collaborative information of a clustered group might be denser than the original dataset. Among various co-clustering algorithms, we adopt the first proposed approach, Spectral Co-Clustering[10], which is described in Section 2. Co-clustering algorithm segments user and item groups simultaneously. The result of Co-clustering is illustrated as a block diagonal matrix as shown in Fig 2.1. Each block represents a user-item group, and this subset contains a strong correlation between data points. Therefore, collaborative filtering in a particular subset works better than entire data.

In this proposed method, each subset is used as a partial input of *local* CF model to deal with only the selected user-item pair. In this way, all subsets are used as inputs for the local model, and all the embedding results from the local collaborative filtering model are aggregated. However, there still exists noisy data outlying from a clustered block. Thus, the local item inputs are extended as all the items which are preferred by the clustered user. The index of each subset is rearranged in the co-clustering step, so we recover the original index in aggregating part. In *global* CF model, the entire data is also used to capture global information such as common preferences and similarities between groups. Finally, weighted summations are conducted for the local and global embedding results. We could improve the performance of NGCF for all datasets using the proposed Local-Ensemble Graph Collaborative Filtering with Co-Clustering(LEGCF). Also, we expect that this approach can utilize any variants of co-clustering and collaborative filtering.

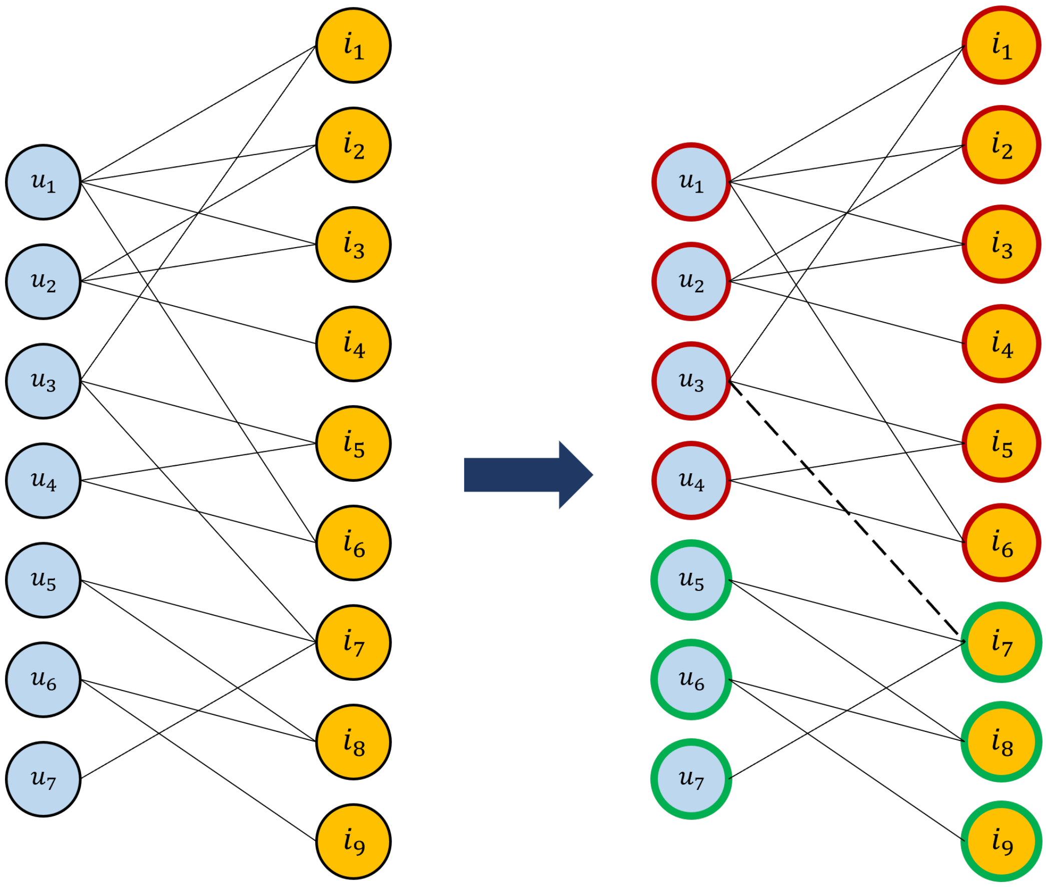


Figure 1.1: User-Item interaction bipartite graph(left) and Spectral Co-Clustering(right)

Chapter 2

Preliminaries

2.1 Spectral Co-Clustering

2.1.1 Bipartite Graph Partitioning

A graph $\mathcal{G} = (V, E)$ is a bipartite graph if there are two disjoint subsets V_i and V_j of V such that

$$V_i \cap V_j = \emptyset, (V_i \times V_i) \cap E = \emptyset, \text{ and } (V_j \times V_j) \cap E = \emptyset \tag{2.1}$$

where V and E is vertex set and edge set, respectively. The definition of cut for the given vertex set V into k subsets is defined by

$$cut(V_1, V_2, \cdots, V_k) = \sum_{i<j} cut(V_i, V_j) = \sum_{i<j} \sum_{i \in V_1, j \in V_2} A_{ij} \tag{2.2}$$

where A_{ij} is an adjacency matrix of the original graph. The example of edge cut is shown in Fig.3.2. We need to cluster both users $U_1, U_2, \cdots, U_k$ and items $I_1, I_2, \cdots, I_k$ which are determined as follows. A given user u_i belongs to the user cluster U_m if its association with the item cluster I_m is greater than its association with any other item cluster, and vice versa.

$$U_m = \{u_i : \sum_{j \in I_m} \mathbf{H}_{ij} \geq \sum_{j \in I_l} \mathbf{H}_{ij}, \forall l = 1, \cdots, k\}$$

$$I_m = \{i_i : \sum_{j \in U_m} \mathbf{H}_{ij} \geq \sum_{j \in U_l} \mathbf{H}_{ij}, \forall l = 1, \cdots, k\}$$

where $\mathbf{H}_{ij}$ is an element of $m \times n$ incidence matrix $\mathbf{H}$ such that equals to the edge-weight E_{ij}. Finding the best clustering corresponds to a partitioning of the graph such that the crossing edges between partitions have minimum weight. This is achieved when

$$cut(U_1 \cup I_1, \cdots, U_k \cup I_k) = \min_{V_1 \cdots V_k} cut(V_1 \cdots V_k)$$

where $V_1 \cdots V_k$ is any k-partitioning of the bipartite graph.

Spectral Graph Bipartitioning

The Laplacian matrix $\mathbf{L}$ of graph $\mathbf{G}$ is an $n \times n$ symmetric matrix and defined by $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{A}$ is the adjacency matrix and $\mathbf{D}$ is the diagonal degree matrix with $D_{ii} = \sum_k E_{ik}$. Given a function on the vertices, $x \in \mathbb{R}^V$, the Laplacian quadratic form of a weighted graph in which edge (a, b) has weight $w_{a,b} > 0$ is

$$\mathbf{x}^{\mathsf{T}}\mathbf{L}\mathbf{x} = \sum_{(a,b) \in E} w_{a,b}(\mathbf{x}(a) - \mathbf{x}(b))^2 \geq 0. \tag{2.3}$$

We can check the first eigenvalue of Laplacian matrix is 0 when we set $\mathbf{x}$ $\in$ $\{n$ mutually orthogonal unit vectors $\boldsymbol{\psi}_1, \cdots \boldsymbol{\psi}_n\}$, then $\boldsymbol{\psi}_k^{\mathsf{T}}\mathbf{L}\boldsymbol{\psi}_k = \lambda_k \geq 0$, and $\mathbf{L1} = 0$.

Given a bipartitioning of V into V_1 and V_2 ($V_1 \cup V_2 = V$), let us define the partition vector $\mathbf{p}$ that captures this division,

$$p_i = \begin{cases} +1, & i \in V_1, \\ -1, & i \in V_2, \end{cases} \tag{2.4}$$

then, the Rayleigh Quotient is

$$\frac{\mathbf{p}^\mathsf{T}\mathbf{L}\mathbf{p}}{\mathbf{p}^\mathsf{T}\mathbf{p}} = \frac{1}{n} \cdot 4cut(V_1, V_2). \tag{2.5}$$

Therefore, the cut is minimized by the trivial solution when all p_i are either -1 or +1. For the generalized form, let each vertex i be associated with a positive weight, denoted by weight(i), and let $\mathbf{W}$ be the diagonal matrix of such weights. Let $\eta_1 = weight(V_1)$ and $\eta_1 = weight(V_2)$, then the generalized partition vector $\mathbf{q}$ with elements

$$q_i = \begin{cases} +\sqrt{\frac{\eta_2}{\eta_1}}, & i \in V_1, \\ -\sqrt{\frac{\eta_2}{\eta_1}}, & i \in V_2, \end{cases} \tag{2.6}$$

satisfies $\mathbf{q}^\mathsf{T}\mathbf{W}\mathbf{e} = 0$, and $\mathbf{q}^\mathsf{T}\mathbf{W}\mathbf{q} = weight(V)$. Then, the generalized Rayleigh Quotient becomes an objective function

$$\frac{\mathbf{q}^\mathsf{T}\mathbf{L}\mathbf{q}}{\mathbf{q}^\mathsf{T}\mathbf{W}\mathbf{q}} = \frac{cut(V_1, V_2)}{weight(V_1)} + \frac{cut(V_1, V_2)}{weight(V_2)}. \tag{2.7}$$

To find the global minimum of the objective function, we can minimize the generalized Rayleigh Quotient

$$\min_{\mathbf{q} \neq \mathbf{0}} \frac{\mathbf{q}^\mathsf{T}\mathbf{L}\mathbf{q}}{\mathbf{q}^\mathsf{T}\mathbf{W}\mathbf{q}} \tag{2.8}$$

when $\mathbf{q}$ is the eigenvector corresponding to the second smallest eigenvalue λ_2 of the generalized eigenvalue problem,

$$\mathbf{L}\mathbf{z} = \lambda \mathbf{W}\mathbf{z}. \tag{2.9}$$

Moreover, we can adopt normalized-cut objective function instead of balanced-cut Eq.(2.7).

$$N(V_1, V_2) = \frac{cut(V_1, V_2)}{\sum_{i \in V_1} \sum_k E_{ik}} + \frac{cut(V_1, V_2)}{\sum_{i \in V_2} \sum_k E_{ik}} = 2 - S(V_1, V_2) \tag{2.10}$$

where $S(V_1, V_2) = \frac{within(V_1)}{weight(V_1)} + \frac{within(V_2)}{weight(V_2)}$.

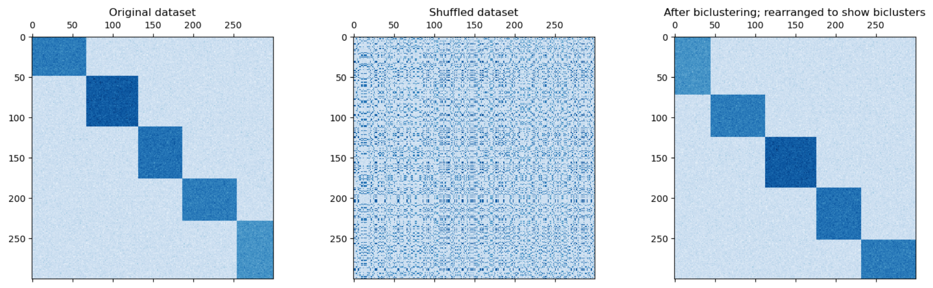


Figure 2.1: An example of Spectral Co-Clustering

2.1.2 Optimization

The generalized eigenvalue problem $\mathbf{Lz} = \lambda\mathbf{Dz}$ provides a real relaxation to the discrete optimization problem of finding the minimum normalized cut.

Computing the eigenvector of the second smallest eigenvalue may cost a lot for a large graph; instead, we can obtain the second-largest singular value $(1 - \lambda)$ by singular value decomposition(SVD) as follows.

For the bipartite graph,

$$L = \begin{bmatrix} \mathbf{D_1} & -\mathbf{A} \\ -\mathbf{A}^\mathsf{T} & \mathbf{D_2} \end{bmatrix}, D = \begin{bmatrix} \mathbf{D_1} & -0 \\ 0 & \mathbf{D_2} \end{bmatrix} \tag{2.11}$$

where $\mathbf{D_1}(i,i) = \sum_j A_{ij}$ and $\mathbf{D_2}(j,j) = \sum_i A_{ij}$ are diagonal matrices. Then $\mathbf{Lz} = \lambda\mathbf{Dz}$ can be rewritten

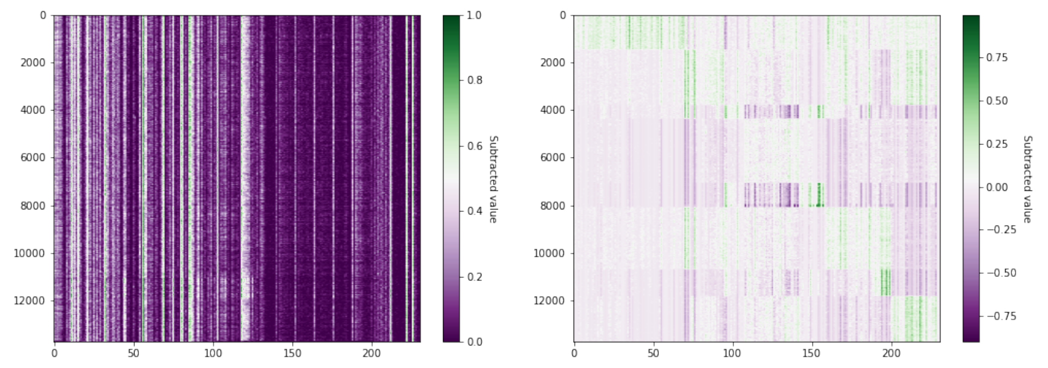
$$\mathbf{D_1x} - \mathbf{Ay} = \lambda\mathbf{D_1x}$$

$$-\mathbf{A}^\mathsf{T}\mathbf{x} + \mathbf{D_2y} = \lambda\mathbf{D_2y}$$

Because the user-item dataset does not include empty columns and rows, both $\mathbf{D_1}$ and $\mathbf{D_2}$ are nonsingular. Then, the above equation

$$\mathbf{D_1}^{1/2}\mathbf{x} - \mathbf{D_1}^{-1/2}\mathbf{Ay} = \lambda\mathbf{D_1}^{1/2}\mathbf{x}$$

$$-\mathbf{D_2}^{-1/2}\mathbf{A}^\mathsf{T}\mathbf{x} + \mathbf{D_2}^{1/2}\mathbf{y} = \lambda\mathbf{D_2}^{1/2}\mathbf{y}$$



(a) Raw incidence matrix of user-item data (b) Clustered incidence matrix of user-item data

Figure 2.2: The result of Co-Clustering can be shown as block-diagonal incidence matrix for real-world data

. Let $\mathbf{u} = \mathbf{D_1}^{1/2}\mathbf{x}$ and $\mathbf{v} = \mathbf{D_2}^{1/2}\mathbf{y}$, we get

$$\mathbf{D_1}^{-1/2}\mathbf{A}\mathbf{D_2}^{-1/2}\mathbf{v} = (1-\lambda)\mathbf{u}, \text{ and } \mathbf{D_2}^{-1/2}\mathbf{A}^\intercal\mathbf{D_1}^{-1/2}\mathbf{u} = (1-\lambda)\mathbf{v}.$$

Therefore, we can compute the left and right singular vectors $\mathbf{u_2}$ and $\mathbf{v_2}$ corresponding to the second largest singular value of normalized matrix $\mathbf{A_n} = \mathbf{D}_1^{-1/2}\mathbf{A}\mathbf{D}_2^{-1/2}$

$$\mathbf{A_n}\mathbf{v_2} = \sigma_2\mathbf{u_2}, \mathbf{A_n}^\intercal\mathbf{u_2} = \sigma_2\mathbf{v_2}, \tag{2.12}$$

where $\sigma_2 = 1 - \lambda_2$. Obviously, working on $\mathbf{A_n}$ rather than $\mathbf{L}$ is beneficial in terms of computational cost since $\mathbf{A} \in \mathbb{R}^{m\times n}$ while $\mathbf{L} \in \mathbb{R}^{(m+n)\times(m+n)}$. The singuluar vector $\mathbf{u_2}$ and $\mathbf{v_2}$ give us clustering information, for users and items, respectively; a real approximation to the discrete optimization problem of minimizing the normalized cut. Let m_1 and m_2 denote the bi-modal values of the assignment of $\mathbf{z}_2(i)$ that we are looking for; the second eigenvector of $\mathbf{L}$ is given by

$$\mathbf{z}_2 = \begin{bmatrix}\mathbf{D_1}^{-1/2}\mathbf{u_2}\\ \mathbf{D_2}^{-1/2}\mathbf{v_2}\end{bmatrix}. \tag{2.13}$$

We can approximate the optimal bipartitioning by minimizing the sum-

Clustered	**Original Dataset**			
Dataset	#User	#Items	#Interactions	Sparsity
Amazon-Book	52,643	91,559	2,984,108	0.00062
Yelp2018	31,668	38,048	1,561,406	0.00130
Gowalla	29,858	40,981	1,027,370	0.00084
Movielens-1M	6,022	3,043	995,154	0.04888
Amazon-CDs	43,169	35,648	777,426	0.00051
Amazon-Electronics	1,435	1,522	35,931	0.01645

Table 2.1: Statistics of the original datasets.

of-squares criterion as an objective function,

$$\sum_{j=1}^{2} \sum_{\mathbf{z}_2(i) \in m_j} (\mathbf{z}_2(i) - m_j)^2.$$

Note that the above equation is exactly the same as the k-means clustering algorithm. Furthermore, we can extend to a multi-partitioning problem to get k multiple clusters by using the bi-partitioning algorithm in a recursive manner. To obtain k-modal information about the dataset, we can form a l-dimensional dataset

$$\mathbf{Z} = \begin{bmatrix} \mathbf{D_1}^{-1/2}\mathbf{U} \\ \mathbf{D_2}^{-1/2}\mathbf{V} \end{bmatrix} \tag{2.14}$$

where $\mathbf{U} = [\mathbf{u_2}, \mathbf{u_3}, \cdots, \mathbf{u_{l+1}}]$ and $\mathbf{V} = [\mathbf{v_2}, \mathbf{v_3}, \cdots, \mathbf{v_{l+1}}], l = [\log_2 k]$, and get the best l-dimensional point $\mathbf{m_1}, \cdots \mathbf{m_k}$ of assignment $\mathbf{Z}(i)$ by minimizing

$$\sum_{j=1}^{k} \sum_{\mathbf{z}_2(i) \in m_j} \|\mathbf{Z}(i) - m_j\|^2. \tag{2.15}$$

2.2 Bayesian Personalized Ranking(BPR) Loss

2.2.1 Implicit Data

In the feedback matrix(or incidence matrix) of the recommendation system, the users' interests are explicitly indicated as binary scores except in the case a user expresses explicit and negative feedback, e.g., 1 of 5 stars for a movie. The incidence matrix is a sparse matrix, and obviously, the non-positive scores do not indicate apparent negative action since the scores for most of the items are not yet defined. The *implicit data* contains *missing value*, and the purpose of the recommendation system is to determine whether the implicit data is real negative feedback or potentially positive interest.

2.2.2 Personalized Total Ranking

Let U be the set of all users, I the set of all items, and $(u, i) = s \in S$ the user-item pair where $S \subseteq U \times I$ is the set of implicit feedback. Our goal is to provide the user with a personalized total ranking, $>_u \subset I^2$, where $>_u$ meets the properties of a total order as follows.

$$\forall i, j \in I : i \neq j \implies (i >_u j) \vee (j >_u i) \tag{2.16}$$

$$\forall i, j \in I : (i >_u j) \wedge (j >_u i) \implies i = j \tag{2.17}$$

$$\forall i, j, k \in I : (i >_u j) \wedge (j >_u k) \implies i >_u k. \tag{2.18}$$

In implicit feedback systems, only positive data is observed, and the remaining data is a mixture of missing and actually negative values. Instead of replacing missing values with negative ones for single items, the item pairs are used as training data. To reconstruct user-item pairs, we follow a basic rule for the given data: the user prefers a positively observed item to all other non-observed ones, and there's no priority among observed items. Therefore, we define the training dataset $D_S : U \times I \times I$ as

$$D_S := \{(u, i, j) | i \in I_u^+ \wedge j \in I \backslash I_u^+\} \tag{2.19}$$

Statistics		Dataset					
# Clusters		Amazon-Book	Yelp2018	Gowalla	Movielens -1M	Amazon -CDs	Amazon -Elec.
1	Users	52,643	31,668	29,558	6,022	43,169	1,435
	Sparsity	0.00062	0.00130	0.00084	0.04888	0.00051	0.01645
2	Min.	1,379	5,632	8,810	2,305	1,491	262
	Max.	51,264	26,036	21,048	3,717	41,678	1,173
	Sparsity	0.00085	0.00202	0.00143	0.04894	0.00061	0.01740
3	Min.	258	1,681	338	1,629	1,488	138
	Max.	51,093	24,389	21,529	2,437	36,557	1,080
	Sparsity	0.00098	0.00259	0.00179	0.04915	0.00074	0.01836
4	Min.	215	1,665	333	869	1,445	36
	Max.	50,736	20,743	21,108	1,754	23,444	899
	Sparsity	0.00107	0.00304	0.00195	0.04970	0.00084	0.01920
5	Min.	216	609	115	827	31	41
	Max.	29,557	21,084	19,747	1,498	26,492	614
	Sparsity	0.00123	0.00353	0.00244	0.05023	0.00106	0.01979

Table 2.2: Statistics of the partitioned dataset about the minimum and maximum number of users and sparsity.

where user u of $(u, i, j) \in D_S$ is assumed to prefer i over j, and

$$I_u^+ := \{i \in I : (u, i) \in S\} \tag{2.20}$$

$$U_i^+ := \{u \in U : (u, i) \in S\} \tag{2.21}$$

D_S allows us the training dataset consists of both positive and negative pairs and missing values. The missing values of pairs of non-observed items are exactly our objective item pairs that would be ranked in the training procedure. The observed subset D_S of $>_u$ is used as training data, while D_S and the test data is disjoint.

2.2.3 Bayesian Personalized Ranking

As a general optimization criterion for personalized ranking, *Bayesian Personalized Ranking*(BPR) optimization [11] will be derived by a Bayesian analysis of the problem using the likelihood function for $p(i >_u j|\Theta)$ and the prior probability for the model parameter $p(\Theta)$. Finding the correct personalized ranking for all items $i \in I$ in terms of Bayesian formulation is equivalent to maximizing the following posterior probability

$$p(\Theta| >_u) \propto p(>_u |\Theta)p(\Theta) \tag{2.22}$$

where Θ represents the parameter vector of a model.
As the user-specific likelihood function $p(>_u |\Theta)$ follows Bernoulli distribution, it can be expressed as a product of single densities, and combined for all users.

$$p(>_u |\Theta) = p(i >_u j)^{\delta((u,i,j)\in D_S)}(1 - p(i >_u j))^{\delta((u,i,j)\notin D_S)}, \tag{2.23}$$

where δ is the indicator function

$$\delta(b) := \begin{cases} 1 & \text{if } \ b \ \text{ is true} \\ 0 & \text{else.} \end{cases} \tag{2.24}$$

To fulfill the properties of a total order(totality, antisymmetry, and transitivity), the individual probability that a user really prefers item i to item j is defined as :

$$p(i >_u j|\Theta) := \sigma(\hat{x}_{uij}(\Theta)) \tag{2.25}$$

where σ is the logistic sigmoid $\sigma(x) := \frac{1}{1+e^{-x}}$ and $\hat{x}_{uij}(\Theta)$ is obtained model parameter vector to capture the relationship between user u, item i, and item j. For all users, the overall likelihood function is

$$\prod_{u\in U} p(>_u |\Theta) = \prod_{(u,i,j)\in D_S} p(i >_u j|\Theta) \tag{2.26}$$

$$= \prod_{(u,i,j)\in U\times I\times I} p(i >_u j|\Theta)^{\delta((u,i,j)\in D_S)}(1 - p(i >_u j|\Theta))^{\delta((u,i,j)\notin D_S)}. \tag{2.27}$$

To assign above likelihood function to Bayesian personalized ranking task, let $p(\Theta) \sim N(0, \sum_\Theta)$ where $\sum_\Theta = \lambda_\Theta I$ while λ_Θ are model specific regularization parameter. Then, the maximum posterior estimator to derive optimization criterion for personalized ranking $\mathtt{BPR} - \mathtt{OPT}$ is

$$\begin{aligned}
\mathtt{BPR} - \mathtt{OPT} :&= \ln p(\Theta| >_u) \\
&= \ln p(>_u |\Theta) p(\Theta) \\
&= \ln \prod_{(u,i,j)\in D_S} \sigma(\hat{x}_{uij} p(\Theta)) \\
&= \sum_{(u,i,j)\in D_S} \ln \sigma(\hat{x}_{uij}) + \ln p(\Theta) \\
&= \sum_{(u,i,j)\in D_S} \ln \sigma(\hat{x}_{uij}) - \lambda_\Theta \|\Theta\|^2.
\end{aligned} \tag{2.28}$$

The *Bayesian Personalized Ranking*(BPR) optimization is adopted as a loss function to minimize. BPR loss is a pairwise loss that encourages following the total order properties:

$$L_{BPR} = -\sum_{u=1}^{M} \sum_{i\in\mathcal{N}_u} \sum_{j\notin\mathcal{N}_u} \ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda \|\mathsf{E}^{(0)}\|^2. \tag{2.29}$$

Chapter 3

Proposed Method

3.1 Dataset

We use six public datasets, including Amazon-Book[12], Amazon-CDs, Amazon-Electronics, Yelp2018, Gowalla[13], and MovieLens-1M, to verify our approach. Many recent graph-based CF methods are evaluated on these six datasets. The original datasets provide various information for users and items such as gender, age, genre, and date. However, these features are not necessary for the Graph-based recommendation, and only the interaction data is used. We follow those recent studies and use the same data split ratio for the training subset and test subset without any detailed information. Table 2.1 shows the statistics of the datasets.

3.2 Spectral Co-Clustering

We assume that a large dataset of user-item interactions consists of several clusters which have a strong user-item relationship in each cluster. We can visualize this by comparing incidence matrices of original data and clustered data.

As shown in Fig2.1, the strong connectivity is shown as a block-diagonal matrix by rearranging indices of rows and columns. Clustered user set u_k and item set i_k is defined by

$$\mathbf{U} = [u_1|u_2|\cdots|u_k]$$

$$\mathbf{I} = [i_1|i_2|\cdots|i_k]$$

where $\mathbf{U}$ and $\mathbf{I}$ mean total users and items, respectively. However, the incidence matrix is not partitioned perfectly since commonly preferred items or noisy data exist. The preferred item set i_{u_k} for given user group u_k does not correspond to k-th item set i_k. Therefore, we utilize i_{u_k} for training each local CF instead of the clustered item set $[i_1|i_2|\cdots|i_k]$ to avoid absence of preferred item. Nevertheless, the co-clustered user subset is still meaningful because the partitions refer to the collaborative information of user-item interaction.

Table 2 shows that the clustered dataset contains denser information than the original data. Although the scales of clustered subsets vary, the more we divide the dataset, we obtain less sparse collaborative data. Currently, we select k empirically by grid search. Finding the particular best k might be impossible because it significantly depends on the co-clustering method and dataset. We assume that we can select proper k by evaluating the number of small eigenvalues of the adjacency matrix but will keep this study as future work. Among various co-clustering methods, we adopt one of the fundamental co-clustering methods, spectral co-clustering, to show the improvement of our approach. Our approach can be combined with other Co-Clustering variants such as CCMOD[14], SCMK[15], ONMTF[16], and SOBG[17].

3.3 Local-Ensemble model

We conducted collaborative filtering utilizing clustered data. We abandon the clustered item subset i_k, and instead obtain item subset i_{u_k} from user subset u_k. We execute the CF method on original data and clustered data simultaneously, regarding original data as global information and clusters as local information, as shown in Fig. 4. Each local embedding $e_{i,k}, e_{u,k}$ have different index information and embedding size according to k, so they are aggregated as same size as global embedding with $e_{i,l}, e_{u,l} \in \mathbb{R}^{n \times m \times d}$ as shown in Fig. 5. The final local embedding and global embedding are merged at the tail part of the framework with an element-wise attention score on local embedding

$$e_i = e_{i,l} \odot w_i + e_{i,g} \tag{3.1}$$

$$e_u = e_{u,l} \odot w_u + e_{u,g} \tag{3.2}$$

where $\odot$ denotes element-wise multiplication, and w_i, w_u are trainable weight. The overall algorithm is shown in Algorithm.1.

We selected the hyperparameters of the original literature, such as message dropout rate, embedding size, batch size, regularizer, and learning rate. Also, we adopt the Bayesian Personalized Ranking(BPR) loss function following the description of NGCF

$$Loss = \sum_{(u,i,j) \in \mathcal{O}} - \ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda \|\Theta\|_2^2 \tag{3.3}$$

where $\mathcal{O} = \{(u,i,j)|(u,i) \in \mathcal{R}^+, (u,j) \in \mathcal{R}^-\}$ denotes the pairwise training data, $\mathcal{R}^+$ indicates the observed interactions, and $\mathcal{R}^-$ is the unobserved interactions, $\sigma(\cdot)$ is the sigmoid function, and $\Theta = \{\mathbf{E}, \{\mathbf{W}_1^{(l)}, \mathbf{W}_2^{(l)}\}_{l=1}^{L}\}$ denotes all trainable model parameters, and λ controls the L_2 regularization strength to prevent overfitting.

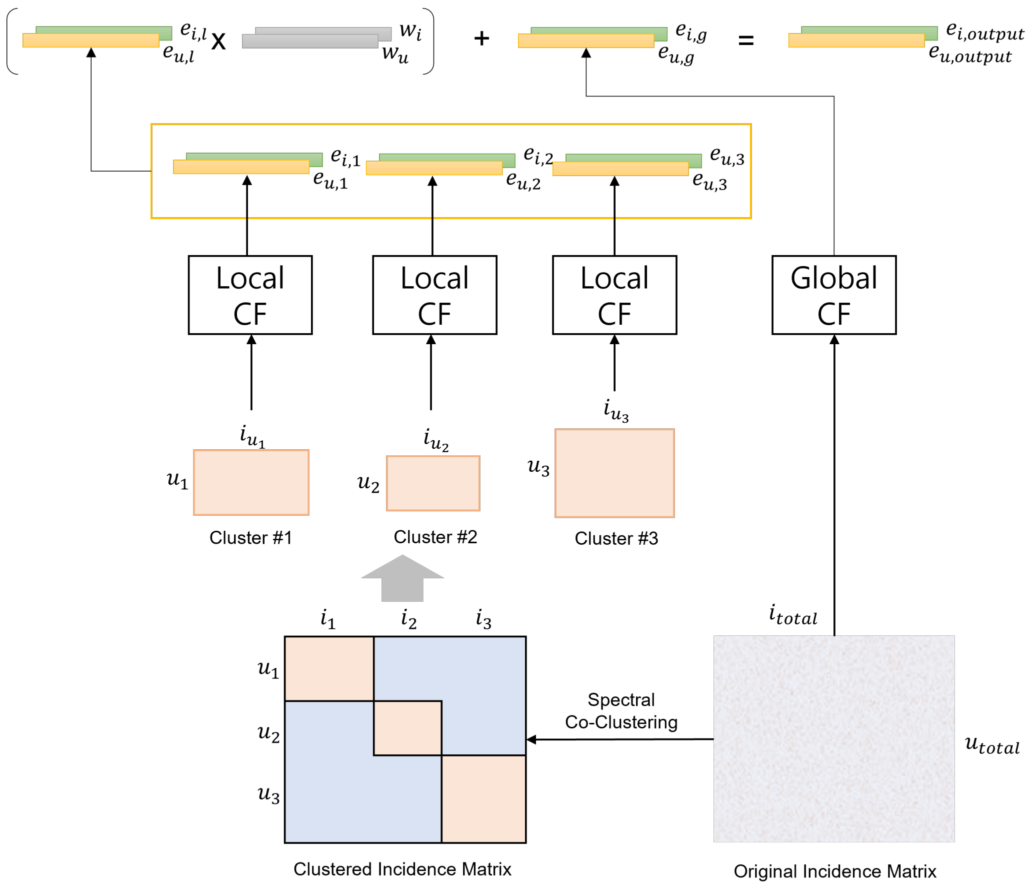


Figure 3.1: Overall Architecture of Separated Collaborative Filtering.

.

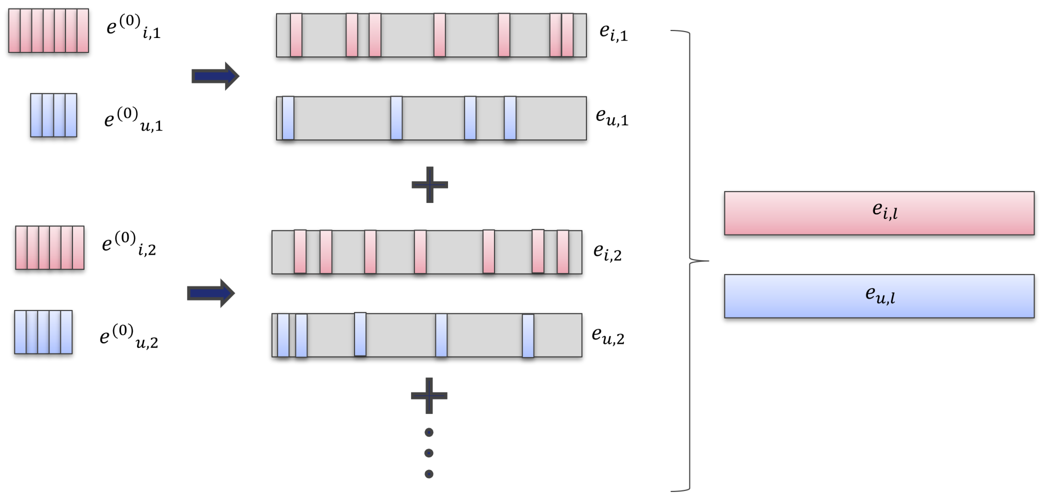


Figure 3.2: Local Embeddings are aggregated again refer to their initial cluster mapping

Algorithm 1 Overall Architectures of Local-Ensemble Collaborative Filtering

Input: Rating Matrix R, the number of cluster N

Output: User embedding e_u, Item embedding e_i

```
 1: $\mathcal{R}_l = \{R_1, \cdots, R_k\} = \texttt{SCC}(R)$
 2: for $k = 0$ to $N$ do
 3:     $A_k = \begin{bmatrix} 0 & R_k \\ R_k^\top & 0 \end{bmatrix}$
 4:     $e_{u,k}, e_{i,k} \leftarrow \texttt{CF}_k(A_k)$ {Global embedding when $k = 0$ }
 5:     if $k \geq 1$ then
 6:         $\hat{e}_{u,k} = \texttt{IndexMapping}(e_{u,k})$
 7:         $\hat{e}_{i,k} = \texttt{IndexMapping}(e_{i,k})$
 8:     end if
 9: end for
10: $e_{u,l} = \sum_{k=1}^{N} \hat{e}_{u,k}$ {Local user embedding}
11: $e_{i,l} = \sum_{k=1}^{N} \hat{e}_{i,k}$ {Local item embedding}
12: $e_u = e_{u,0} + e_{u,l} \odot w_u$
13: $e_i = e_{i,0} + e_{i,l} \odot w_i$
```

Chapter 4

Experimental Result

4.1 Evaluation Metric

To evaluate the ability of the proposed method, *Normalized Discounted Cumulative Gain* (NDCG)@K[18, 19], Precision@K, and Recall@K are used. These metrics are widely used to measure the performance of recommendation system. Above all, NDCG metric considers graded relevance allowing higher position in the ouput list gets more importance. Discounted Cumulative Gain(DCG) for a particular rank position p is obtained by

$$\mathrm{DCG}_p = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)} \tag{4.1}$$

where rel_i is the *grade relevance* of the result at position i such that $rel_i = 1$ if the i-th output item is actually relevant to the query, $rel_i = 0$ otherwise. The alternative formulation of DCG emphasizes the relevance by replacing numerator to exponential term.

$$\mathrm{DCG}_p = \sum_{i=1}^{p} \frac{2^{rel_i}}{\log_2(i+1).} \tag{4.2}$$

The two expressions of DCG are same when the relevance values are binary, $rel_i \in \{0, 1\}$. The DCG score means that highly relevant items appearing lower rank should be penalized as the graded relevance reduces. Maximum possible DCG depends on how many relevant items are in ground truth.

On the other hand, NDCG is the normalized score which is obtained by the ratio between DCG_p and IDCG_p for that query.

$$\text{NDCG}_p = \frac{\text{DCG}_p}{\text{IDCG}_p} \tag{4.3}$$

where ideal discounted cumulative gain(IDCG) is the maximum score when the items are listed in the order of relevance, REL_p.

$$\text{IDCG}_p = \sum_{i=1}^{|REL_p|} \frac{2^{rel_i}}{\log_2(i+1).} \tag{4.4}$$

In addition, Precision@K and Recall@K is calculated by the fraction of K recommended items, regardless of the order.

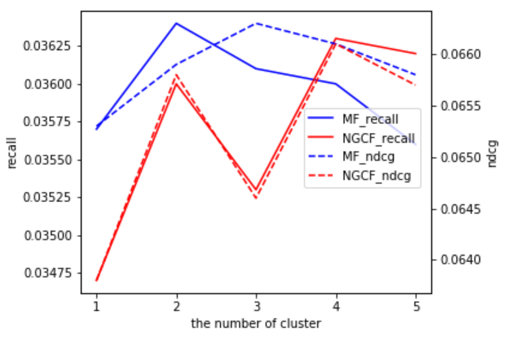
$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \tag{4.5}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \tag{4.6}$$

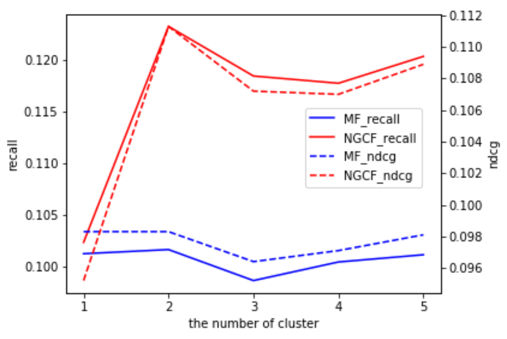
4.2 Result Analysis

The result of the experiments is shown in Table 3-5. The experiments are conducted on six datasets and three baseline models, MF-BPR, NGCF, and LightGCN. For Amazon-Book, Yelp2018, Amazon-CDs, and Amazon-Electronics dataset, our proposed method improve the recommendation ability for all baseline models. Fig.6 shows the plotted result of the above four datasets. Although the performances depend on the number of clusters k, we can get a better result using the local-ensemble method compared to the single CF model, $k = 1$.

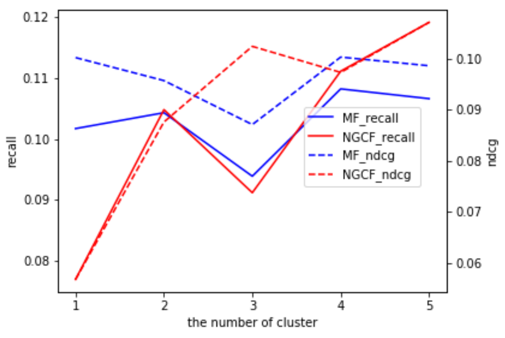
On the other hand, the proposed method could not enhance the recommendation ability for some datasets, such as the Movielens-1M and Gowalla datasets. To verify this, we visualized UMAP[20] projection and checked the connectivity. When the data points are uniformly distributed, such as Amazon-book, Amazon-CDs, and Amazon-Electronics datasets, Spectral Co-Clustering can reveal the locality of each dataset, and the performances were improved by the local-ensemble approach. On the other hand, unlike our expectation, the proposed method does not enhance the recommendation ability much for well-clustered data. Although Yelp2018 and Gowalla datasets are well separated into a few clusters, they still have many connections between different groups. We suppose that there exist important few relations between user groups, and ignoring these signals could worsen the recommendation ability.



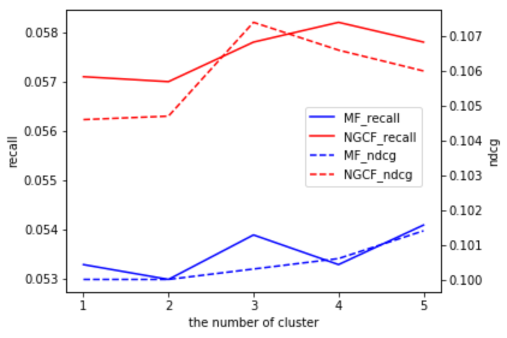
(a) Amazon-Book



(b) Amazon-CDs



(c) Amazon-Electronics



(d) Yelp2018

Figure 4.1: Evaluation Result for four datasets with MF-BPR and NGCF model.

Cases		Base	2 Clusters	3 Clusters	4 Clusters	5 Clusters	Best	Diff.
Amazon-Book	Recall	0.0357	**0.0364**	0.0361	0.0360	0.0356	0.0364	0.07%
	NDCG	0.0653	0.0659	**0.0663**	0.0661	0.0658	0.0663	0.1%
Yelp2018	Recall	0.0533	0.0530	0.0539	0.0533	**0.0541**	0.0541	0.08%
	NDCG	0.1000	0.1000	0.1003	0.1006	**0.1014**	0.1014	0.14%
Gowalla	Recall	0.1564	0.1549	0.1561	0.1551	**0.1564**	0.1564	-
	NDCG	0.2260	0.2263	**0.2267**	0.2255	0.2268	0.2267	0.07%
Movielens-1M	Recall	**0.2485**	0.2464	0.2471	0.2476	0.2483	0.2485	-
	NDCG	0.3047	0.3034	0.3034	0.3046	**0.3058**	0.3058	0.11%
Amazon-CDs	Recall	0.1013	**0.1017**	0.0987	0.1005	0.1012	0.1017	0.04%
	NDCG	0.0983	**0.0983**	0.0964	0.0971	0.0981	0.0983	-
Amazon-Electronics	Recall	0.1017	0.1043	0.0939	**0.1082**	0.1066	0.1082	0.65%
	NDCG	0.1002	0.0957	0.0871	**0.1003**	0.0986	0.1003	0.01%

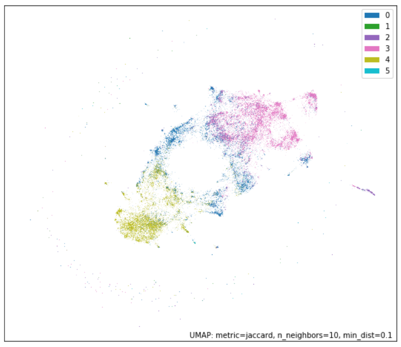
Table 4.1: Experimental result for MF-BPR(Recall@20 and NDCG@20.)

Cases		Base	2 Clusters	3 Clusters	4 Clusters	5 Clusters	Best	Diff.
Amazon-	Recall	0.0347	0.0360	0.0353	**0.0363**	0.0362	0.0363	0.16%
Book	NDCG	0.0638	0.0658	0.0646	**0.0661**	0.0657	0.0661	0.23%
Yelp2018	Recall	0.0571	0.0570	0.0578	**0.0582**	0.0578	0.0582	0.09%
	NDCG	0.1046	0.1047	**0.1074**	0.1066	0.1060	0.1074	0.28%
Gowalla	Recall	0.1633	0.1637	0.1638	**0.1641**	0.1625	0.1641	0.08%
	NDCG	0.2313	0.2330	**0.2353**	0.2346	0.2341	0.2353	0.4%
Movielens	Recall	0.2509	0.2518	0.2535	**0.2542**	0.2533	0.2542	0.33%
-1M	NDCG	0.3047	0.3088	0.3080	0.3091	**0.3095**	0.3095	0.48%
Amazon-	Recall	0.1024	**0.1233**	0.1185	0.1178	0.1204	0.1233	2.09%
CDs	NDCG	0.0952	**0.1113**	0.1072	0.1070	0.1089	0.1113	1.61%
Amazon-	Recall	0.0770	0.1048	0.0912	0.1111	**0.1191**	0.1191	4.21%
Electronics	NDCG	0.0568	0.0876	0.1024	0.0973	**0.1071**	0.1071	5.03%

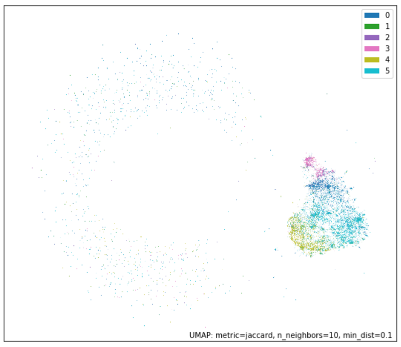
Table 4.2: Experimental result for NGCF(Recall@20 and NDCG@20.)

Cases		Base	2 Clusters	3 Clusters	4 Clusters	5 Clusters	Best	Diff.
Amazon-	Recall	0.0397	0.0400	0.0397	**0.0404**	0.0397	0.0404	0.07%
Book	NDCG	0.0163	0.0165	0.0163	**0.0165**	0.0163	0.0165	0.02%
Yelp2018	Recall	0.0574	0.0573	0.0575	0.0574	**0.0576**	0.0576	0.02%
	NDCG	0.1052	0.1059	**0.1061**	0.1059	0.1056	0.1061	0.09%
Gowalla	Recall	**0.1624**	0.1614	0.1613	0.1618	0.1616	0.1624	-
	NDCG	0.2270	**0.2279**	0.2257	0.2242	0.2264	0.2279	0.09%
Movielens	Recall	**0.2547**	0.2536	0.2521	0.2539	0.2521	0.2547	-
-1M	NDCG	**0.3081**	0.3071	0.3053	0.3062	0.3050	0.3081	-
Amazon-	Recall	0.1447	0.1442	0.1453	**0.1457**	0.1447	0.1457	0.1%
CDs	NDCG	0.1316	0.1314	0.1312	**0.1324**	0.1321	0.1324	0.08%
Amazon-	Recall	0.1305	0.1267	0.1308	**0.1332**	0.1269	0.1332	0.27%
Electronics	NDCG	0.1157	0.1161	0.1216	**0.1250**	0.1149	0.1250	0.93%

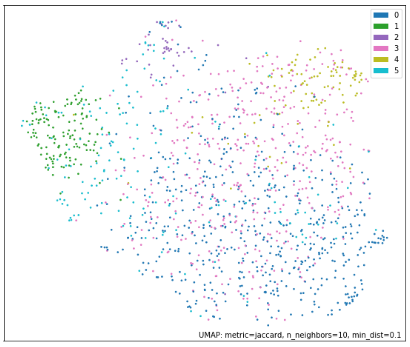
Table 4.3: Experimental result for LightGCN(Recall@20 and NDCG@20.)



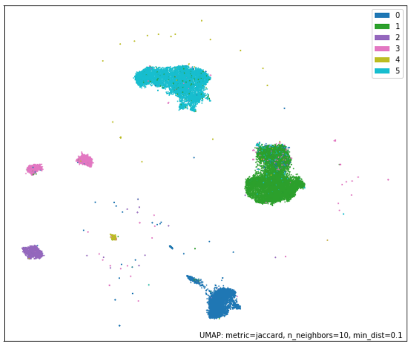
(a) Amazon-Book



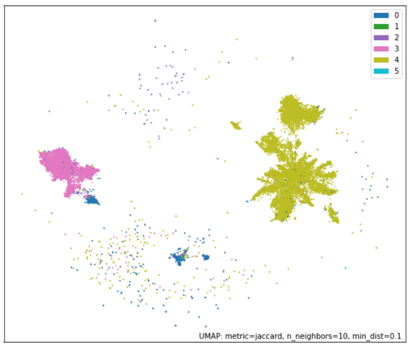
(b) Amazon-CDs



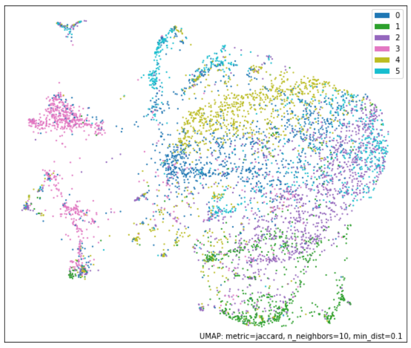
(c) Amazon-Elec.



(d) Yelp2018

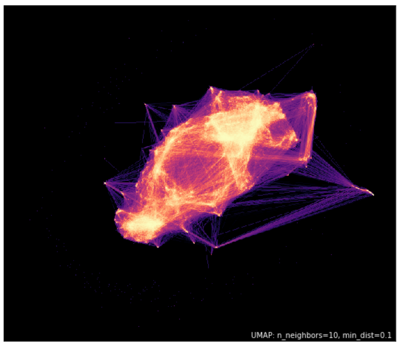


(e) Gowalla

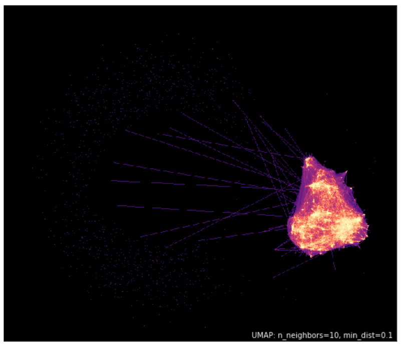


(f) MovieLens-1M

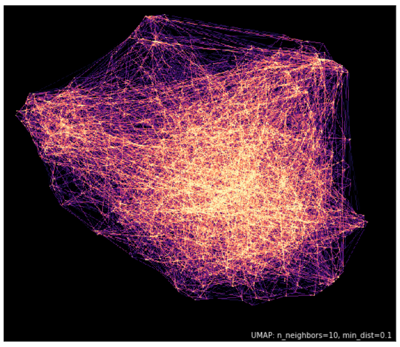
Figure 4.2: Visualization result of each dataset using UMAP.



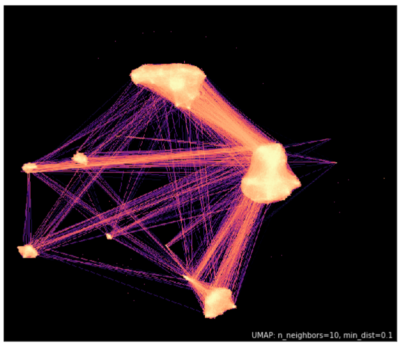
(a) Amazon-Book



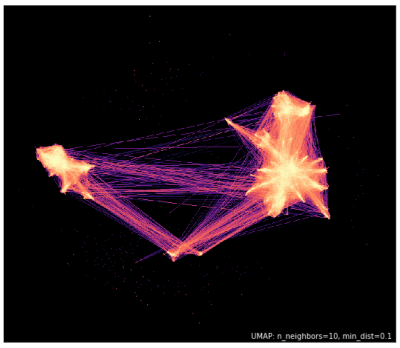
(b) Amazon-CDs]



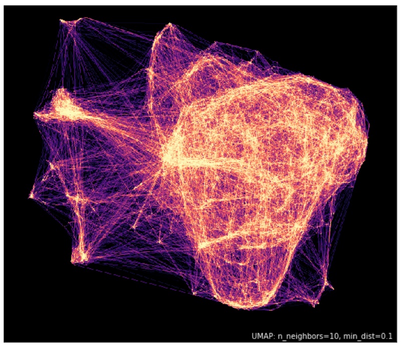
(c) Amazon-Elec.



(d) Yelp2018



(e) Gowalla



(f) MovieLens-1M

Figure 4.3: Connectivity Visualization result of each dataset using UMAP.

Chapter 5

Conclusion

The experimental results and visualization show that Spectral Co-Clustering can separate a given dataset as bi-partition well, and the collaborative filtering within a specific co-cluster can utilize strong collaborative information in spite of the unbalanced clustering result of Spectral Co-Clustering. However, the Recall@20 and NDCG@20 scores indicate that our approach cannot cover all datasets and CF models. Also, the evaluation results are not significantly improved. On the other hand, we can easily expect that well-clustered co-clustering such as `CoclustMod` and ONMTF can improve the result. Moreover, we chose the hyperparameter k empirically, but finding proper k with the number of small eigenvalues of graph or using ensemble-coclustering[21] may help our model to find a way to better performance.

There are many opportunities to obtain the best performance by adopting our approach when it combines state-of-the-art collaborative filtering models and modifying the aggregation part better. We will keep these studies for future work as below.

- We will modify the entire unsupervised clustering and semi-supervised training process in an end-to-end manner.

- The clustered groups refer to 'local information,' while the original data refer to 'global information.' We may build a whole new architecture that combines local and global information to avoid information loss between user groups while utilizing strong local information.

- We should implement more CF models such as UltraGCN[9], ENMF[22], GCMC[23], and so on.
- Build a mathematical foundation finding proper k by spectral theory or probabilistic graphical models.

Bibliography

[1] P. B. Thorat, R. Goudar, and S. Barve, “Survey on collaborative filtering, content-based filtering and hybrid recommendation system,” *International Journal of Computer Applications*, vol. 110, no. 4, pp. 31–36, 2015.

[2] Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[3] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural collaborative filtering,” in *Proceedings of the 26th international conference on world wide web*, pp. 173–182, 2017.

[4] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, “Neural graph collaborative filtering,” in *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, pp. 165–174, 2019.

[5] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, “Lightgcn: Simplifying and powering graph convolution network for recommendation,” in *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pp. 639–648, 2020.

[6] X. Wang, H. Jin, A. Zhang, X. He, T. Xu, and T.-S. Chua, “Disentangled graph collaborative filtering,” in *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, pp. 1001–1010, 2020.

[7] L. Chen, L. Wu, R. Hong, K. Zhang, and M. Wang, "Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, pp. 27–34, 2020.

[8] J. Sun, Y. Zhang, W. Guo, H. Guo, R. Tang, X. He, C. Ma, and M. Coates, "Neighbor interaction aware graph convolution networks for recommendation," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1289–1298, 2020.

[9] K. Mao, J. Zhu, X. Xiao, B. Lu, Z. Wang, and X. He, "Ultragcn: Ultra simplification of graph convolutional networks for recommendation," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 1253–1262, 2021.

[10] I. S. Dhillon, "Co-clustering documents and words using bipartite spectral graph partitioning," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 269–274, 2001.

[11] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, (Arlington, Virginia, USA), p. 452–461, AUAI Press, 2009.

[12] R. He and J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *proceedings of the 25th international conference on world wide web*, pp. 507–517, 2016.

[13] D. Liang, L. Charlin, J. McInerney, and D. M. Blei, "Modeling user exposure in recommendation," in *Proceedings of the 25th international conference on World Wide Web*, pp. 951–961, 2016.

[14] M. Ailem, F. Role, and M. Nadif, "Co-clustering document-term matrices by direct maximization of graph modularity," in *Proceedings of the*

24th ACM international on conference on information and knowledge management, pp. 1807–1810, 2015.

[15] Z. Kang, C. Peng, and Q. Cheng, "Twin learning for similarity and clustering: A unified kernel approach," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.

[16] H. Abe and H. Yadohisa, "Orthogonal nonnegative matrix tri-factorization based on tweedie distributions," *Advances in Data Analysis and Classification*, vol. 13, no. 4, pp. 825–853, 2019.

[17] F. Nie, X. Wang, C. Deng, and H. Huang, "Learning a structured optimal bipartite graph for co-clustering," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[18] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of ir techniques," *ACM Transactions on Information Systems (TOIS)*, vol. 20, no. 4, pp. 422–446, 2002.

[19] X. He, T. Chen, M.-Y. Kan, and X. Chen, "Trirank: Review-aware explainable recommendation by modeling aspects," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pp. 1661–1670, 2015.

[20] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[21] S. Affeldt, L. Labiod, and M. Nadif, "Ensemble block co-clustering: a unified framework for text data," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 5–14, 2020.

[22] C. Chen, M. Zhang, C. Wang, W. Ma, M. Li, Y. Liu, and S. Ma, "An efficient adaptive transfer neural network for social-aware recommendation," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 225–234, ACM, 2019.

[23] R. v. d. Berg, T. N. Kipf, and M. Welling, “Graph convolutional matrix completion,” *arXiv preprint arXiv:1706.02263*, 2017.

국문초록

본 논문에서는 추천시스템을 위한 그래프 기반 협업 필터링 모델을 스펙트럴 이중 분할하여 생성된 부분 그래프를 앙상블(Ensemble) 하여 추천 성능을 개선하는 방법에 대해 연구하였다. 그래프 인공 신경망 (Graph Neural Networks, GNN)을 이용한 협업 필터링 기반의 추천 시스템의 기본 모델은, 사용자나 아이템에 대한 사전 정보를 전혀 사용하지 않고 사용자-아이템 간 상호작용 정보만을 활용하여 신경망 모델의 임베딩을 구성한다. 따라서 사용자와 아이템의 사전정보만으로 유추할 수 있는 특정 사용자 그룹의 경향성을 추천시스템에 사용할 수 없는 단점이 있다. 한편, 스펙트럴 이중 분할 방법은 특잇값 분해를 반복하여 이분 그래프를 양 도메인의 정보를 모두 포함한 부분 그래프로 분할한다. 추천시스템을 위한 데이터 세트를 스펙트럴 이중 분할 할 경우, 특정 사용자그룹과 아이템 그룹을 전체 데이터로부터 분리할 수 있으며, 분할된 그룹은 높은 데이터 밀도와 강한 상호작용 신호를 갖게 된다. 따라서 분할된 그룹 데이터에 대해 협업 필터링을 적용할 경우, 데이터 세트나 협업 필터링 모델의 종류와 관계없이 해당 데이터그룹에서는 추천 능력이 향상된다. 나아가서, 분할된 부분 그래프들을 개별적으로 협업 필터링한 뒤 앙상블 하여 그룹별 상호작용 신호를 분석한 지역임베딩(Local Embedding)과 전체 데이터를 아우를 수 있는 전역임베딩(Global Embedding)을 통합하여 최종임베딩을 구성하였다. 여섯 개의 데이터 세트와 세 가지의 협업 필터링 모델을 스펙트럴 이중 분할하여 앙상블 한 결과, 모델 종류와 관계없이 추천 능력이 향상되었다. 그러나 몇 가지 데이터 세트의 경우 성능향상이 거의 이루어지지 않았는데, 이는 데이터가 이미 적절히 분할되어있는 경우 스펙트럴 이중 분할이 추천 성능을 향상하지 못한 것으로 분석된다. 반면 데이터가 골고루 분포되어있어 기본 협업 필터링 모델이 상호작용 신호를 분석하기 어려운 경우, 스펙트럴 이준 분할을 통한 앙상블 방법으로 모든 협업 필터링 모델에 대하여 추천 능력이 향상되었다.

주요어 : 추천 시스템, 그래프 인공 신경망, 협업 필터링, 스펙트럴 이중 분할
학번 : 2020–21619

Dedication

I dedicate this thesis to my loving wife,
Jae Eun Park.