Geosocial Co-Clustering: A Novel Framework for Geosocial Community Detection

JUNGEUN KIM, Kongju National University
JAE-GIL LEE*, Korea Advanced Institute of Science and Technology
BYUNG SUK LEE, University of Vermont
JIAJUN LIU, Renmin University

As location-based services using mobile devices have become globally popular these days, social network analysis (especially, community detection) increasingly benefits from combining social relationships with geographic preferences. In this regard, this paper addresses the emerging problem of geosocial community detection. We first formalize the problem of *geosocial co-clustering*, which co-clusters the users in social networks and the locations they visited. Geosocial co-clustering detects higher-quality communities than existing approaches by improving the *mapping clusterability*, whereby users in the same community tend to visit locations in the same region. While geosocial co-clustering is soundly formalized as *non-negative matrix tri-factorization*, conventional matrix tri-factorization algorithms suffer from a significant computational overhead when handling large-scale data sets. Thus, we also develop an efficient framework for geosocial co-clustering, called *GEOsocial COarsening and DEcomposition (GEOCODE)*. In order to achieve efficient matrix tri-factorization, GEOCODE reduces the numbers of users and locations through coarsening and then decomposes the single whole matrix tri-factorization into a set of multiple smaller sub-matrix tri-factorizations. Thorough experiments conducted using real-world geosocial networks show that GEOCODE reduces the elapsed time by 19–69 times while achieving the accuracy of up to 94.8% compared with the state-of-the-art co-clustering algorithm. Furthermore, the benefit of the mapping clusterability is clearly demonstrated through a local expert recommendation application.

CCS Concepts: • **Information systems** → Data mining;

Additional Key Words and Phrases: Geosocial networks, co-clustering, mapping clusterability, social similarity, spatial similarity, non-negative matrix factorization

ACM Reference Format:
Jungeun Kim, Jae-Gil Lee, Byung Suk Lee, and Jiajun Liu. 0000. Geosocial Co-Clustering: A Novel Framework for Geosocial Community Detection. *ACM Trans. Intell. Syst. Technol.* 0, 0, Article 00 (0000), 24 pages. https://doi.org/0000001.0000001

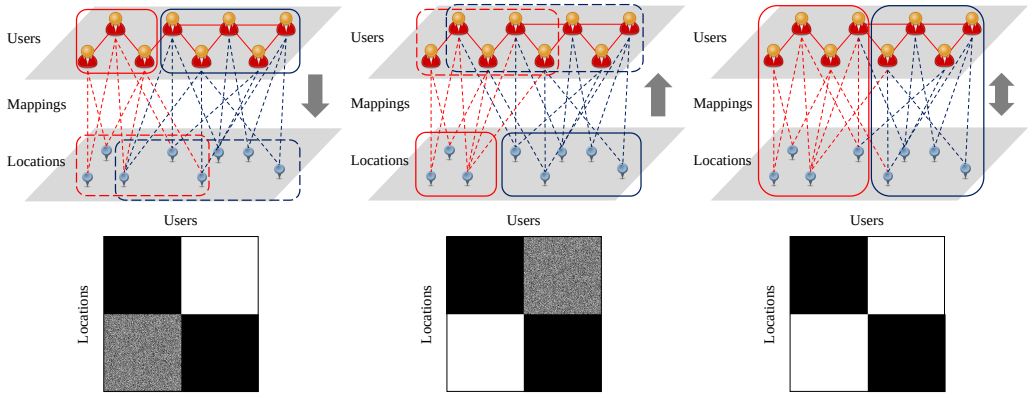
1 INTRODUCTION

The widespread use of location-aware mobile devices has made it feasible to collect precise locations of the users. In many social networking services, the users' locations play an important role in social

*Jae-Gil Lee is the corresponding author.

Authors' addresses: Jungeun Kim, Kongju National University, Department of Computer Science and Engineering, jekim@kongju.ac.kr; Jae-Gil Lee, Korea Advanced Institute of Science and Technology, Graduate School of Knowledge Service Engineering, jaegil@kaist.ac.kr; Byung Suk Lee, University of Vermont, Department of Computer Science, bslee@uvm.edu; Jiajun Liu, Renmin University, School of Information, jiajunliu@ruc.edu.cn.

2157-6904/0000/0-ART00 $15.00
https://doi.org/0000001.0000001



(a) User-centric approach. (b) Location-centric approach. (c) Mapping-centric approach.

Fig. 1. Comparison of the approaches for geosocial community detection.

analytics by enriching social relationships between users [25]. Particularly, *geosocial* networking services such as Foursquare and Yelp have attracted much attention since they can provide location intelligence through the analysis of not only users' social relationships but also their favorite places. For example, they recommend to users some popular places that their friends also like. Additionally, traditional social networking services such as Foursquare and Twitter have adopted geo-tagging capabilities to capture the location preferences of their users. Along these lines, a majority of social networking services of today retain abundant data about users' location preferences as well as social relationships.

There is strong correlation between social relationships and spatial preferences, as has been empirically proven by several researchers [4, 10, 42]. That is, people tend to interact more frequently with other people who live closer than those who live farther [4] or tend to visit nearby places which their friends or friends-of-friends have already visited [21, 42]. Thus, it is widely recognized that the quality of social network analysis improves by considering spatial preferences together with social relationships [44, 47]. Among various related analytic problems, we tackle the problem of *community detection* [22–24] or, more specifically, *geosocial community detection* for finding the sets of users who are densely connected through both close social relationships and similar spatial preferences. In this paper we consider *disjoint* community detection, which has been actively employed in its own set of applications, as indicated in a recent benchmark proposed by Wang et al. [43].

Most of the existing algorithms use clustering based on a similarity measure that *combines social* similarity and *spatial* similarity. Their approaches are distinguished into two categories depending on which side the clustering performed is centered on. In the *user-centric* approach, the clustering algorithm is essentially applied on the user side, with the spatial similarity incorporated into the social similarity represented as the edge weight [37, 40], and in the *location-centric* approach, on the location side with the social similarity incorporated into the spatial similarity measure [39]. As such, each category puts emphasis on a different perspective. As geosocial features become increasingly integrated in real applications, however, the strong correlation between the user side and the location side warrants a new, *mapping-centric* approach, which exploits the two sides of similarity equitably and simultaneously.

Example 1.1. Figure 1 shows clustering results obtained by the three approaches, with a social network view at the top and a mapping matrix view at the bottom. Figures 1a and 1b respectively show what the clusters from the user-centric and location-centric approaches would look like. Note

that these approaches are both disjoint community detection and are equipped to do clustering only on one side—either users or locations—and, therefore, clustering on the other side—locations or users—must be derived through *mapping* from clusters on the original one side. It has been observed that, as a result, clusters on the other side are blurred in their boundaries, hence poor in the quality (see Section 5.2 for the empirical results). In contrast, Figure 1c shows what the clusters from our mapping-centric approach would look like. It shows clearer clustering of mappings between users and locations, i.e., with few crossings between the subsets of mappings. That is, the mappings are neatly divided into bundles of mapping edges. □

1.1 Geosocial Co-Clustering

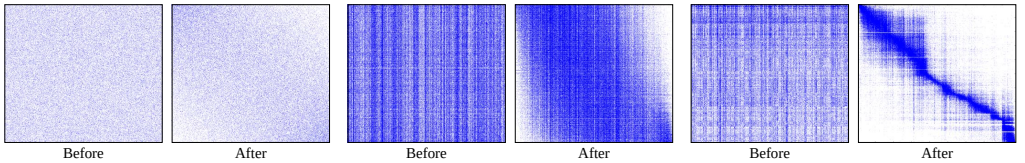
In this paper, we address mapping-centric geosocial community detection by way of geosocial co-clustering. *Co-clustering* is a generic technique that clusters the rows and columns of a matrix *simultaneously* [12]. In *geosocial* co-clustering, a row and a column of a matrix correspond to a user and a location, respectively. Therefore, users are clustered based on their spatial preferences (i.e., the distributions of the locations visited by them) and, at the same time, locations are clustered based on their user populations (i.e., the distributions of the users visiting them). As a result, it can produce pairs of a cluster on the user side and a cluster on the location side simultaneously.

Geosocial co-clustering improves the quality of the detected communities by achieving *mapping clusterability*, which indicates how well the mappings between the users and the locations are clustered into sets of matching user-location pairs. In other words, higher mapping clusterability means that there is a higher degree of one-to-one mapping between subsets of users and subsets of locations. Intuitively, it means that users who belong to the same community tend to visit a certain group of locations that are not visited by users who belong to a different community. Note that the existing algorithms (e.g., [37, 39, 40]), which perform clustering only on *one* side, i.e., either users or locations, do not care about the mapping clusterability.

Many applications can benefit from this mapping clusterability. One example application is to organize marketing teams so that each team shares good collaborative relationship and common expertise in the same region. Another example application is to recommend local experts [9] by finding a community of mutually familiar users who are experts in the same region, as activities concentrated on a certain topic often indicate high expertise in that topic [1]. This "regionalization" is an important marketing (or business in general) strategy commonly preached and practiced in view of today's globalization climate [19].

Example 1.2. Let us elaborate on the first application mentioned above as an example. Suppose we want to organize teams of personnel and assign a target region to each team for marketing and promotion. Each team needs to have good collaborative relationship to make the team trustworthy and harmonious and be sufficiently experienced in the target region to work efficiently. In addition, locations in the region assigned to a team need to be spatially close to reduce the moving time. Under this scenario, collaborative teams and their assigned regions need to be compactly *co-regulated* in terms of the social, spatial, and mapping aspect as shown in Figure 1c. Note that one-sided algorithms cannot be applied to this case since they only produce either collaborative teams or target regions (but not both) as shown in Figures 1a and 1b. One may say that the corresponding target regions (or collaborative teams) can be induced through mapping based on discovered collaborative teams (or regions). However, since induced sets tend to be spread widely, a too large region (or collaborative team) may be assigned to a small collaborative team (or region). In addition, induced sets tend to easily overlap, resulting in unnecessary duplication. □

We formulate geosocial co-clustering as a *matrix tri-factorization* problem. Here, a *user-to-location matrix* is constructed with elements that represent the frequencies of visits to a set of locations by a set of users. This matrix is then factorized into a *user membership matrix*, a scale matrix, and a



(a) Random data set. (b) Word-to-Doc data set. (c) Geosocial data set.

Fig. 2. Efficacy of GEOCODE for different degrees of mapping clusterability. The geosocial data set manifests the highest degree of mapping clusterability and thus benefits from GEOCODE most.

location membership matrix. The user membership matrix defines the membership of a user in a user-side cluster, and the location membership matrix defines the membership of a location in a location-side cluster. While this formulation is theoretically sound, actually computing the solution is known to be very costly. The state-of-the-art algorithm [16] has *quadratic* time complexity per iteration with the number of users and the number of locations.

1.2 GEOCODE Framework

In order to address the efficiency issue of geosocial co-clustering, we propose a framework for finding an approximate solution of the matrix factorization. We call this framework *GEOCODE*, which stands for *GEOsocial COarsening and DEcomposition*. GEOCODE is geared for geosocial networks which have a few intrinsic properties stemming from mapping clusterability (see Example 1.3): notably, (i) social relationships and geographic preferences are strongly correlated [4, 10, 42], and (ii) the degrees of social relationships and the populations of the places all follow the power-law distribution. We take advantage of these properties in order to improve the efficiency of geosocial co-clustering without losing the accuracy.

Our GEOCODE framework follows a divide-and-conquer strategy and consists of three steps: the first step reduces the size of the whole data set, and then the second step divides it into multiple subsets; the third step conquers each subset.

- **Step 1 (Coarsening)**: This step *coarsens* users and locations in order to reduce the numbers of rows and columns in the user-to-location matrix. Here, a set of users is collapsed into a (virtual) user, and a set of locations is collapsed into a (virtual) location. Because of the power-law distributions, it is reasonable for heavy users or big locations to absorb nearby insignificant users or locations. Furthermore, because of the correlation between users and locations, this coarsening does *not* significantly change the structure of the user-to-location matrix.
- **Step 2 (Decomposition)**: This step *decomposes* the two-layer graph in Figure 1c into multiple subgraphs such that the mappings between a pair of clusters that do not cross subgraph boundaries are minimized. For this purpose, this step first orders users and locations separately in a line by *crossing minimization* [2] and then detects the best cuts by the *minimum description length (MDL)* [15] principle. Each subgraph serves as a sub-matrix of the user-to-location matrix. The time complexity of this step is only linear logarithmic in terms of the numbers of users or locations.
- **Step 3 (Partial Co-Clustering)**: This step runs the state-of-the-art dual regularized co-clustering (DRCC) algorithm [16] for the matrix factorization against each sub-matrix identified in Step 2. Running the algorithm on each sub-matrix finishes much faster than running it on the whole matrix and can overlap through parallel execution. Therefore, the elapsed time for partial co-clustering is much shorter than that of co-clustering on the whole.

Example 1.3. Figure 2 shows the efficacy of GEOCODE visually using three data sets in an increasing degree of correlation: (i) random data set, (ii) word-to-doc data set (NIPS) [34], and (iii) geosocial data set (Gowalla). The random and word-to-doc data sets show poor correlation and thus show poor mapping clusterability. As a result, the efficacy of GEOCODE is significantly degraded

since the whole matrix cannot be decomposed into small sub-matrices and, thus, co-clustering cannot be performed in parallel. □

1.3 Summary

Key Contributions:

- *Problem Formalization* (Section 3): We formally define the problem of *geosocial co-clustering* to guarantee the mapping clusterability in geosocial community detection. Then, we formulate this problem mathematically as non-negative matrix tri-factorization with dual regularizers.
- *Efficiency Enhancement* (Section 4): We develop the GEOCODE framework to quickly find an approximate solution. Our framework reduces the size of the problem through coarsening and decomposition and also supports parallel execution of the sub-problems.
- *Comprehensive Evaluation* (Section 5): We empirically demonstrate the benefit of geosocial co-clustering and the efficiency improvement of GEOCODE (19–69 times) using four real-world data sets. GEOCODE successfully finds the geosocial communities whose members' interests are highly coherent.

Problem Significance: We propose to use *co-clustering* for geosocial community detection. Although co-clustering has been popularly used for related problems with different goals, e.g., *geosocial recommendation* [5, 29, 42, 48], it has *not* been a dominant player in geosocial community detection. Thus, it is worthwhile to promote co-clustering for geosocial community detection in order to share the benefit of the co-clustering approach.

2 RELATED WORK

2.1 Geosocial Community Detection

The major algorithms can be classified depending on how the two types of similarity are combined.

- *User-centric*: In these algorithms, a social distance measure is extended to consider spatial proximity [37, 40]. Shakarian et al. [37] proposed a community detection algorithm, Louvain-D, based on modularity maximization. Instead of using the original modularity, they used the *distance modularity* where the probability that two vertices are connected in a null model is dependent on the distance between the two vertices. Van Gennip et al. [40] proposed a spectral clustering algorithm for geosocial networks. The value of the adjacency matrix is updated by the weighted sum of the original value (i.e., connectivity) and the Euclidean distance between two individuals.
- *Location-centric*: In these algorithms, a spatial distance measure is extended to consider social proximity [39]. Shi et al. [39] proposed a model called *density-based clustering places in geo-social networks (DCPGS)*. A new geosocial distance measure between two places is defined by the weighted sum of social distance and spatial distance between the two places, where the social distance is defined by the number of users who visited both places. Then, a variant of DBSCAN is performed using this new distance measure.

2.2 Co-Clustering

Co-clustering is based on the duality between data points (e.g., locations) and features (e.g., users). That is, data points are grouped according to their distributions on features, and features are grouped according to their distributions on data points. A majority of the algorithms adopt matrix factorization [6, 14, 16, 38].

Algorithms based on matrix factorization are categorized by the numbers of factors and regularizers. Ding et al. [6] decompose a matrix into two factor matrices with one regularizer on data points only. Cai et al. [14] decompose a matrix into three factor matrices without regularizers. More recently, Gu and Zhou [16] and Shang et al. [38] decompose a matrix into three factor matrices with two regularizers on data points and features, respectively. Because of the dual regularizers, the

algorithm by Gu and Zhou is called *dual regularized co-clustering (DRCC)*. It optimizes one variable while fixing the other two variables, and the dynamic variable that is optimized alternates among the three variables. This procedure repeats until convergence, and convergence is theoretically guaranteed. The cost of this iterative optimization is known to be very high [41]. In fact, the work on co-clustering has used only medium-size data sets which contain up to a few hundred thousand data points.

2.3 Other Related Problems

2.3.1 Geosocial Recommendation. Geosocial recommendation (e.g., [5, 29, 42, 48]) and geosocial community detection commonly try to exploit social influence (social relationship), geographic influence (spatial distance), and user preference (mapping between users and locations) all together. However, the goal of geosocial recommendation is clearly different from that of geosocial community detection, because geosocial recommendation aims at predicting the probability that a user will visit an unvisited location.

Because a few recommendation algorithms such as CLR [29] and CCCF [45] adopt co-clustering, one might argue that the communities obtained by these algorithms could be used for our purpose. However, co-clustering in recommendation is mainly used to confine the search space to a specific community. Thus, co-clustering should be accompanied with further refinement or collaborative filtering, and it does not need to be as sophisticated as our geosocial co-clustering. As far as we know, none of co-clustering in recommendation fully uses the above-mentioned three aspects.

2.3.2 Geosocial Group Querying. Geosocial group querying (e.g., [3, 30, 33, 46]) and geosocial community detection commonly aim at finding users that satisfy both social and spatial constraints, but geosocial group querying is clearly distinct in that it requires a query user or location to be given whereas geosocial community detection does not. Besides, although geosocial group querying is proven to be useful for several applications, there are still many other applications that need geosocial community detection, especially when providing a query point or user is infeasible or inapplicable, e.g., terrorist monitoring [32, 37] and criminal monitoring [40].

3 PROBLEM STATEMENT

In this section, we formally define the problem of *geosocial co-clustering* and formulate it by matrix factorization. Table 1 summarizes the notation used throughout this paper.

3.1 Geosocial Co-Clustering

First, the *user-to-location matrix* $\mathbf{X}$ is constructed by Eq. (1), where $f(v_i, l_j)$ indicates the number of visits by a user v_i to a location l_j. The *location vector* and the *visitor vector* are introduced using $\mathbf{X}$ in Definitions 3.1 and 3.2.

$$\mathbf{X} = [X_{i,j}], \text{ where } X_{i,j} = f(v_i, l_j) \tag{1}$$

Definition 3.1. The *location vector* of a user v_i, $\mathbf{L}(v_i)$, is the i-th *row* vector of $\mathbf{X}$, which represents the number of times v_i has visited l_j ($j = 1, \ldots, n_L$). □

Definition 3.2. The *visitor vector* of a location l_j, $\mathbf{V}(l_j)$, is the j-th *column* vector of $\mathbf{X}$, which represents the number of times l_j has been visited by v_i ($i = 1, \ldots, n_U$). □

Next, a *geosocial network* is formally defined by adding the mappings to an ordinary graph, as in Definition 3.3.

Definition 3.3. A *geosocial network* is an attributed graph $\mathbb{G} = (V, E, L, \mathcal{F})$, where V is the set of vertices (i.e., users), E is the set of edges (i.e., social relationships between users), L is the set of locations, and $\mathcal{F}$ is a mapping from a user v_i onto a location vector $\mathbf{L}(v_i)$. Formally, $\mathcal{F} : V \rightarrow \mathbb{N}^{n_L}$. □

Table 1. Summary of the notation.

Notation	Description
$\mathbb{G} = (V, E, L, \mathcal{F})$	a geosocial network (V: users; E: user-user relationships; L: locations; $\mathcal{F}$: user-to-location mappings)
v_i or $v_j \in V$	a vertex, i.e., a user
l_i or $l_j \in L$	a location
$n_U = \|V\|$	the total number of users
$n_L = \|L\|$	the total number of locations
$f(v_i, l_j)$	frequency of user-to-location mappings
$\mathbf{L}(v_i)$	the location vector of the user v_i
$\mathbf{V}(l_i)$	the visitor vector of the location l_i
$D_S(v_i, v_j)$	the *social* distance between v_i and v_j
$D_P(l_i, l_j)$	the *spatial* distance between l_i and l_j
$\mathcal{G}_L = \{V_L, E_L\}$	a square grid graph (V_L: grid centers; E_L: grid connections)
$C \in \mathcal{C}$	a geosocial *community*
$n_C = \|\mathcal{C}\|$	the number of geosocial communities
$\mathbf{U}$	the user membership matrix
$R \in \mathcal{R}$	a geosocial *region*
$n_R = \|\mathcal{R}\|$	the number of geosocial regions
$\mathbf{K}$	the location membership matrix
$\mathbf{X}$	the user-to-location matrix

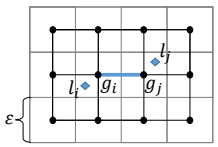


Fig. 3. Square grid graph over the spatial domain.

For V and L, *any* distance metrics can be used for the social distance D_S and the spatial distance D_P. We use the following metrics for simplicity, both based on the commonly used notion of the shortest-path distance in a graph. Again, the framework is independent of the choice of distance metrics.

- $D_S(v_i, v_j)$: We use the *shortest-path distance* [31] between two users v_i and v_j on $\mathbb{G}$. That is, it is the length (number of edges) of the shortest paths.
- $D_P(l_i, l_j)$: We use the Manhattan-like distance because the users are likely to move within cities. We first partition the spatial extent into square grids of side length ε, thus constructing a *square grid graph* $\mathcal{G}_L = \{V_L, E_L\}$ in Figure 3. Here, a vertex is the center of a grid square, and an edge connects the centers of two adjacent grid squares. Let g_i and g_j be the vertices (i.e., centers) of grid squares holding two locations l_i and l_j. Then, $D_P(l_i, l_j)$ is the shortest-path distance between g_i and g_j on the square grid graph.

Then, the communities on the user side and the clusters (regions) on the location side are defined in Definitions 3.4 and 3.5 respectively. Here, the number of geosocial communities, n_C, and the number of geosocial regions, n_R, are given by users as in the k-means algorithm. There are several heuristics (e.g., [7, 8, 17]) that can be applied to determine the values of n_C and n_R.

Definition 3.4. Geosocial communities are disjoint sets of similar users in terms of their spatial presences and social relationships. The optimal set of n_C geosocial communities is defined as $\mathcal{C} = \{C_1, \ldots, C_{n_C}\}$ that satisfies the following conditions simultaneously:

(1) $\sum_{C \in \mathcal{C}} \sum_{v_i \in C, v_j \in C, v_i \neq v_j} \|\mathbf{L}(v_i) - \mathbf{L}(v_j)\|$ is minimized, and

(2) $\sum_{C \in \mathcal{C}} \sum_{v_i \in C, v_j \in C, v_i \neq v_j} D_S(v_i, v_j)$ is minimized. □

Definition 3.5. *Geosocial regions* are disjoint sets of similar locations in terms of their spatial distances and shared visitors. The optimal set of n_R geosocial regions is defined as $\mathcal{R} = \{R_1, \ldots, R_{n_R}\}$ that satisfies the following conditions simultaneously:

(1) $\sum_{R \in \mathcal{R}} \sum_{l_i \in R, l_j \in R, l_i \neq l_j} \|\mathbf{V}(l_i) - \mathbf{V}(l_j)\|$ is minimized, and
(2) $\sum_{R \in \mathcal{R}} \sum_{l_i \in R, l_j \in R, l_i \neq l_j} D_P(l_i, l_j)$ is minimized. □

In addition, the goodness of geosocial co-clustering is formalized by Definition 3.6.

Definition 3.6. The *mapping clusterability* between communities and regions is the proportion of the mappings in the matching pairs of a community and a region (e.g., solid-line boxes in Figure 1c) out of all pairs between them. To formalize the *matching* pairs, first let $MS(C, R)$ be the strength of mapping between a community C and a region R defined by Eq. (2).

$$MS(C, R) = \sum_{v_i \in C, l_j \in R} f(v_i, l_j) \tag{2}$$

Then, $\mathcal{M}(\mathcal{C}, \mathcal{R})$ defined by Eq. (3) is the set of matching pairs, with the maximum strength of mapping, between $\mathcal{C}$ and $\mathcal{R}$; the mapping is from the larger set to the smaller set, and argmax here returns a singleton set by breaking a tie in the mapping strength arbitrarily.

$$\mathcal{M}(\mathcal{C}, \mathcal{R}) = \begin{cases} \{(C; R') \mid C \in \mathcal{C}, R' \in \text{argmax}_{R \in \mathcal{R}} MS(C, R)\} & \text{if } n_C \geq n_R \\ \{(C'; R) \mid C' \in \text{argmax}_{C \in \mathcal{C}} MS(C, R), R \in \mathcal{R}\} & \text{otherwise} \end{cases} \tag{3}$$

The *mapping clusterability* is then formulated by Eq. (4), which calculates the ratio of the total mapping strength of matching pairs over the total mapping strength of all pairs from $\mathcal{C}$ and $\mathcal{R}$.

$$Mapping\_clusterability(\mathcal{C}, \mathcal{R}) = \frac{\sum_{(C;R) \in \mathcal{M}(\mathcal{C}, \mathcal{R})} MS(C, R)}{\sum_{C \in \mathcal{C}, R \in \mathcal{R}} MS(C, R)} \tag{4}$$

There are *three* aspects considered altogether in geosocial co-clustering: (i) social similarity on the user side, (ii) spatial similarity on the location side, and (iii) mapping between users and locations. The *social similarity* aspect is addressed by the second requirement of Definition 3.4, and the *spatial similarity* aspect by the second requirement of Definition 3.5. The *mapping* aspect, that is mapping clusterability, is satisfied by the first requirements of Definitions 3.4 and 3.5, as they make the mappings from a subset on one side head for the same subset on the other side.

Based on these three aspects, the *geosocial co-clustering* is defined as in Definition 3.7.

Definition 3.7. Given a geosocial network $\mathbb{G}$, *geosocial co-clustering* finds a disjoint set of geosocial communities, $\mathcal{C} = \{C_1, \ldots, C_{n_C}\}$, and a disjoint set of geosocial regions, $\mathcal{R} = \{R_1, \ldots, R_{n_R}\}$, simultaneously in one process to maximize an objective that combines the three aspects in Definitions 3.4, 3.5, and 3.6. □

3.2 Formulation by Matrix Factorization

We translate the problem of geosocial co-clustering in Definition 3.7 to that of non-negative matrix tri-factorization [16].

3.2.1 Input Data Matrices. There are three input matrices for the three aspects of geosocial co-clustering.

- A *user affinity matrix* $\mathbf{W}^{\mathrm{U}}$ in Eq. (5) is an $n_U \times n_U$ matrix, where an element $W^U_{i,j}$ represents whether two users v_i and v_j are adjacent in $\mathbb{G}$.

$$W^U_{i,j} = \begin{cases} 1, & D_S(v_i, v_j) \leq 1 \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

- A *location affinity matrix* $\mathbf{W^L}$ in Eq. (6) is an $n_L \times n_L$ matrix, where an element $W^L_{i,j}$ represents whether the locations l_i and l_j are placed in the same grid square or in two adjacent grid squares.

$$W^L_{i,j} = \begin{cases} 1, & D_P(l_i, l_j) \leq 1 \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

- A *user-to-location matrix* $\mathbf{X}$ in Eq. (1) is an $n_U \times n_L$ matrix.

This simple affinity is widely used in co-clustering studies (e.g., [16, 41, 49]) since it has an advantage that does not require tuning parameters [16]. Of course, other kinds of affinity, such as heat kernel [18] and dot-product weighting [6], can be utilized as well; in such other affinity, $\mathbf{W^U}$ and $\mathbf{W^L}$ have a real value between 0 and 1, not just either 0 or 1.

3.2.2 Output Membership Matrices. The goal of our problem is to group the users into n_C communities $\mathcal{C} = \{C_1, \ldots, C_{n_C}\}$ as well as to group the locations into n_R regions $\mathcal{R} = \{R_1, \ldots, R_{n_R}\}$. Thus, we need to obtain the matrices that represent the *membership* of each user in a community and that of each location to a region. The membership matrix $\mathbf{U} \in \{0,1\}^{n_U \times n_C}$ holds the clustering result of users. $U_{i,j}$ becomes 1 if the user v_i belongs to the community C_j and 0 otherwise. Similarly, we use another membership matrix $\mathbf{K} \in \{0,1\}^{n_L \times n_R}$ that holds the clustering result of locations.

3.2.3 Objective Function. The objective function incorporates the three aspects of geosocial co-clustering.

First, to support the *social similarity* aspect as in Definition 3.4, high similarity between two users is penalized unless they belong to the same community. This requirement is formulated by Eq. (7). Here, the *graph Laplacian* is given by $\mathbf{K}_U = \mathbf{D}^U - \mathbf{W}^U$; $\mathbf{W}^U$ defined as Eq. (5) is the symmetric adjacency matrix, and $\mathbf{D}^U$ is a diagonal matrix with the degree of each vertex where $D^U_{i,i} = \sum_j W^U_{i,j}$, i.e., the column sums of $\mathbf{W}^U$.

$$J_{social} = \sum_{i,j} \|U_i - U_j\|^2 W^U_{i,j} = 2\text{Tr}(\mathbf{U}^T\mathbf{D}^U\mathbf{U}) - 2\text{Tr}(\mathbf{U}^T\mathbf{W}^U\mathbf{U}) = 2\text{Tr}(\mathbf{U}^T\mathbf{K}_U\mathbf{U}) \tag{7}$$

Second, to support the *spatial similarity* aspect as in Definition 3.5, high similarity between two locations is penalized unless they belong to the same region. This requirement is formulated by Eq. (8). Here, the graph Laplacian is given by $K_L = \mathbf{D}^L - \mathbf{W}^L$, where $D^L_{i,i} = \sum_j W^L_{i,j}$.

$$J_{spatial} = \sum_{i,j} \|L_i - L_j\|^2 W^L_{i,j} = 2\text{Tr}(\mathbf{K}^T\mathbf{D}^L\mathbf{K}) - 2\text{Tr}(\mathbf{K}^T\mathbf{W}^L\mathbf{K}) = 2\text{Tr}(\mathbf{K}^T\mathbf{K}_L\mathbf{K}) \tag{8}$$

Third, to support the *mapping* aspect as in Definition 3.6, the users in a specific community and the locations in a specific region should have as tight connections as possible. This is where *co-clustering* [16, 38, 41] kicks in. The objective function of co-clustering is usually represented by non-negative matrix (tri-)factorization, as in Eq. (9). Here, $\|\cdot\|$ represents the Frobenius norm. Compared to *two*-factor factorization, *tri*-factorization provides a better approximation by the addition of the scale matrix $\mathbf{S}$, which absorbs the different scales of $\mathbf{X}$, $\mathbf{U}$, and $\mathbf{K}$ [41].

$$J_{mapping} = \|\mathbf{X} - \mathbf{USK}^T\|^2_F \tag{9}$$

Putting Eqs. (7), (8), and (9) together, our problem is to minimize Eq. (10) in accordance with Definition 3.7. Here, $\lambda \geq 0$ and $\mu \geq 0$ are used to balance the reconstruction error of co-clustering in the first term (Eq. (9)) against the quality of the geosocial communities in the second term (Eq. (7)) and that of the geosocial regions in the third term (Eq. (8)), respectively. Thus, Eq. (10) is represented as non-negative matrix tri-factorization with dual regularizers (i.e., the second term and the third term). We relax the binary constraints of $\mathbf{U}$ and $\mathbf{K}$ to facilitate optimization as in other co-clustering algorithms.

$$J_{GEO\_CC} = \|\mathbf{X} - \mathbf{USK}^T\|^2_F + \lambda \cdot 2\text{Tr}(\mathbf{U}^T\mathbf{K}_U\mathbf{U}) + \mu \cdot 2\text{Tr}(\mathbf{K}^T\mathbf{K}_L\mathbf{K}) \text{ s.t. } \mathbf{U} \geq 0, \mathbf{K} \geq 0 \tag{10}$$

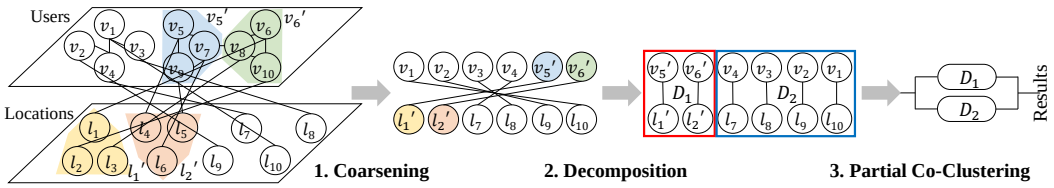


Fig. 4. Illustration of the overall procedure of GEOCODE.

ALGORITHM 1: GEOCODE (Main Framework)

INPUT: $\mathbb{G}$, n_C, n_R: Table 1; λ, μ: regularization parameters;
OUTPUT: C and $\mathcal{R}$;

```
C ← ∅, R ← ∅;   /* Initialization */
/* Step 1: Coarsening (Section 4.1) */
G' = (V', E', L', F') ← Coarsening(G);
/* Step 2: Decomposition (Sections 4.2–4.3) */
Ordered V' and L' ← Crossing Minimization(G');          /* Step 2-1: Algorithm 2 */
G'_1, G'_2, ... ← Cut Point Detection(Ordered V' and L');   /* Step 2-2: Algorithm 3 */
/* Step 3: Partial Co-Clustering (Section 4.4) */
parallel for each G'_i do
    n'_C, n'_R ← Parameter Setting(G'_i, n_C, n_R);
    C_i, R_i ← Partial Co-Clustering(G'_i, n'_C, n'_R, λ, μ);
    C ← C ∪ C_i, R ← R ∪ R_i;
end parallel for
return C and R;
```

3.2.4 Co-Clustering Algorithm. The DRCC algorithm [16] optimizes the objective with respect to one variable while fixing the other variables. This procedure repeats until convergence, because convergence is theoretically guaranteed [16]. Using *dual* regularizers and *tri*-factorization is shown to improve the quality of the clustering result [16, 38]. Despite this advantage, the algorithm notoriously suffers from slow computation speed because of intensive matrix multiplications involved in each iteration step [41], which makes co-clustering difficult to support large-scale geosocial networks in the real world. Its time complexity is proven to be $O(N^2)$, where $N = max(n_U, n_L)$.

4 GEOCODE FRAMEWORK

This section proposes GEOCODE developed for fast and accurate *approximate* processing of geosocial co-clustering in Section 3.

Recap on Overall Procedure: GEOCODE consists of three steps, as illustrated in Figure 4. Algorithm 1 outlines the overall procedure of GEOCODE.

- **Step 1 (Coarsening)** coarsens users and locations in order to reduce the size of a geosocial network (Section 4.1). For example, v'_5 replaces a merger of v_5, v_7, and v_9 in Figure 4.
- **Step 2 (Decomposition)** first reorders users and locations for easy decomposition (Section 4.2) and then finds the optimal cuts to derive multiple sub-matrices (Section 4.3). For example, in Figure 4 the nodes v_1–v_4 and the nodes l_7–l_{10} constitute a sub-matrix.
- **Step 3 (Partial Co-Clustering)** solves co-clustering for each sub-matrix and merge the solutions (Section 4.4).

Design Rationale: The coarsening and decomposition do *not* damage the accuracy significantly, due to the following rationales behind them.

- *Coarsening* is mainly motivated by the power-law distributions of social connections and spatial densities. The active users with many social connections become the centers of groups in social

coarsening, and other neighboring users collapse into their groups. A similar idea holds for very densely-populated grids in spatial coarsening.

- *Decomposition* benefits from the correlation between friendship and distance [4, 10, 42]. The locations visited by a user have strong clustering tendency because their frequency degrades *inversely proportional* to the distance from home [42, 48]. This clustering tendency tends to be shared by the users tied with social friendship. Therefore, the subsets of the users and locations connected by the common user-to-location mapping are typically cleanly identifiable.

4.1 Step 1: Coarsening

A geosocial network $\mathbb{G} = (V, E, L, \mathcal{F})$ is reduced in size to $\mathbb{G}' = (V', E', L', \mathcal{F}')$ through social and spatial coarsening. *Social coarsening* encapsulates a socially connected user group into one virtual user (Definition 4.1). *Spatial coarsening* approximates the spatial coordinate of locations by space partitioning (Definition 4.2).

Definition 4.1. Social coarsening converts the original set of vertices and edges (V, E) to a reduced set of vertices and edges (V', E'), as defined by Eqs. (11) and (12). A single edge $e_{i,j} \in E$ is selected by a certain rule, and the two endpoints $v_i, v_j \in V$ are merged into v_{new}. The edges that were incident to either v_i or v_j now point to the newly added vertex v_{new}. □

$$V' = V \setminus \{v_i, v_j\} \cup \{v_{new}\} \tag{11}$$

$$\begin{aligned} E' = E &\setminus \{e_{i,j}\} \\ &\setminus \{e_{i,k} \mid e_{i,k} \in E \wedge 1 \le k \le n_U\} \setminus \{e_{k,j} \mid e_{k,j} \in E \wedge 1 \le k \le n_U\} \\ &\cup \{e_{new,k} \mid e_{i,k} \in E \wedge 1 \le k \le n_U\} \cup \{e_{k,new} \mid e_{k,j} \in E \wedge 1 \le k \le n_U\} \end{aligned} \tag{12}$$

In social coarsening, the key issue is how to pick the edges whose end vertices will be merged. We use a simple yet effective heuristic called the *sorted heavy edge matching* [20], following the power-law distributions of social connections. It visits the vertices in the ascending order of their degrees while breaking ties randomly; for each vertex visited, it selects an incident edge with the *highest weight* among those whose end vertices have not merged yet, and then goes to the next vertex in the order of degree. This step repeats until the total number of vertices becomes a certain percentage of that of the original graph. A benefit of this heuristic is that the partitions of a coarser graph tend to have relatively small edge cuts [20].

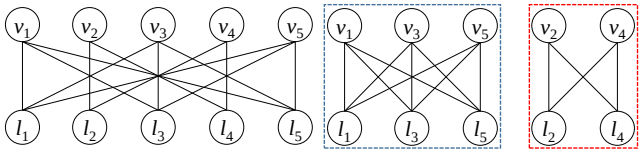
Definition 4.2. Spatial coarsening converts the original set of locations L to a reduced set of locations L' by dividing the spatial domain into larger grid squares of side length ε' ($> \varepsilon$). Note that locations are collapsed to the center points of the grid squares of $\mathcal{G}_L$. (See Figure 3 about $\mathcal{G}_L$.) □

The mapping between users and locations, $\mathcal{F}$, is updated to reflect the coarsened users and locations.

4.2 Step 2-1: Crossing Minimization

This step aims to achieve the same effects as the three terms in Eq. (10) with regard to the co-clustering. In Eq. (10), the first term pressures the algorithm toward clustering the *mappings* between users and locations into "bundles" (i.e., partitioned collections of mappings), the second term toward clustering similar *users* together, and the third term toward clustering similar *locations* together. With the coarsening in place, in GEOCODE we use crossing minimization to pressure the algorithm toward clustering the mappings and use two types of network motifs—*social motif* and *spatial motif*—toward clustering similar users and similar locations, respectively.

Crossing minimization reorders the vertices on both sides of a bipartite graph such that the number of edge crossings is minimized. For example, the input bipartite graph in Figure 5a has 28 crossings, whereas the optimal solution in Figure 5b has only 10 crossings. It has the same effect of minimizing the first term in Eq. (10) because it can be used to obtain a solution of spectral



(a) Input bipartite graph. (b) Optimal solution.

Fig. 5. Example of crossing minimization.

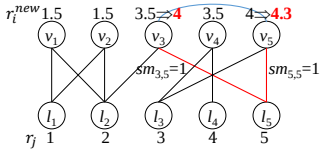


Fig. 6. Concept of the motif-based weighting.

clustering [2]. Note that Laplacian-based spectral clustering is equivalent to non-negative matrix factorization [13]. Because optimal crossing minimization is NP-hard [2], several heuristics have been proposed, including the *barycenter* heuristic [2]. This heuristic assigns a new rank of a vertex on one side using the weighted mean of the ranks of its neighbors on the other side until convergence is reached. The intuition here is that the vertices with similar mappings (i.e., set of neighbors on the other side) should have similar ranks.

4.2.1 Motif-Based Weighting Scheme. In order to support the second and third terms as well, we extend the barycenter heuristic by proposing a novel weighting scheme, called the *motif-based weighting*. In Definition 4.3, the weight of an edge $\langle v_i, l_j \rangle$ between a user v_i and a location l_j contains not only the number of visits by v_i to l_j but also the number of triads formed by social relationship, called *social motifs*, and that of triads formed by spatial adjacency, called *spatial motifs*. For example, a social motif is formed by v_3, v_5, and l_5 in Figure 6, where the edge between v_3 and v_5 indicates social relationship.

Definition 4.3. The *motif-based weighting* is to assign the weight of an edge $\langle v_i, l_j \rangle$ by $w_{i,j} = X_{i,j} + \lambda \cdot sm_{i,j} + \mu \cdot pm_{i,j}$, where $sm_{i,j}$ and $pm_{i,j}$ are, respectively, the numbers of social and spatial motifs that the edge belongs to. $X_{i,j}$ is the number of visits by v_i to l_j, and λ and μ are the regularization parameters in Eq. (10). □

By adding the motif-based weighting to the barycenter heuristic, the rank of a vertex on each side is determined by Eq. (13).

$$r_i^{new} = \frac{\sum_{j \in \mathbb{N}(i)} w_{i,j} \times r_j}{\sum_{j \in \mathbb{N}(i)} w_{i,j}} \tag{13}$$

Here, r_i^{new} is the *new* rank of the i-th vertex on one side, r_j is the *current* rank of the j-th vertex on the other side, and $\mathbb{N}(i)$ is the neighbors of the i-th vertex.

The weight of an edge is *boosted* if it is involved in social or spatial motifs, which makes the rank of a common neighbor vertex *more heavily affect* those of the vertices socially or spatially close. For example, the ranks of v_3 and v_5, which are socially close, are heavily affected by the rank of l_5. In this way, reordering takes account of social relationship and spatial adjacency as well as mapping clusterability.

Example 4.4. In Figure 6, suppose that $X_{i,j}$'s are all 1. In the typical weighting scheme using only $X_{i,j}$, the new ranks on the user side are determined by the averages of the ranks of the neighbors, which are 1.5, 1.5, 3.5, 3.5, and 4. Then, v_3 and v_5 may *not* have consecutive ranks since v_4 can be placed between them. In contrast, assuming $\lambda = \mu = 1$ for simplicity of calculation, the motif-based

weighting boosts the ranks of v_3 and v_5 to 4 and 4.3 respectively. Now, v_3 and v_5 should have consecutive ranks. □

ALGORITHM 2: Motif-Based Crossing Minimization

INPUT: $\mathbf{G}'$; unordered V' and L';
OUTPUT: ordered V' and L';
1: $dynamicSide \leftarrow V'$, $staticSide \leftarrow L'$;
2: **repeat**
3: $isChanged \leftarrow false$;
4: **for each** $i \in dynamicSide$ **do**
5: $r_i^{new} \leftarrow$ Calculate using Eq. (13); /* Determine i's new rank */
6: **if** $r_i \neq \lfloor r_i^{new} \rfloor$ **and** $r_i \neq \lceil r_i^{new} \rceil$ **then**
7: $r_i \leftarrow r_i^{new}$, $isChanged \leftarrow true$;
8: **end if**
9: **end for**
10: Sort all vertices in $dynamicSide$ by r_i's;
11: Adjust their ranks in the sorted order;
12: Swap $dynamicSide$ and $staticSide$;
13: **until** $isChanged \neq true$;
14: **return** ordered V' and L';

4.2.2 Algorithm Details. Algorithm 2 outlines the steps of motif-based crossing minimization. One side of the input bipartite graph in Figure 1 is static, and the other sides is dynamic; the roles of the two sides are switched through iterations until there occurs no more change of ranks. Specifically, in each iteration, for each vertex i on the dynamic side (either V' or L'), its new rank r_i^{new} is calculated by Eq. (13) (Line 5). Finally, the ranks are determined by assigning a unique integer index in the order of r_i (Lines 10 to 11).

THEOREM 4.5. *The time complexity of the motif-based crossing minimization in Algorithm 2 is* $O(t\,(|V'| + |V'|\log|V'| + |L'| + |L'|\log|L'|))$, *where* t *is the number of iterations performed. This complexity can be expressed as* $O(t\,N\log N)$, *where* $N = \max(|V'|, |L'|)$.

PROOF. Suppose that the dynamic side is V'. In each iteration of the REPEAT loop (Lines 2 to 13), calculating r_i^{new} for each vertex takes $O(\langle d_v\rangle \cdot |V'|)$, where $\langle d_v\rangle$ is the average number of user-to-location mappings per user (Lines 4 to 9); sorting all vertices takes $O(|V'|\log|V'|)$ (Line 10); adjusting the ranks of the vertices takes $O(|V'|)$. When the dynamic side is switched to L', $|V'|$ should be replaced with $|L'|$. Since $d_v \ll |V'|$ and $d_l \ll |L'|$, the total complexity of the REPEAT loop can be expressed as $O(t\,(|V'| + |V'|\log|V'| + |L'| + |L'|\log|L'|))$. It can be further simplified to $O(t\,N\log N)$, where $N = \max(|V'|, |L'|)$. □

4.3 Step 2-2: Cut Detection

After the vertices on the two sides are reordered individually, this step aims to split them into multiple subsets like those boxed in Figure 5b, each of which forms a sub-matrix of the user-to-location matrix $\mathbf{X}$. A set of $k-1$ cuts divide the set of users (or locations) into k subsets with maintaining their order. That is, the elements until the cut and those after the cut are allocated to different subsets. Then, each sub-matrix is determined by Definition 4.6. The two subsets of the same order form a sub-matrix since the users should have dense mappings with the locations of similar ranks, and vice versa.

Definition 4.6. V' is split into $\{V'_1, \ldots, V'_k\}$ where $\forall i \neq j$, $V'_i \cap V'_j = \emptyset$ and $\bigcup_{i=1}^{k} V'_i = V'$; L' is split into $\{L'_1, \ldots, L'_k\}$ where $\forall i \neq j$, $L'_i \cap L'_j = \emptyset$ and $\bigcup_{j=1}^{k} L'_i = L'$. Then, a *sub-matrix* $D_i = \langle V'_i, L'_i\rangle$ is defined as mappings between users in V'_i and locations in L'_i. □

Table 2. The notation for the MDL formulation.

Notation	Description
D_i	a sub-matrix
$s_V(D_i)$	the number of users in D_i
$s_L(D_i)$	the number of locations in D_i
$n(D_i)$	the number of possible mappings in D_i ($= s_V(D_i) \times s_L(D_i)$)
$n_0(D_i)$	the number of mapping absences in D_i
$n_1(D_i)$	the number of mapping existences in D_i
$C(D_i)$	the code cost of D_i

The key technical challenge is to select the *optimal set of cuts*. To this end, we contend that *conciseness* and *homogeneity* should be satisfied. Conciseness keeps the cut from generating too many small sub-matrices, because communities or regions crossing sub-matrix boundaries cannot be discovered by GEOCODE. Homogeneity drives the user-to-location mappings within each sub-matrix to be as similar as possible. That is, the users should have similar location vectors (Definition 3.1), and at the same time, the locations should have similar visitor vectors (Definition 3.2).

Evidently, there is a trade-off between these two properties. In one extreme, one can make the entire user-to-location matrix a sub-matrix, but then the homogeneity in the entire matrix will be very poor. In the other extreme, one can make each pair of vertices a sub-matrix, which achieves the highest homogeneity because a single user or location has the same mapping by itself. It thus offers an optimization problem, which can be formulated by the minimum description length (MDL) principle [15].

4.3.1 Formalization Using the MDL Principle. The cost of the MDL principle consists of two components: $L(H)$ and $L(D|H)$. Here, H means the hypothesis, and D means the data. The two components are informally stated as follows [15]: "$L(H)$ is the length, in bits, of the description of the hypothesis; and $L(D|H)$ is the length, in bits, of the description of the data when encoded with the help of the hypothesis." The best hypothesis H to explain D is the one that minimizes the sum of $L(H)$ and $L(D|H)$.

H corresponds to the cuts of the user and location sides, and D corresponds to the user-to-location mappings. As a result, finding a good split translates to finding the best hypothesis using the MDL principle. Table 2 summarizes the notation used here.

- $L(H)$ is formulated by Eq. (14). The first term is required to describe the number of sub-matrices. Here, $\log^*$ is the universal code length for integers [26]. The second and third terms are required to describe the number of users and the number of locations, in each sub-matrix D_i.

$$L(H) = \log^* k + \sum_{i=1}^{k} \left\{ \lceil \log_2 s_V(D_i) \rceil + \lceil \log_2 s_L(D_i) \rceil \right\} \tag{14}$$

- $L(D|H)$ is formulated by Eq. (15). The first term is required to describe the number of possible mappings in D_i. The second term, called the *code cost*, represents the number of bits required to transmit the user-to-location mappings within D_i. If the mappings coincide among all users (or locations), which means that D_i is perfectly homogeneous, $C(D_i)$ becomes zero, but if the mappings are different among them, $C(D_i)$ becomes high.

$$L(D|H) = \sum_{i=1}^{k} \left\{ \lceil \log_2 (n(D_i) + 1) \rceil + C(D_i) \right\} \tag{15}$$

For the derivation of $C(D_i)$, we regard that a mapping between a user and a location is binary, ignoring the frequency for simplicity of entropy calculation. Then, $n(D_i) = n_0(D_i) + n_1(D_i)$. By the information theory, an existence of a mapping and an absence of a mapping can be

ALGORITHM 3: Cut Detection by MDL

INPUT: Ordered V' and L' by Algorithm 2;
OUTPUT: A set of sub-matrices $\{\mathbb{G}'_1, \ldots, \mathbb{G}'_k\}$

1: $\mathbb{D} \leftarrow \{\langle V', L' \rangle\}$, $minCost \leftarrow MDL(\mathbb{D})$;
2: **repeat**
3: $isDecreasing \leftarrow false$;
4: /* Pick up the one with the largest code cost */
5: $m \leftarrow \text{argmax}_{1 \le m \le |\mathbb{D}|} C(\langle V'_m, L'_m \rangle)$;
6: /* Find the best split on the *location* side */
7: $p \leftarrow \text{argmin}_{1 \le p \le l_{end}} \{C(\langle V'_m, L'_m[1,p] \rangle) + C(\langle V'_m, L'_m(p, l_{end}] \rangle)\}$;
8: /* Find the best split on the *user* side */
9: $q \leftarrow \text{argmin}_{1 \le q \le v_{end}} \{C(\langle V'_m[1,q], L'_m[1,p] \rangle) + C(\langle V'_m(q, v_{end}], L'_m(p, l_{end}] \rangle)\}$;
10: /* Check if the MDL cost decreases */
11: $\mathbb{D}' \leftarrow \mathbb{D} \setminus \{\langle V'_m, L'_m \rangle\} \cup \{\langle V'_m[1,q], L'_m[1,p] \rangle, \langle V'_m(q, v_{end}], L'_m(p, l_{end}] \rangle\}$;
12: $newCost \leftarrow MDL(\mathbb{D}')$;
13: **if** $newCost < minCost$ **then**
14: $\mathbb{D} \leftarrow \mathbb{D}'$, $minCost \leftarrow newCost$;
15: $isDecreasing \leftarrow true$;
16: **end if**
17: **until** $isDecreasing \neq false$;
18: **return** $\mathbb{D}$; /* Best set of sub-matrices */

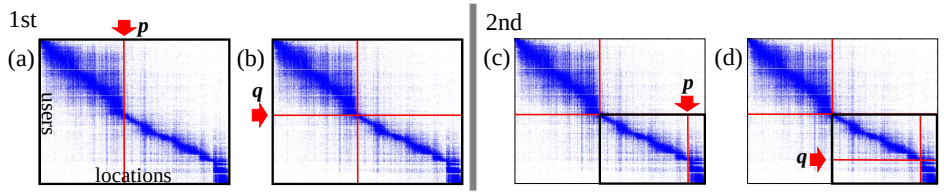


Fig. 7. The first two iterations of Algorithm 3.

encoded using $-\log_2 n_1(D_i)/n(D_i)$ bits and $-\log_2 n_0(D_i)/n(D_i)$ bits, respectively, on average. Thus, $C(D_i)$ is formulated by Eq. (16).

$$C(D_i) = -n(D_i) \sum_{j \in \{0,1\}} \frac{n_j(D_i)}{n(D_i)} \log_2 \frac{n_j(D_i)}{n(D_i)} \tag{16}$$

4.3.2 Algorithm Details. Although theoretically plausible, finding the best hypothesis is computationally prohibitive since we need to consider all possible splits of the entire matrix *even without knowing the number of cuts*. Thus, Algorithm 3 is designed as an approximation algorithm that progressively divides the matrix and finds a better set of sub-matrices alternately for the users and for the locations in a greedy fashion. Initially, all users and locations form a single sub-matrix (Line 1). In every iteration, the sub-matrix with the largest code cost is selected for split (Lines 4 to 5). Then, the best split point is determined each for the location side and the user side such that the split minimizes the code cost (Lines 6 to 9). Here, a split is denoted by the interval of ranks in a subset, e.g., $[1, p]$ and $(p, l_{end}]$. If this split reduces the MDL cost, the set of sub-matrices $\mathbb{D}$ is updated to reflect the split (Lines 10 to 16). These procedures repeat until the MDL cost does not decrease any longer. We note that the number of cuts is automatically determined.

THEOREM 4.7. *The time complexity of the cut detection in Algorithm 3 is $O(t\,(|V'| + |L'|))$, where t is the number of iterations performed. This complexity can be expressed as $O(t\,N)$, where $N = \max(|V'|, |L'|)$.*

Table 3. Profiles of the four real-world geosocial networks.

Data Set	#users (n_U)	#locations (n_L)	#check-in's ($\sum_{ij} f(v_i, l_j)$)	#edges ($\|E\|$)	geosocial properties		
					(α)	(β_{social})	($\beta_{spatial}$)
Brightkite	50,886	72,434	4,730,778	197,167	0.64	2.48	1.83
Gowalla	99,563	79,725	6,272,297	456,830	0.68	2.65	1.85
Foursquare	301,228	152,427	10,021,823	2,616,276	0.52	1.57	2.13
Foursquare_s	383	451	9,188	258	0.32	1.71	1.64

Proof. In each iteration of the REPEAT loop (Lines 2 to 17), finding the best split takes at most $O(|L'|)$ for the location side (Line 7) and $O(|V'|)$ for the user side (Line 9), because there are up to $l_{end} \leq |L'|$ and $v_{end} \leq |V'|$ possible choices, respectively. Thus, the total complexity of the REPEAT loop can be expressed as $O(t\,(|V'| + |L'|))$. □

Figure 7 illustrates the first two iterations of Algorithm 3. In this figure, the rows indicate users, and the columns indicate locations. A mapping is denoted by a point at the intersection of a user and a location. In the first iteration, the best split is found along the location side (Figure 7(a)) and then the user side of the entire matrix (Figure 7(b)). In the second iteration, there are two sub-matrices, and the second sub-matrix is selected for split. The same procedure is applied to this sub-matrix (Figures 7(c)–(d)).

4.4 Step 3: Partial Co-Clustering

After identifying the set of sub-matrices, we can use any conventional co-clustering algorithm for each sub-matrix. In this paper, our natural choice is the DRCC algorithm [16]. When applying DRCC to each sub-matrix, we need to provide the number of geosocial communities and the number of geosocial regions in that sub-matrix. To this end, n_C and n_R are distributed to each sub-matrix in proportion to the sum of $X_{i,j}$ (i.e., the numbers of visits) in that sub-matrix out of the total sum of $X_{i,j}$. Then, the DRCC algorithm runs concurrently by using multiple threads or machines. The results of DRCC for all sub-matrices are merged to form the final result of geosocial co-clustering.

5 EVALUATION

The evaluation was done through two sets of experiments with the following respective focuses: (i) the benefits of using the geosocial *co-clustering* approach on the cluster quality (Section 5.2) and (ii) the resulting performances of GEOCODE (Section 5.3).

All experiments were performed on a PC with Intel Core i7-3770 3.4 GHz CPU and 32 Gbyte RAM, running Windows 7. The number of parallel threads in Step 3 was set to be 4, equal to the number of cores. All algorithms were implemented in Java.

5.1 Experiment Setup

5.1.1 Data Sets. Four real-world geosocial networks were used in our experiments. *Brightkite* and *Gowalla* are the popular data sets available at the SNAP repository [27]. *Foursquare* is provided by the courtesy of Sarwat et al. [36]. *Foursquare_s* is a proprietary data set [11] collected from Gangnam District in Seoul, Korea during the period from April to December 2012, and contains the reviews left by users for each venue. Since *Foursquare_s* covers only a single district, we use it only for case study in Section 5.2.2.

Table 3 shows the profiles of the data sets. The first four columns are generic network parameters, and the last three columns under geosocial properties are unique to GEOCODE. Specifically, the last three columns are geosocial correlation's exponent (α) and the power-law's exponent (β_{social} for social connections and $\beta_{spatial}$ for spatial densities), expressing the two properties benefiting GEOCODE (see Section 1.2) according to the model described below.

Geosocial Properties Model: The geosocial correlation is defined as $P(d) = d^{-\alpha}$, the probability that two users at distance d are friends given the exponent α that indicates the degree of correlation. The power-law distribution is defined as $Y = kX^{-\beta}$, the frequency Y of a node (a user or a spatial grid) that has the degree X, given the scale k and the power-law's exponent β. Taking the log on both sides of the equations gives us $\log P(d) = -\alpha d$ and $\log Y = -k\beta X$, and therefore the relationship between $P(d)$ and d and the relationship between Y and X are linear in a log-log scale when the relationships are exponential in a linear-linear scale. In this regard, the exponents α and β can be used as parameters indicating the geosocial properties mentioned in Section 1.2. All geosocial network data sets showed adequate linear fitting in the log-log scale.

5.1.2 Compared Algorithms. In the first set of experiments, we compared GEOCODE with the user-centric approach of **Louvain-D** [37] and the location-centric approach of **DCPGS** [39]. In the second set of experiments, we compared GEOCODE with the state-of-the-art co-clustering algorithm **DRCC** [16].

5.1.3 Parameter Settings. For both GEOCODE and DRCC, which are based on co-clustering, we used the same parameter values. The regularization parameters in Eq. (10) were set to be $\lambda = \mu = 250$ as empirically determined by Gu and Zhou [16]. We also verify the impact of these parameter values in Section 5.3.3. Additionally, we set n_C and n_R to be a common number by a widely-accepted heuristic for estimating the number of clusters [7, 8, 17], i.e., to $\lfloor n_U \cdot n_L / |\mathbf{X}_{>0}| \rfloor$, where $|\mathbf{X}_{>0}|$ denotes the number of non-zero elements in $\mathbf{X}$. For the parameters unique to GEOCODE, the number of sub-matrices was *automatically* determined by Algorithm 3, and the proportions of n_C and n_R for each sub-matrix were distributed by the heuristic in Section 4.4.

For user-centric and location-centric approaches, since the semantics of their parameters are not compatible with our approach, we used the default settings suggested by the authors. For DCPGS, we set the minimum number of neighbors *minPts*, social constraint τ, geosocial distance constraint ϵ, and balancing parameter ω for social and spatial distance to be 5, 0.7, 0.4, and 0.5, respectively, following the default values. *maxD*, the maximum distance between two adjacent spatial grid squares, was naturally $2\sqrt{2} \times$ grid square size. For Louvain-D, we set the exponential distance decay γ to be $1/e$, following the default value, too.

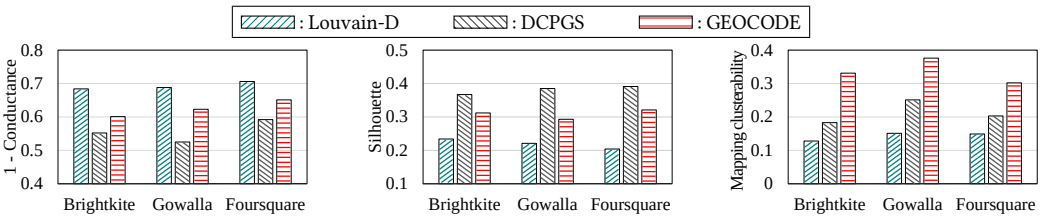
5.1.4 Coarsening Schemes. Coarsening is performed by GEOCODE to reduce the numbers of users and locations almost without sacrificing accuracy. Level 0 means the *original* sets of users and locations. At level 0, the grid square size in Figure 3 was set to be 5 kilometers in the first three data sets and 100 meters in the fourth data set because it was collected from a much smaller region. As the level increases by 1, the numbers of users and locations are reduced by approximately 20%. In social coarsening, the sorted heavy matching algorithm [20] stops when the 20% criterion is met; in spatial coarsening, the grid square size is gradually enlarged until the criterion is met. As a result, $|V'| \approx 0.80|V|$ and $|L'| \approx 0.80|L|$ at level 1, $|V'| \approx 0.64|V|$ and $|L'| \approx 0.64|L|$ at level 2, and so on.

5.1.5 Performance Metrics. The following quality metrics were adopted or designed to show the merits of GEOCODE.

User-Side Community Quality (Figure 8a): The *conductance* [28] was used because it is designed for clustering of graph data.[1] It is defined as the fraction of total edge volume that points outside a cluster. A lower value means higher cluster quality.

Location-Side Cluster Quality (Figure 8b): The *silhouette coefficient* [35] was used because it is designed for clustering of spatial or numeric data. It measures how similar locations within the

[1]We also used the *internal density* [28] as another metric. The results showed almost the same trend as the conductance.



(a) User-side community quality. (b) Location-side cluster quality. (c) Mapping cluster quality.

Fig. 8. Comparison of clustering quality in three aspects between geosocial co-clustering and existing approaches.

same cluster are compared with locations in different clusters. A higher value means higher cluster quality.

Mapping Clusterability (Figures 8c, 10, and 13): Eq. (4) in Definition 3.6 was used to quantify the mapping clusterability.

Semantic Similarity (Table 4 and Figure 9): Additionally, we designed the following two *semantic similarity* metrics for the quantifiable benefits of geosocial co-clustering in real-world applications discussed in Section 5.2.2.

- The *review similarity* in Eq. (17) means the textual similarity of the review collections written by the users in a community C. For this measure, a term vector is created for each user using his/her reviews, and cosine similarity is calculated for each pair of the term vectors $\boldsymbol{u}_i$ and $\boldsymbol{u}_j$. Then, the cosine similarities between pairs, $cos(\boldsymbol{u}_i, \boldsymbol{u}_i)$, are averaged over all pairs.

$$\text{Review similarity} = \frac{\sum_{i=1}^{|C|}\sum_{j=1}^{|C|} cos(\boldsymbol{u}_i, \boldsymbol{u}_j)}{|C|(|C|-1)/2}, \text{ where } i < j \tag{17}$$

- The *category similarity* in Eq. (18) means the similarity in the categories of the places visited by the users in a community. Let R be the set of such places visited. Then, for each pair of places l_i and l_j in R, the sum of the path lengths to the lowest common ancestor, $LCA(l_i, l_j)$, is measured in the hierarchical category tree[2]. Then, converting dissimilarity to similarity by an exponential decay function, $e^{-LCA(l_i,l_j)}$ between pairs are averaged over all pairs.

$$\text{Category similarity} = \frac{\sum_{i=1}^{|R|}\sum_{j=1}^{|R|} e^{-LCA(l_i,l_j)}}{|R|(|R|-1)/2}, \text{ where } i < j \tag{18}$$

High review and category similarities in a community or cluster indicate *coherent* interests of users in the community, that is, the high quality of the cluster as a community representation.

5.2 Benefits of Geosocial Co-Clustering

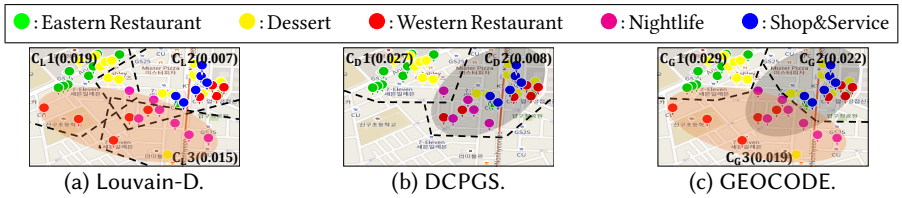
5.2.1 Quantitative Analysis. In order to verify that geosocial co-clustering works as designed with real-world data sets, we compared GEOCODE, the user-centric approach of Louvain-D, and the location-centric approach of DCPGS in *three* aspects: (i) the quality of communities on the user side, (ii) the quality of regions (clusters) on the location side, and (iii) the mapping clusterability between them, as in Figure 8. In these experiments, the coarsening level was set to be 0. Since the user-centric and location-centric approaches produce clusters on only one side, the clusters on the other side were derived for comparison purpose through the mappings, as shown with the overlapping broken-line boxes in Figure 1.

As for the quality of *user*-side communities in Figure 8a, Louvain-D performed the best since its clustering was centered on the user side, and DCPGS performed the worst. GEOCODE achieved high quality, too, as close as 81.2–86.7% of Louvain-D. In contrast, as for the quality of *location*-side clusters in Figure 8b, DCPGS performed the best, and Louvain-D performed the worst. Again,

[2]See https://developer.foursquare.com/docs/resources/categories.

Table 4. Semantic similarity of the users in communities.

	Review similarity	Category similarity
Louvain-D	0.141	0.011
DCPGS	0.152	0.014
GEOCODE	**0.228**	**0.024**



(a) Louvain-D. (b) DCPGS. (c) GEOCODE.

Fig. 9. Location-side clusters discovered by Louvain-D, DCPGS, and GEOCODE (best viewed in color).

GEOCODE achieved high quality, too, as close as 78.7–87.7% of DCPGS. Therefore, GEOCODE achieved balanced high quality on *both sides* as opposed to the one-sided quality (at the expense of the other side) achieved by Louvain-D and DCPGS.

As for the mapping clusterability in Figure 8c, GEOCODE significantly outperformed the other two approaches by up to 2.6 times, as expected from the fact that GEOCODE improves the mapping clusterability through multiple iterations alternating on each side. In summary, we confirm that GEOCODE additionally guarantees the mapping clusterability while marginally degrading the quality of communities or regions on either side.

5.2.2 Case Study with Real Application. We aim to demonstrate the effect of the mapping clusterability in *local expert recommendation* using the Foursquare_s data set. This dataset contains records of users' visits to restaurants, including the venue categories[3] and locations of restaurants and comments left by users about the restaurants. As discussed earlier, the mapping clusterability enables us to find the communities of local experts who exclusively concentrate on specific topics. Thus, high coherency within a community or region should be achieved by GEOCODE. The review similarity measures the coherency *within a community* (i.e., on the user side) in terms of the review contents, and the category similarity measures the coherency *within a region* (i.e., on the location side) in terms of the venue categories.

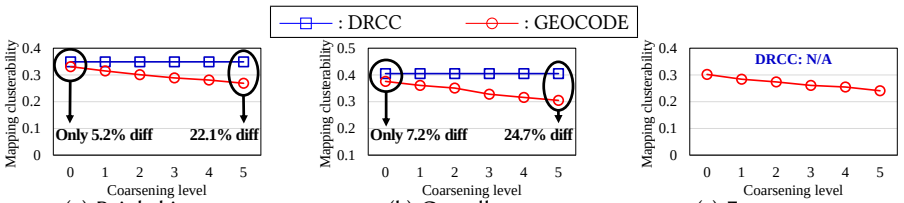
Louvain-D, DCPGS, and GEOCODE produced 18, 16, and 18 communities or regions, respectively, from this data set. Table 4 shows the average values of the review and category similarities from the three approaches. GEOCODE naturally achieved significantly higher similarities than Louvain-D and DCPGS. This result means that the interests of the users were shared more closely in GEOCODE than in the other two approaches.

Location-Side Semantics: Figure 9 shows the category similarity aspect through visualization. In this figure, the categories of different venues are color-coded, and the boundaries of regions (clusters) are shown in broken lines; the category similarity of each region is shown in parentheses next to the cluster label. In Figure 9a, Louvain-D incorrectly separated venues of the same category (in the orange area) into different regions. In contrast, GEOCODE in Figure 9c correctly put them in the same region. In Figure 9b, DCPGS incorrectly put venues of several different categories (in the grey area) in the same region. In contrast, GEOCODE in Figure 9c more correctly put them in separate regions. Overall, GEOCODE is doing the best job in putting venues in regions coherently and completely among the three approaches. Note that the category information is *not* used at all for geosocial co-clustering.

[3]"Venue category" is a FourSquare Developers term. See footnote 2.

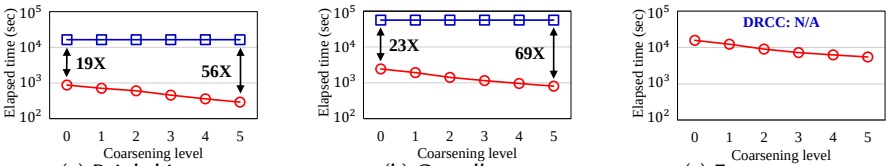
Table 5. Top ten keywords from the reviews in the colored areas of Figure 9 (best viewed in color).

Louvain-D vs. GEOCODE(**orange** area)		DCPGS vs. GEOCODE (**grey** area)	
Louvain-D	GEOCODE	DCPGS	GEOCODE
coffee, ice flakes, spicy seafood noodle, bread, beer, parking, refill, wine, service, spaghetti	bread, beer, pizza, fried food, salad, music, reservation, discount, wine, weekend	coffee, ice flakes, beer, refill, service, pizza, wine, kindness, spaghetti, noodle	coffee, ice flakes, refill, service, pizza, beer, wine, bread, pizza, salad



(a) Brightkite. (b) Gowalla. (c) Foursquare.

Fig. 10. Mapping clusterability of GEOCODE and DRCC with varying the coarsening level.



(a) Brightkite. (b) Gowalla. (c) Foursquare.

Fig. 11. Elapsed time of GEOCODE and DRCC with varying the coarsening level.

User-Side Semantics: Regarding the review similarity aspect, Table 5 shows the top ten most *frequent* keywords appearing in the reviews by the users of the communities mapped from the colored areas in Figure 9. The keywords are color-coded by the colors of their categories. Louvain-D and GEOCODE were compared for the orange area in Figures 9a and 9c, and DCPGS and GEOCODE for the grey area in Figures 9b and 9c. In both cases, the topics (i.e., *colors*) of the keywords were more coherent in GEOCODE than in Louvain-D or DCPGS, which can be visually confirmed by counting the number of different colors in the keyword list as well as confirmed numerically (see Table 4). For instance, for the DCPGS in the grey area, the keyword "noodle" about *eastern restaurants* blurred the overall main topics, *western restaurants* and *nightlife*. Overall, the review similarity observed is highly coherent, thus showcasing the community quality of GEOCODE on the user side as well.

5.3 Performances of GEOCODE

5.3.1 Comparison with Co-Clustering on the Whole. Please recall that GEOCODE transforms co-clustering into the parallel execution of multiple partial co-clustering. In this regard, we aim to verify that GEOCODE significantly improves the efficiency over the state-of-the-art co-clustering algorithm DRCC while observing the resulting degree of compromise in accuracy. Figures 10 and 11 show the accuracy and efficiency, respectively, resulting from GEOCODE and DRCC with varying the coarsening level for each data set. Evidently, the coarsening level had a nontrivial effect in GEOCODE but was irrelevant to DRCC. Besides, DRCC failed to produce results for the Foursquare data set (in Figures 10c and 11c) because of an out-of-memory error.

Accuracy: The accuracy shown in Figure 10 is the mapping clusterability measured when the coarsening level varied from 0 to 5. The value of GEOCODE was very close to that of DRCC (different by as small as 5.2–7.2%) when the coarsening level was 0 (i.e., no coarsening), and when

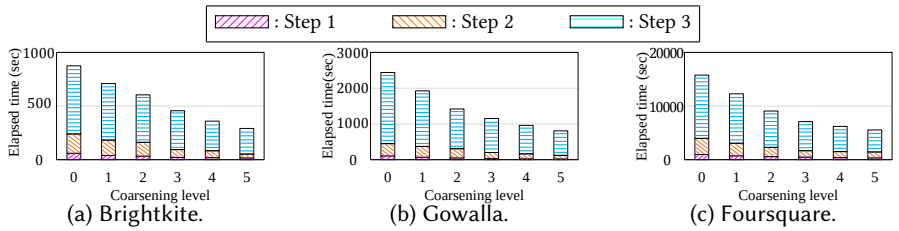


Fig. 12. Breakdown of GEOCODE elapsed time with varying the coarsening level.

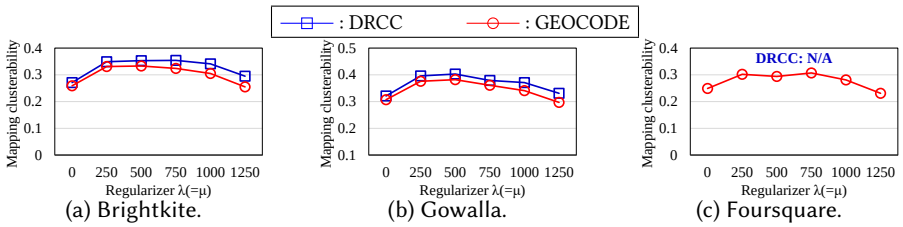


Fig. 13. Impact of the regularization parameters λ and μ ($\lambda = \mu$) on the mapping clusterability.

the level was increased to 5, the gap of the value increased to only 22.1–24.7%, about 4 times the initial gap.

Efficiency: The efficiency shown in Figure 11 is the elapsed time when the coarsening level varied from 0 to 5. Even without coarsening (i.e., at level 0), GEOCODE greatly outperformed DRCC by 19–23 times and by the time the level was increased to 5, the ratio reached 56–69 times, about 3 times the initial ratio.

These results demonstrate GEOCODE's ability to enhance the efficiency of geosocial co-clustering at only marginal expense of the accuracy. We assert that this improved efficiency is significant enough to make GEOCODE resolve the inability of the conventional co-clustering algorithm to handle large networks due to the high computational overhead.

5.3.2 Breakdown of Elapsed Time. Figure 12 shows the breakdown of the elapsed time of GEOCODE into the three steps discussed in Section 4. We observe that the first two steps took only a minor portion of the total elapsed time—no more than 6.8% for Step 1 and no more than 21.2% for Step 2, both much less than 72.6–82.1% for Step 3. This difference is understandable from the running time complexities of the individual steps. Step 1 (coarsening) takes linear-logarithmic time [20], Step 2 (decomposition) also takes linear-logarithmic time by Theorems 4.5 and 4.7, and Step 3 (partial co-clustering) takes quadratic time. The breakdown profile thus indicates that the overhead of the extra two steps (Steps 1 and 2) on the total elapsed time is not significant at all; in other words, the efficiency gain achieved by GEOCODE comes at little additional cost for the pre-processing steps.

5.3.3 Sensitivity to Regularization Parameters. We performed a sensitivity test of the mapping clusterability[4] for varying the values of the two regularization parameters λ and μ. Please recall that λ and μ are used to adjust the influences of the second and third terms, respectively, in the GEOCODE motif-based weighting scheme (Definition 4.3) and also in the DRCC objective function (Eq. (10)). Thus, our goal in this test was to see whether and how the second and third terms of the modif-based weighting scheme play the same roles as those in the DRCC objective function.

Figure 13 shows the results, where λ and μ were always set to the same value varying from 0 to 1250, and the coarsening level was set to be 0. We observe that GEOCODE and DRCC showed

[4]We conducted the tests on the qualities of communities and regions as well, and the results looked almost the same.

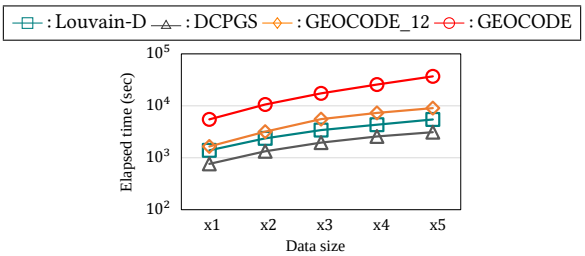


Fig. 14. Scalability with data size (×1 to ×5).

almost exactly the same pattern as the parameter values changed. The quality remained stable and close to the maximum while the parameter values were in the range of 250 to 750. Additionally, the parameter value 250, used for both λ and μ for DRCC according to Gu and Zhou [16], was shown to be a reasonable value for GEOCODE as well. Thus, the results empirically confirm that the social and spatial motifs of the motif-based weighting scheme do address the social similarity and spatial similarity aspects, respectively.

5.3.4 Scalability with Data Size. We increased the size of the Foursquare data set, the largest among the four data sets in Table 3, up to 5 times. In each step of the increase, we replicated the original network structure and added new users, locations, and mappings between them.

Figure 14 shows the resulting elapsed time of GEOCODE. DRCC failed to run even for the original Foursquare data set. It also shows GEOCODE_12, the portion of GEOCODE including only the preprocessing steps (i.e., Steps 1 and 2) and excluding the partial co-clustering (i.e., Step 3). When the data size increased five times from ×1 to ×5, the elapsed time increased by 5.4 times for GEOCODE_12 and by 6.7 times for GEOCODE. These results show that GEOCODE's runtime, in fact, increases sub-quadratically with dataset size, as the worst-case time complexity of Steps 1 and 2 is $O(N \log N)$ but that of Step 3, which takes about 70% of the total execution time, is $O(N^2)$.

Figure 14 shows the elapsed times of Lovain-D and DCPGS as well. Admittedly GEOCODE is not as fast as Louvain-D and DCPGS, apparently because of the overhead of multiple iterations. Notwithstanding, we assert that GEOCODE is sufficiently fast, as it finishes in 5 to 6 hours for a large-scale geosocial network with about 1 million users and 40 million check-in's on a single machine.

6 CONCLUSION

In this paper, we proposed *geosocial co-clustering*, which efficiently co-clusters the users in social networks and the locations they visited. Geosocial co-clustering proved to detect more desirable communities than the existing approaches by maximizing the *mapping clusterability*. We first formulated it as a mathematical non-negative matrix tri-factorization problem. Then, we developed the *GEOCODE* framework to improve its computational efficiency through the coarsening and decomposition of social users and locations, respectively, into groups. The decomposition used the crossing minimization technique as well as the MDL principle and enabled partitioning of a user-to-location matrix into sub-matrices so that individual sub-matrices could be co-clustered separately. The effect was several orders of magnitude speedup while maintaining comparable accuracy. Besides, the overheads of the coarsening and decomposition steps proved to be very light, thus adding to the efficacy of the GEOCODE framework. Overall, we believe that GEOCODE is a practical and useful framework for running geosocial community detection from large-scale geosocial networks.

The future work includes supporting overlapping communities and handling temporal evolution of communities.

ACKNOWLEDGMENTS

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) (No. 2017R1E1A1A01075927).

REFERENCES

[1] Lada A. Adamic, Jun Zhang, Eytan Bakshy, and Mark S. Ackerman. 2008. Knowledge Sharing and Yahoo Answers: Everyone Knows Something. In *Proc. 17th Int'l Conf. on World Wide Web*. 665–674.

[2] Waseem Ahmad and Ashfaq Khokhar. 2007. cHawk: An Efficient Biclustering Algorithm based on Bipartite Graph Crossing Minimization. In *Proc. VLDB Workshop on Data Mining in Bioinformatics*. 1553–1558.

[3] Nikos Armenatzoglou, Stavros Papadopoulos, and Dimitris Papadias. 2013. A General Framework for Geo-Social Query Processing. *Proc. VLDB Endowment* 6, 10 (2013), 913–924.

[4] Lars Backstrom, Eric Sun, and Cameron Marlow. 2010. Find Me If You Can: Improving Geographical Prediction with Social and Spatial Proximity. In *Proc. 19th Int'l Conf. on World Wide Web*. 61–70.

[5] Jie Bao, Yu Zheng, and Mohamed F. Mokbel. 2012. Location-based and Preference-Aware Recommendation Using Sparse Geo-Social Networking Data. In *Proc. 20th ACM SIGSPATIAL Int'l Conf. on Advances in Geographic Information Systems*. 199–208.

[6] Deng Cai, Xiaofei He, Jiawei Han, and Thomas S. Huang. 2011. Graph Regularized Nonnegative Matrix Factorization for Data Representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 33, 8 (2011), 1548–1560.

[7] Fazli Can and Esen A. Ozkarahan. 1990. Concepts and Effectiveness of the Cover-Coefficient-based Clustering Methodology for Text Databases. *ACM Trans. on Database Systems* 15, 4 (1990), 483–517.

[8] Carlos Castro-Herrera, Chuan Duan, Jane Cleland-Huang, and Bamshad Mobasher. 2009. A Recommender System for Requirements Elicitation in Large-scale Software Projects. In *Proc. 2009 ACM Sympo. on Applied Computing*. 1419–1426.

[9] Zhiyuan Cheng, James Caverlee, Himanshu Barthwal, and Vandana Bachani. 2014. Who is the Barbecue King of Texas?: A Geo-Spatial Approach to Finding Local Experts on Twitter. In *Proc. 37th Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*. 335–344.

[10] Eunjoon Cho, Seth A. Myers, and Jure Leskovec. 2011. Friendship and Mobility: User Movement in Location-Based Social Networks. In *Proc. 17th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*. 1082–1090.

[11] Minsoo Choy, Jae-Gil Lee, Gahgene Gweon, and Daehoon Kim. 2014. Glaucus: Exploiting the Wisdom of Crowds for Location-Based Queries in Mobile Environments. In *Proc. 8th AAAI Int'l Conf. on Weblogs and Social Media*. 61–70.

[12] Inderjit S. Dhillon, Subramanyam Mallela, and Dharmendra S. Modha. 2003. Information-Theoretic Co-Clustering. In *Proc. 9th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*. 89–98.

[13] Chris Ding, Xiaofeng He, and Horst D. Simon. 2005. On the Equivalence of Nonnegative Matrix Factorization and Spectral Clustering. In *Proc. 2005 SIAM Int'l Conf. on Data Mining*. 606–610.

[14] Chris Ding, Tao Li, Wei Peng, and Haesun Park. 2006. Orthogonal Nonnegative Matrix Tri-factorizations for Clustering. In *Proc. 12th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*. 126–135.

[15] Peter D. Grünwald, In Jae Myung, and Mark A Pitt. 2005. *Advances in Minimum Description Length: Theory and Applications*. MIT press.

[16] Quanquan Gu and Jie Zhou. 2009. Co-Clustering on Manifolds. In *Proc. 15th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*. 359–368.

[17] Negar Hariri, Carlos Castro-Herrera, Mehdi Mirakhorli, Jane Cleland-Huang, and Bamshad Mobasher. 2013. Supporting Domain Analysis through Mining and Recommending Features from Online Product Listings. *IEEE Trans. on Software Engineering* 39, 12 (2013), 1736–1752.

[18] Xiaofei He and Partha Niyogi. 2004. Locality Preserving Projections. In *Advances in neural information processing systems*. 153–160.

[19] Andrew E. G. Jonas. 2012. Region and place: Regionalism in question. *Progress in Human Geography* 36, 2 (2012), 263–272.

[20] George Karypis and Vipin Kumar. 1998. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing* 20, 1 (1998), 359–392.

[21] Jungeun Kim, Minseo Kang, Sungsu Lim, and Jae-Gil Lee. 2015. Triangle Counting in Networks using a Multi-Level Branching Technique. In *Proc. 2015 Int'l Conf. on Big Data and Smart Computing*. 47–50.

[22] Jungeun Kim and Jae-Gil Lee. 2015. Community Detection in Multi-Layer Graphs: A Survey. *ACM SIGMOD Record* 44, 3 (2015), 37–48.

[23] Jungeun Kim, Jae-Gil Lee, and Sungsu Lim. 2016. Differential Flattening: A Novel Framework for Community Detection in Multi-Layer Graphs. *ACM Trans. on Intelligent Systems and Technology* 8, 2 (2016), 27.

[24] Jungeun Kim, Sungsu Lim, Jae-Gil Lee, and Byung Lee. 2018. LinkBlackHole*: Robust Overlapping Community Detection Using Link Embedding. *IEEE Trans. on Knowledge and Data Engineering* 31, 11 (2018), 2138–2150.

[25] Jae-Gil Lee and Minseo Kang. 2015. Geospatial Big Data: Challenges and Opportunities. *Big Data Research* 2, 2 (2015), 74–81.
[26] Thomas Lee. 2001. An Introduction to Coding Theory and the Two-Part Minimum Description Length Principle. *International Statistical Review* 69, 2 (2001), 169–183.
[27] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data. (June 2014).
[28] Jure Leskovec, Kevin J. Lang, and Michael Mahoney. 2010. Empirical Comparison of Algorithms for Network Community Detection. In *Proc. 19th Int'l Conf. on World Wide Web*. 631–640.
[29] Kenneth Wai-Ting Leung, Dik Lun Lee, and Wang-Chien Lee. 2011. CLR: A Collaborative Location Recommendation Framework Based on Co-Clustering. In *Proc. 34th Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*. 305–314.
[30] Yafei Li, Rui Chen, Jianliang Xu, Qiao Huang, Haibo Hu, and Byron Choi. 2015. Geo-Social K-Cover Group Queries for Collaborative Spatial Computing. *IEEE Trans. on Knowledge and Data Engineering* 27, 10 (2015), 2729–2742.
[31] David Liben-Nowell and Jon M. Kleinberg. 2007. The Link-Prediction Problem for Social Networks. *Journal of the American Society for Information Science and Technology* 58, 7 (2007), 1019–1031.
[32] Richard M. Medina and George F. Hepner. 2011. Advancing the Understanding of Sociospatial Dependencies in Terrorist Networks. *Transactions in GIS* 15, 5 (2011), 577–597.
[33] Kyriakos Mouratidis, Jing Li, Yu Tang, and Nikos Mamoulis. 2015. Joint Search by Social and Spatial Proximity. *IEEE Trans. on Knowledge and Data Engineering* 27, 3 (2015), 781–793.
[34] Valerio Perrone, Paul A Jenkins, Dario Spano, and Yee Whye Teh. 2017. Poisson Random Fields for Dynamic Feature Models. *The Journal of Machine Learning Research* 18, 1 (2017), 4626–4670.
[35] Peter J. Rousseeuw. 1987. Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis. *J. Comput. Appl. Math.* 20 (1987), 53–65.
[36] Mohamed Sarwat, Justin J Levandoski, Ahmed Eldawy, and Mohamed F Mokbel. 2014. LARS*: An Efficient and Scalable Location-Aware Recommender System. *IEEE Trans. on Knowledge and Data Engineering* 26, 6 (2014), 1384–1399.
[37] Paulo Shakarian, Patrick Roos, Devon Callahan, and Cory Kirk. 2013. Mining for Geographically Disperse Communities in Social Networks by Leveraging Distance Modularity. In *Proc. 19th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*. 1402–1409.
[38] Fanhua Shang, L. C. Jiao, and Fei Wang. 2012. Graph Dual Regularization Non-negative Matrix Factorization for Co-clustering. *Pattern Recognition* 45, 6 (2012), 2237–2250.
[39] Jieming Shi, Nikos Mamoulis, Dingming Wu, and David W. Cheung. 2014. Density-based Place Clustering in Geo-Social Networks. In *Proc. 2014 ACM SIGMOD Int'l Conf. on Management of Data*. 99–110.
[40] Yves van Gennip, Blake Hunter, Raymond Ahn, Peter Elliott, Kyle Luh, Megan Halvorson, Shannon Reid, Matthew Valasik, James Wo, George E. Tita, Andrea L. Bertozzi, and P. Jeffrey Brantingham. 2013. Community Detection Using Spectral Clustering on Sparse Geosocial Data. *SIAM Journal of Applied Mathematics* 73, 1 (2013), 67–83.
[41] Hua Wang, Feiping Nie, Heng Huang, and Chris Ding. 2011. Nonnegative Matrix Tri-factorization Based High-Order Co-clustering and Its Fast Implementation. In *Proc. 11th IEEE Int'l Conf. on Data Mining*. 774–783.
[42] Hao Wang, Manolis Terrovitis, and Nikos Mamoulis. 2013. Location Recommendation in Location-based Social Networks using User Check-in Data. In *Proc. 21st ACM SIGSPATIAL Int'l Conf. on Advances in Geographic Information Systems*. 364–374.
[43] Meng Wang, Chaokun Wang, Jeffrey Xu Yu, and Jun Zhang. 2015. Community Detection in Social Networks: An In-depth Benchmarking Study with a Procedure-Oriented Framework. *Proc. VLDB Endowment* 8, 10 (2015), 998–1009.
[44] Xiaoyang Wang, Ying Zhang, Wenjie Zhang, and Xuemin Lin. 2016. Distance-Aware Influence Maximization in Geo-social Network. In *Proc. 32nd IEEE Int'l Conf. on Data Engineering*. 1–12.
[45] Yao Wu, Xudong Liu, Min Xie, Martin Ester, and Qing Yang. 2015. CCCF: Improving Collaborative Filtering via Scalable User-Item Co-Clustering. In *Proc. 9th ACM Int'l Conf. on Web Search and Data Mining*. 73–82.
[46] De-Nian Yang, Chih-Ya Shen, Wang-Chien Lee, and Ming-Syan Chen. 2012. On Socio-Spatial Group Query for Location-Based Social Networks. In *Proc. 18th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*. 949–957.
[47] Hongzhi Yin, Zhiting Hu, Xiaofang Zhou, Hao Wang, Kai Zheng, Nguyen Quoc Viet Hung, and Shazia Wasim Sadiq. 2016. Discovering Interpretable Geo-Social Communities for User Behavior Prediction. In *Proc. 32nd IEEE Int'l Conf. on Data Engineering*. 942–953.
[48] Jia-Dong Zhang and Chi-Yin Chow. 2013. iGSLR: Personalized Geo-Social Location Recommendation - A Kernel Density Estimation Approach. In *Proc. 21st ACM SIGSPATIAL Int'l Conf. on Advances in Geographic Information Systems*. 334–343.
[49] Marinka Zitnik and Blaz Zupan. 2015. Data Fusion by Matrix Factorization. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 37, 1 (2015), 41–53.