CLR: A Collaborative Location Recommendation Framework based on Co-Clustering

Kenneth Wai-Ting Leung
Hong Kong University of Science and Technology
kwtleung@cse.ust.hk

Dik Lun Lee
Hong Kong University of Science and Technology
dlee@cse.ust.hk

Wang-Chien Lee
The Pennsylvania State University
wlee@cse.psu.edu

ABSTRACT

GPS data tracked on mobile devices contains rich information about human activities and preferences. In this paper, GPS data is used in location-based services (*LBSs*) to provide collaborative location recommendations. We observe that most existing LBSs provide location recommendations by clustering the User-Location matrix. Since the User-Location matrix created based on GPS data is huge, there are two major problems with these methods. First, the number of similar locations that need to be considered in computing the recommendations can be numerous. As a result, the identification of truly relevant locations from numerous candidates is challenging. Second, the clustering process on large matrix is time consuming. Thus, when new GPS data arrives, complete re-clustering of the whole matrix is infeasible. To tackle these two problems, we propose the Collaborative Location Recommendation (CLR) framework for location recommendation. By considering activities (i.e., temporal preferences) and different user classes (i.e., *Pattern Users*, *Normal Users*, and *Travelers*) in the recommendation process, CLR is capable of generating more precise and refined recommendations to the users compared to the existing methods. Moreover, CLR employs a dynamic clustering algorithm CADC to cluster the trajectory data into groups of similar users, similar activities and similar locations efficiently by supporting incremental update of the groups when new GPS trajectory data arrives. We evaluate CLR with a real-world GPS dataset, and confirm that the CLR framework provides more accurate location recommendations compared to the existing methods.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data Mining*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Clustering, Information Filtering*

General Terms

Algorithms, Experimentation

Keywords

Co-Clustering, Collaborative Filtering, Location Recommendation

SIGIR'11, July 24–28, 2011, Beijing, China.

1. INTRODUCTION

One of the major functions of a location-based service (LBS) is to recommend interesting locations to the users. Most existing LBSs [2] recommend all points of interest (POIs) that are near the users. However, the number of POIs that are close to the user's current position (called the *active location* in this paper) can be numerous. Thus, it is difficult for a user to pick the most relevant suggestions from the large number of recommendations. In order to provide more accurate location suggestions according to the users' interests, many existing LBSs [1], [9], [10], [15] provide location recommendations by first clustering the User-Location matrix, which represents the locations visited by each user, and then making location recommendations based on the user and location clusters. However, a major problem with these methods is that the User-Location matrix created based on GPS data is huge, so the number of recommendations is still very large and the problem of selecting the truly relevant locations remains challenging.

To tackle this problem, we propose the *collaborative location recommendation* (CLR) framework for location recommendation. CLR collects users' GPS trajectory data to discover a set of points of interest (POIs) as candidate locations. Instead of merely considering users and locations represented in a User-Location matrix, we introduce the *Community Location Model* (CLM), which incorporates activities in addition to users and locations. CLM captures the relations between the three important entities, namely, users, activities and locations, in GPS trajectory data with a User-Activity-Location tripartite data structure, which we call CLM graph for simplicity. We argue that activities are important because most users visit locations with the intention to perform some activities (e.g., go to school to study, go to office to work, go to a cinema to watch movie). A novelty of CLM is its ability to derive from GPS trajectories users' temporal preferences through their activities. CLM explores the unique properties among users, activities, and locations to make effective location recommendations. We observe that (1) locations exhibit spatial properties as they contain geographical information; (2) activities exhibit temporal-spatial properties as they represent temporal sequences of visited locations; and (3) users exhibit long-term spatial properties as different users tend to visit different locations according to their long term habits and geographical limits. We argue that, by capturing temporal-spatial information in user activities, the proposed CLM is more effective in discovering similar locations comparing to existing techniques that are based on the User-Location matrix.

In CLR, a co-clustering algorithm *CADC* is employed to cluster the CLM graph in order to obtain high-quality location recommendations. There are a couple of advantages of applying CADC on the CLM graph. First, by exploring the relations between users, activities and locations modeled in CLM, CADC can iteratively

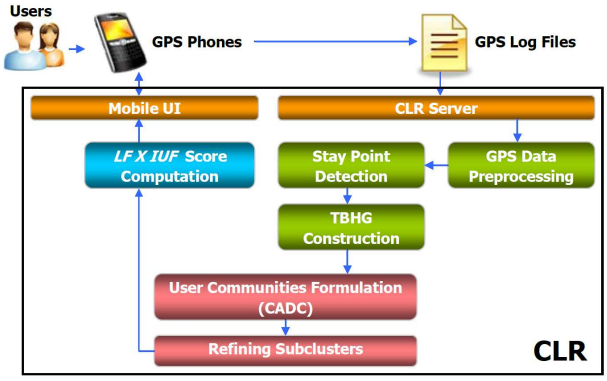


Figure 1: The general process flow of CLR.

cluster different types of similar entities into coherent clusters of similar users, similar activities, and similar locations. Entities in a cluster can be used to discover refined subclusters for making precise recommendations. Moreover, when new GPS trajectory data arrives, CADC can incrementally update the clusters using two separate phases, namely, the *agglomerative phase* and *divisive phase*, without performing the clustering all over again.

While CADC can effectively cluster similar entities together, we observe that the number of similar entities in the same cluster is still too large for making precise recommendations. In order to obtain more precise location recommendations, we introduce two additional steps. First, we propose an *in situ* recommendation refinement method to refine the recommendations for a particular activity performed by a specific user. Second, we introduce *behavioral* refinement, which classifies users into three classes, namely, *Pattern Users*, *Normal Users*, and *Travelers*, according to the sequences of locations that they visited in their activities, and filter out recommendations from different classes of users. Experimental results confirm that recommendations from Travelers are the most accurate among the three classes of users.

In addition to efficient clustering and recommendation refinement, an important issue in CLR is to define an effective ranking function to rank the recommendations according to their relevance. In this paper, we propose a relevance ranking scheme, called $LF \times IUF$ (i.e., Location Frequency $\times$ Inverse User Frequency), to rank the location recommendations. The $LF \times IUF$ ranking scheme is inspired by the well-known $TF \times IDF$ scheme in information retrieval. Basically, the interestingness of a location increases proportionally to the number of visits, but is offset by the number of users who have visited the location. The idea is motivated by the observation that a location visited by many users (e.g., a train station or an airport) may not be always an interesting location recommendation.

As discussed, the proposed CLR framework consists of several components, including CADC, CLM, in situ and behavioral refinement methods, and the $LF \times IUF$ ranking scheme. Figure 1 shows the components and the general processing flow in the CLR framework. We conduct an evaluation on the accuracy of the recommendations using a real-world GPS dataset. Experimental results confirm that our approach provides more accurate recommendations comparing to the existing methods based on clustering and collaborative filtering.

The main contributions of this paper are summarized below:

- Unlike methods based on User-Location matrix, we propose the Collaborative Location Model (CLM), which incorporates a new entity type, namely, *activity*, to exploit the relations between activities, users and locations. Activities bring rich temporal-spatial information into CLM to facilitate effective co-clustering.
- To avoid clustering the large trajectory dataset all over again when new GPS data arrives, we adopt a co-clustering method, CADC, which not only discovers similar objects by exploring their similarity ties, but also allows efficient incremental clustering in response to new incoming GPS trajectory data.
- We propose an in situ recommendation refinement method to refine the recommendations corresponding to a location in a particular activity issued by a specific user. We also introduce behavioral refinement to identify three classes of users, namely, *Pattern Users*, *Normal Users* and and *Travelers*, according to the sequence of locations that users visited in their activities, and study the recommendation effectiveness of these three classes of users.
- We devise a new relevance ranking scheme, $LF \times IUF$ (*Location Frequency, Inverse User Frequency*), for recommendation ranking of the locations.
- We implement CLR on the Google Android platform and evaluate the effectiveness of the recommendations using a real-world GPS dataset. The experimental results confirm that CLR provides more accurate recommendations compared to the existing methods.

2. RELATED WORK

Location recommendation is an important feature in location-based services (LBSs). It aims to provide location suggestions that a user may interest in. Different methods have been developed to discover similar locations for recommendation. They can be classified as *similarity-based* and *collaborative-filtering-based (i.e., CF-based)*. Similarity-based recommendations are generated based on the similarities of the candidate locations comparing to the active location (i.e., the current position of a user or a specific location specified by the user). The similarity metrics can be different among different systems. For example, Geodelic generates location recommendations according to the distances of the candidate locations to the active location [3]. Similarly, [5] provides information about the surrounding points of interests (POIs). On Twitter [4], users can identify and provide textual descriptions on their own POIs on a map. They can then search for interesting locations according to the supplied textual descriptions.

On the other hand, CF-based methods make use of the travel histories of a group of similar users (i.e., user-based collaborative filtering) or a set of similar locations (i.e., location-based collaborative filtering) to generate location recommendations. The recommendations are personalized because they are generated according to users who are similar to the active user, or locations that are frequently visited by the active user. In [11], a system was proposed to recommend shops based on users' location data histories. It employs an item-based collaborative filtering algorithm to identify shops for recommendation according to users' movements and geographical conditions of the city. Similarly, in [14], a location-based recommendation system was proposed to suggest locations based on users' current positions and their personal preferences.

More recently, Yu et. al. [16] proposed to mine interesting locations and classical travel sequences from multiple users' GPS trajectory data. They proposed a structure called Tree-Based Hierarchical Graph (TBHG) to model users' GPS trajectory data, and a HIT-based [8] influence model to compute the interestingness scores of locations and authority scores of users. The interestingness scores measure the representativeness of locations, while the authority scores measure the travel experience of users. The two scores are employed to predict a set of interesting locations from experienced users for the location recommendation.

We observe that most of the existing CF-based methods rely on a User-Location rating matrix. The predictions are obtained based on ratings of the same location by similar users (User-based CF), ratings of similar locations made by the same user (Location-based CF), or ratings of similar locations made by similar users (Fusion-based CF). The matrix involved in the CF process is usually huge. Thus, the number of possible candidates to be examined for recommendation can be numerous. Unlike the existing methods, our CLR method introduces *activity* as an additional dimension to obtain more accurate location recommendations by exploiting the temporal-spatial properties in activities. Moreover, we propose a refinement method to recommend locations for a specific user, activity, and location (i.e., according to the current activity performed at the current location by the active user). Therefore, the active user can discover locations that match his/her activities and preferences.

3. COMMUNITY LOCATION MODEL

GPS trajectory data is a sequence of unstructured time-stamped latitude/longitude pairs. In order to organize the unstructured trajectory data into a meaningful data structure to facilitate co-clustering and location recommendation, we propose the *Community Location Model* (CLM) to capture the relations between users, activities, and locations. CLM graphs are input to a co-clustering algorithm presented in Section 4 to discover refined subclusters for location recommendation. In Sections 3.1 and 3.2, we present our method to mine GPS routes and important locations from the GPS trajectory data. CLM is presented in Section 3.3.

3.1 GPS Route

UTC DATE	UTC TIME	LATITUDE	LONGITUDE	ALTITUDE
29/3/2010	2:32:36	22.335950	114.207337	314.721222
29/3/2010	2:32:41	22.335934	114.207505	333.723785
29/3/2010	2:32:46	22.335936	114.207506	335.357819
29/3/2010	2:32:51	22.335932	114.207568	342.328796
29/3/2010	2:32:56	22.335949	114.207525	338.480896
29/3/2010	2:33:01	22.335879	114.207505	336.999359
...	...	...	...	...

(a) Example GPS Points.

(b) Example GPS Route.

Figure 2: Example GPS Points and GPS Route.

A GPS log composes of latitudes, longitudes, speed and time, recorded at a regular time interval. As shown in Figure 2(a), a set of *GPS points* $P = p_1, p_2, ..., p_n$ for user u forms a *GPS route* r^u. Figure 2(b) shows the GPS route constructed using the GPS points in Figure 2(a).

From the GPS log, *stay points* representing geographic regions where activities happen can be discovered. We assume that if a user stays in a particular region for a certain period of time, the region is an important place, e.g., a restaurant that the user visited for breakfast, a classroom where the user attended a lecture, a cinema where the user watched a movie, etc. To detect a stay point from the GPS route, two thresholds, sp_t and sp_d, are used in our study:

- sp_t: the time threshold sp_t dictates the duration the user stays within a stay point s.
- sp_d: the distance threshold sp_d dictates the spatial region the user stays within a stay point s.

Figure 2(b) shows two stay points detected from the GPS route. Note that there could be many stay points in the middle of a route as long as the above thresholds are met.

3.2 Location History

The set of stay points $S = s_1, s_2, ..., s_m$ visited by a user u over a period of time forms the location history LH^u of u. Formally, we define $LH^u = s_1 \rightarrow_{\Delta t_1} s_2 \rightarrow_{\Delta t_2} ... \rightarrow_{\Delta t_{m-1}} s_m$, where Δt_k is the time between the leaving time at s_k and the arriving time at s_{k+1}. The location histories of different users can be quite different because of the wide variety of stay points that users can possibly visit. To alleviate this stay point sparsity problem, we employ the tree-based hierarchical graph (*TBHG*) [15] to model the location histories from different users. TBHG is a hierarchical graph containing different-sized location clusters at different levels. The first level consists of all of the users' stay points, which are then hierarchically clustered into location clusters using a divisive clustering algorithms with a distance threshold d. The deeper the level in TBHG, the finer (more precise) the location clusters in that level. Thus, stays points that are physically close to one another will be assigned to the same location clusters at different levels in TBHG. Figure 3 shows an example TBHG constructed in a divisive manner. We observe that the locations in the top level of TBHG are too general for location recommendation, since all of the stay points are grouped into large general location clusters. On the other hand, the locations in the deeper levels of TBHG are too fine-grained for location recommendation, since the location clusters are small resulting in only a few stay points.

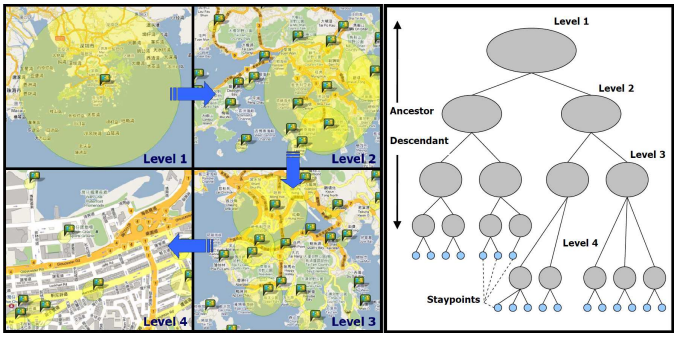
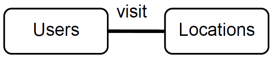


Figure 3: Example TBHG constructed in a divisive manner.

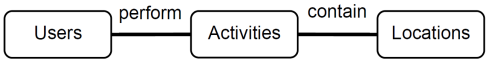
A user's GPS route r^u can be generated using TBHG by finding the locations clusters containing the stay points in r^u and creating edges between the location clusters according to the time sequence of the locations clusters visited by u. Let us first consider that two users are similar if they have visited similar locations. Since deeper levels of the TBHG contain more precise location clusters, if we use them to find similar users we will find fewer but more similar users. On the other hand, the large clusters at the top levels may bring both similar and dissimilar users into a few large clusters, yielding user clusters with low precisions but high recalls.

3.3 Community Location Model

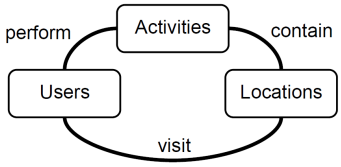
Since GPS trajectories are just sequences of unstructured time-stamped coordinates, we propose the *Community Location Model* (CLM) to organize trajectory data into a meaningful data structure to facilitate co-clustering. A novelty of CLM is the incorporation of activities into the model. Basically, an activity is a segment of GPS trajectory within a certain time frame. Figure 2(b) shows an example of daily activities (i.e., in 24 hours) of a student between her



(a) User-Location Model.



(b) User-Activity and Activity-Location Model.



(c) Community Location Model.

Figure 4: Different models for co-clustering.

home (i.e., Ngau Chi Wan) and her university (i.e., HKUST). CLM represents users, activities and locations as three entity types and the relations between the three entity types as binary relations (see Figure 4(c)). An instance of CLM is a User-Activity-Location tripartite graph (called CLM graph), rather than a User-Location matrix, capturing the relations between users, activities and locations in the model. A CLM graph consists of three disjoint node sets representing the sets of users, activities and locations. As described in Section 4, a CLM graph can be clustered with a co-cluster algorithm to form clusters of similar users, similar activities and similar locations. This allows refined clusters to be obtained for location recommendations (also see Section 1).

Figure 5(a) shows an example GPS trajectory data from 5 users with 12 activities on 10 different locations. The arrows in Figure 5(a) represent sequences of location visits in the GPS trajectory data. Figure 5(b) shows the relations between users, activities, and locations in the CLM graph.

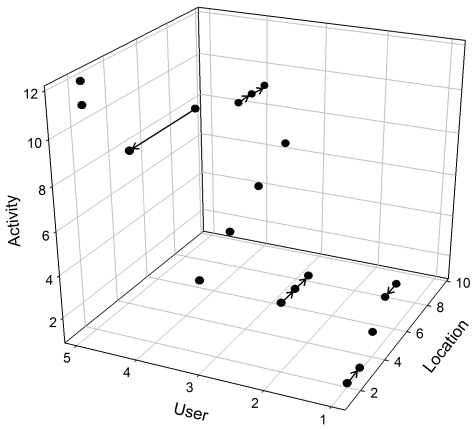
We observe that a model that only involves User and Location, as shown in Figure 4(a), is not accurate and effective for clustering relevant locations together, because it does not exploit the temporal-spatial information in Activity. Figure 4(b) shows an alternative model that captures User, Activity, and Location in two separate bipartite graphs. However, it misses the opportunity to exploit long-term spatial properties captured by the direct link between User and Location. On the other hand, CLM exploits all of the spatial properties, temporal-spatial properties, and long-term spatial properties in the co-clustering process. We observe that the three entity types (i.e., User, Activity, and Location) in CLM exhibit different properties that improve the effectiveness of clustering: 1) Location exhibits spatial properties as it contains geographical information; 2) Activity exhibits temporal-spatial properties as it is a temporal sequence of visited locations; and 3) User exhibits long-term spatial properties as different users would visit different locations due to their long term habits or geographical limits.

3.4 Similarity Fusion in CLM

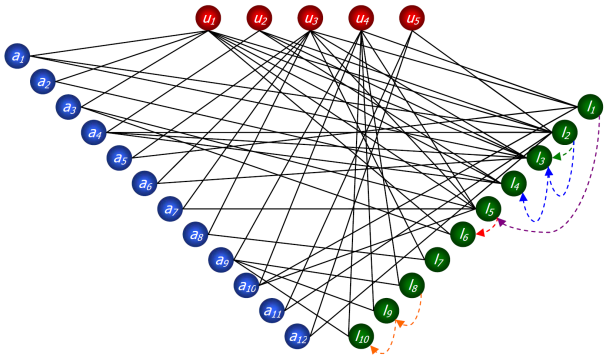
After constructing a CLM graph, we propose to model the similarity between different objects in the CLM graph as follows.

1. Two users are similar if they have similar activity patterns and have visited similar locations.
2. Two activities are similar if they take place in similar sequence of locations and performed by similar users.
3. Two locations are similar if they are visited by similar user and occurred in similar activities.

Formally, we propose the following similarity functions to compute the similarity between pair of users, pair of activities, and pair of locations:



(a) Example location histories from 5 users with 12 activities on 10 locations.



(b) An example tripartite CLM graph G_3.

Figure 5: An Example CLM graph.

$$sim(u_i,u_j)=\frac{\sum_{k=1}^{n}\frac{LCS(a_{(k,u_i)},a_{(k,u_j)})}{max(|a_{(k,u_i)}|,|a_{(k,u_j)}|)}}{n}\alpha_1+\frac{L_{u_i}\cdot L_{u_j}}{\parallel L_{u_i}\parallel\parallel L_{u_j}\parallel}(1-\alpha_1) \quad (1)$$

$$sim(a_i,a_j)=\frac{LCS(a_i,a_j)}{max(|a_i|,|a_j|)}\alpha_2+\frac{U_{a_i}\cdot U_{a_j}}{\parallel U_{a_i}\parallel\parallel U_{a_j}\parallel}(1-\alpha_2) \quad (2)$$

$$sim(l_i,l_j)=\frac{U_{l_i}\cdot U_{l_j}}{\parallel U_{l_i}\parallel\parallel U_{l_j}\parallel}\alpha_3+\frac{A_{l_i}\cdot A_{l_j}}{\parallel A_{l_i}\parallel\parallel A_{l_j}\parallel}(1-\alpha_3) \quad (3)$$

where $LCS(a_i,a_j)$ is the *longest common subsequence* of a_i and a_j, $a_{(k,u_i)}$ is the activity performed by u_i on day k, $|a_i|$ is the number of locations visited by a_i, L_{u_i} is a weight vector for the set of neighbor location nodes of the user node u_i, and the weight of a location neighbor node $l_{(k,u_i)}$ in L_{u_i} is the weight of the link connecting u_i and $l_{(k,u_i)}$ in the CLM. Similarly, U_{l_i} is a weight vector for the set of neighbor user nodes of the location node l_i, A_{l_i} is a weight vector for the set of neighbor activity nodes of the location node l_i, and A_{l_i} is a weight vector for the set of neighbor activity nodes of the location node l_i, and the weight of an activity neighbor node $a_{(k,l_i)}$ in the weight vector A_{l_i} is the weight of the link connecting l_i and $a_{(k,l_i)}$ in the CLM. Since LCS and cosine similarities lie between [0,1], it is easy to see that the resulting similarities in Equations (1), (2), and (3) also lie between [0,1].

4. DISCOVERING USER COMMUNITIES

In this section, we describe the co-clustering process on GPS trajectory data based on CLM to form clusters of similar users, locations and activities. Traditional agglomerative-only co-clustering algorithms require the tripartite graph to be clustered all over again when new data are inserted into the graph. Thus, it is not suitable

for dynamic environments. To alleviate this problem, we adopt the *Community-based Agglomerative-Divisive Clustering* (CADC) algorithm to iteratively cluster different types of entities simultaneously based on CLM.

To begin with, the location histories from all users are used to construct the tripartite CLM graph G_3 (see Figure 5(b)). CADC then clusters G_3 into a new tripartite graph G_3^C, in which the *Users* set is clustered into groups of similar users, and likewise the *Activities* and *Locations* sets are clustered, respectively, into groups of similar activities and similar locations. When new GPS trajectory data arrives, G_3^C is updated with the new data and then incrementally re-clustered using two separate phases, namely, the **agglomerative phase** and **divisive phase**, without re-clustering the whole CLM graph all over again. The agglomerative phase iteratively merges similar clusters, while the divisive phase splits large clusters into small ones to prevent clusters from growing without bound when new data arrives. Figure 6 shows an example of splitting one large user cluster into two smaller, more meaningful ones, when new incoming users arrive. CADC produces clusters of similar users, clusters of similar activities, and clusters of similar locations at the same time. These three types of similar entities will be used in the refinement phase (see Section 5) to obtain more precise subclusters for making recommendation.

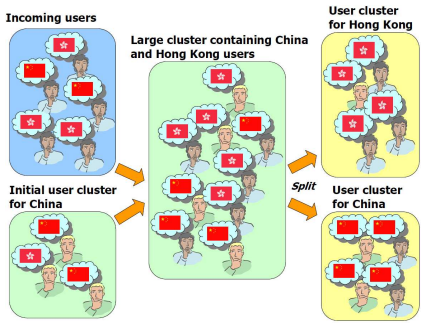


Figure 6: Split of a large cluster into two smaller clusters.

4.1 Agglomerative Phase

The agglomerative phase merges similar entities into clusters. Since we are dealing with tripartite CLM, we extend an agglomerative hierarchical clustering approach for bipartite data structure [6] for the tripartite CLM. Basically, an agglomerative algorithm starts with single entities, and successively merges entities/clusters together. The agglomerative phase of CADC iteratively merges the two most similar users, then the two most similar activities, and then the two most similar locations using the similarity definitions in Equations (1), (2), and (3), respectively. The procedure repeats until no new user, activity or location clusters can be formed by merging.

4.2 Divisive Phase

The divisive phase of CADC splits large clusters into smaller, but tighter ones. It is needed to maintain the coherence of the clusters and to restructure the existing clusters to facilitate incremental update of the affected clusters as new GPS trajectory data arrives. Basically, a divisive algorithm starts with large clusters, and recursively splits the clusters into smaller ones until no new clusters can be formed by splitting. One major problem in the divisive phase is to determine the minimum number of observations (i.e., evidence to perform a split) necessary for the phase to converge. To resolve the problem, the Hoeffding bound [7] is employed to ensure that after n independent observations of a real-valued random variable r with range R, and with confidence $1-\delta$ (where δ is the split threshold), the true mean of r is at least $\overline{r}-\epsilon$, where $\overline{r}$ is the observed mean of the samples and $\epsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}}$.

In the divisive phase, each cluster is assigned with a different ϵ, namely, ϵ_k. The distances between pair of users, pair of activities, and pair of locations are defined, respectively, as the inverse of the similarities defined in Equations (1), (2), and (3).

$$d(u_i,u_j) = (1 - \frac{\sum_{k=1}^{n} \frac{LCS(a_{(k,u_i)},a_{(k,u_j)})}{max(|a_{(k,u_i)}|,|a_{(k,u_j)}|)}}{n})\beta_1 + \sqrt{\sum_{k=1}^{m}(l_{(k,u_i)} - l_{(k,u_j)})^2(1-\beta_1)} \quad (4)$$

$$d(a_i,a_j) = \frac{LCS(a_i,a_j)}{max(|a_i|,|a_j|)}\beta_2 + \sqrt{\sum_{k=1}^{m}(u_{(k,a_i)} - u_{(k,a_j)})^2(1-\beta_2)} \quad (5)$$

$$d(l_i,l_j) = \sqrt{\sum_{k=1}^{n}(u_{(k,l_i)} - u_{(k,l_j)})^2\beta_3} + \sqrt{\sum_{k=1}^{m}(a_{(k,l_i)} - a_{(k,l_j)})^2(1-\beta_3)} \quad (6)$$

where $LCS(a_i,a_j)$ is the *longest common subsequence* of a_i and a_j, $a_{(k,u_i)}$ is the activity performed by u_i on day k, $|a_i|$ is the number of locations visited by a_i, $l_{(k,u_i)} \in L_{u_i}$ is the weight of the link connecting u_i and $l_{(k,u_i)}$. Similarly, $u_{(k,l_i)} \in U_{l_i}$ and $a_{(k,l_i)} \in A_{l_i}$.

Assume that two pairs of nodes $d1_n = d(n_i,n_j)$ and $d2_n = d(n_k,n_l)$ are the top-most and second top-most dissimilar nodes in a cluster (based on the distance Equation 4), (5), or (6). Further assume that $\triangle d = d(n_i,n_j) - d(n_k,n_l)$, if $\triangle d > \epsilon_k$, with probability $1-\delta$, the differences between $d(n_i,n_j)$ and $d(n_k,n_l)$ is larger than zero, and pick (n_i,n_j) as the boundary of the cluster when applying Hoeffding bound with $\triangle d$. In the divisive phase, n_i and n_j are selected as the pivots for the splitting, and the clusters are split according to the statistical confidence given by Hoeffding bound. Figure 7 shows an example of splitting a large cluster into two smaller ones.

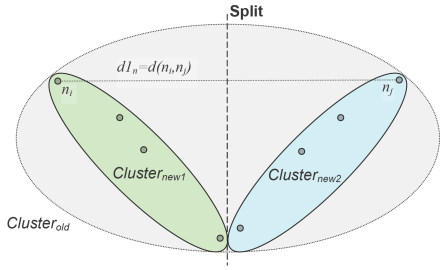


Figure 7: Split $Cluster_{old}$ into $Cluster_{new1}$ and $Cluster_{new2}$.

4.3 The Clustering Algorithm

The clustering algorithm iteratively merges and splits nodes in G_3 until the termination condition is reached. Then the clusters in the final CLM graph G_3^C are output. When new location histories arrive, the new users, activities, and locations are added as singleton nodes to G_3^C and the links between the new nodes are also added to G_3^C. The new graph G_3' is then served as input to the clustering algorithm again. The new nodes are grouped to the correct clusters by the agglomerative phase. If a particular cluster becomes too large because too many new nodes are added to the cluster, the divisive phase will divide the cluster into smaller ones according to the statistical confidence given by Hoeffding bound. These steps are repeated until the termination condition is reached, and the algorithm outputs the final CLM graph.

5. COMMUNITY REFINEMENT

In order to provide high-quality location recommendations to the users, we further refine clusters of users, activities, and locations generated by our CADC co-clustering algorithm. The community refinement component of CLR generates a number of subclusters by intersecting the user, activity, and location clusters in the clustered tripartite graph. Assume that the active user u_a is visiting location l_c when she is performing activity a_b, then we can obtain from the clustered tripartite graph a set of users U_{u_a} similar to u_a, a set of activities A_{a_b} similar to a_b, and a set of locations L_{l_c} similar to l_c. By using the clusters U_{u_a}, A_{a_b}, and L_{l_c} in three different dimensions, we can obtain the following subclusters for location recommendations.

In Situ Refined Subclusters:

- $CADC(L)$: Set of locations similar to location l_c.
- $CADC(U)$: Set of locations visited by users similar to u_a.
- $CADC(A)$: Set of locations where activities similar to a_b are performed.
- $CADC(U, L)$: Set of locations similar to l_c, and visited by users similar to u_a.
- $CADC(U, A)$: Set of locations where activities similar to a_b are performed, and visited by users similar to u_a.
- $CADC(A, L)$: Set of locations similar to l_c where activities similar to a_b are performed.
- $CADC(U, A, L)$: Set of locations similar to l_c, activities similar to a_b are performed, and visited by users similar to u_a.

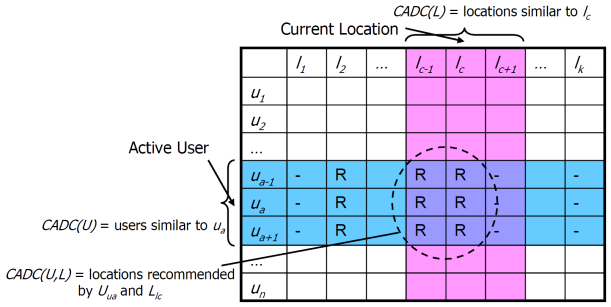


Figure 8: Intersection of $CADC(U)$ and $CADC(L)$ to obtain $CADC(U, L)$.

Figure 8 shows an example of intersecting $CADC(U)$ and $CADC(L)$ to obtain $CADC(U, L)$. $CADC(L)$ can be easily obtained by getting the locations in the clustered CLM graph that are in the same cluster as the active location l_c. However, for $CADC(U)$, we need to get the locations visited by users that are similar to the active user u_a. Thus, we need to first obtain the set of users, $U(u_a)$ and then retrieve locations that have been visited by users in $U(u_a)$, which can be easily obtained by finding the locations nodes that are connected to $U(u_a)$ in the clustered CLM graph. After $CADC(U)$ and $CADC(L)$ are obtained, the locations resulted from $CADC(U)$ are intersected with the locations resulted from $CADC(L)$ to obtain $CADC(U, L)$ as shown in Figure 8.

After determining the subclusters by intersecting $CADC(U)$, $CADC(A)$, and $CADC(L)$, the top k most visited locations in a subclusters will be recommended to the user. The top k locations are picked according to the *Location Frequency (LF)* and *Inverse User Frequency (IUF)*. IUF is taken into account because we find that a location visited by most users may be a general well-known location which is not interesting to the users, and thus its score is penalized by IUF. The following equation is adopted from the well-known $TF \times IDF$ term weighting scheme in information retrieval with LF and IUF replacing, respectively, TF and IDF:

$$score(l) = LF(l) \times IUF(l) \tag{7}$$

where $IUF(l) = \log_2 \frac{\text{total number of users in a subcluster}}{\text{number of users visited} l}$. After computing the scores for all of the locations in a subclusters using Equation (7), the top k locations in a subcluster are recommended to the user.

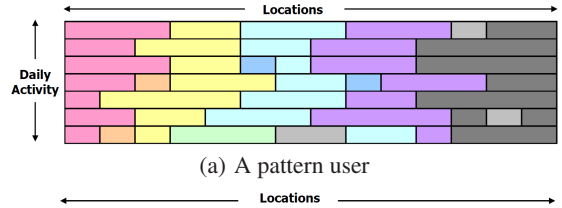
The in situ subclusters can help filtering out useful important locations for collaborative location recommendations. However, the size of a refined subcluster can still be large, thus resulting in many locations for recommendations. Therefore, we further classify the users into three classes, namely *Pattern Users*, *Normal Users*, and *Travelers*, according to the sequence of locations that they visited in their daily activities, in order to filter out location recommendations from different classes of users. We use the location entropy $S_L(u)$ to measure the randomness of the locations being visited by a user u.

$$S_L(u) = -\sum_{i=1}^{k} p(l_i) \log p(l_i) \tag{8}$$

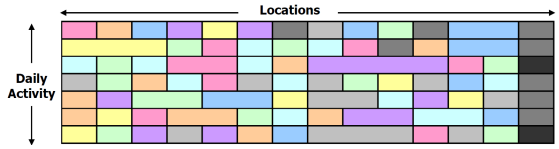
where k is the total number of locations $L = l_1, l_2, ..., l_k$ that u has visited, $|l_i|$ is the number of daily activities containing the location l_i, $|L| = |l_1| + |l_2| + ... + |l_k|$, and $p(l_i) = \frac{|l_i|}{L}$.

After computing $S_L(u)$ for each user, we can then employ K-Means to classify users into the three classes as follows.

- **Pattern Users:** Users with low location entropies, i.e., they repeat what they do daily, thus having small randomness in the locations they visited.
- **Normal Users:** Users with higher location entropies. They visit more different locations comparing to the pattern users.
- **Travelers:** Users with high location entropies, meaning that they visit many different locations everyday. They can be considered knowledgeable users, since they have good knowledge about different locations.



(a) A pattern user



(b) A traveler

Figure 9: Example activities from pattern user and traveler.

Figures 9(a) and 9(b) show example activities from a pattern user and a traveler, respectively, where different colors represent different locations that the users have visited. After classifying the users into Pattern Users, Normal Users, and Travelers, we can then employ them to refine the in situ subclusters into the following behavioral subclusters, each of which only contain one type of users.

Behaviorally Refined Subclusters:

- $PTN(U)$: Set of locations visited by **Pattern** users similar to u_a.
- $PTN(U, L)$: Set of locations similar to l_c and visited by **Pattern** users similar to u_a.
- $PTN(U, A)$: Set of locations performed in activities similar to a_b and visited by **Pattern** users similar to u_a.
- $PTN(U, A, L)$: Set of locations similar to l_c and performed in activities similar to a_b and visited by **Pattern** users similar to u_a.
- $TRL(U)$: Set of locations visited by **Travelers** similar to u_a.

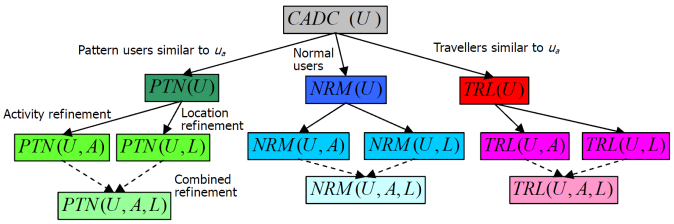


Figure 10: Recommendations from different user classes.

Figure 11: GPS devices used for GPS trajectory data collection.

- $TRL(U, L)$: Set of locations similar to l_c and visited by **Travelers** similar to u_a.
- $TRL(U, A)$: Set of locations performed in activities similar to a_b and visited by **Travelers** similar to u_a.
- $TRL(U, A, L)$: Set of locations similar to l_c and performed in activities similar to a_b and visited by **Travelers** similar to u_a.

After obtaining the refined subclusters according to u_a, a_b, and l_c, the top k locations with the highest $LF \times IUF$ scores in the refined subclusters will be recommended to the active user u_a. In the experiments, we will evaluate and compare the effectiveness of the location recommendations obtained from different in situ and behaviorally refined communities. Figure 10 shows some of the top k location recommendations L^k from different user communities.

6. EXPERIMENTAL RESULTS

In this section, we first present the experimental setup for collecting the GPS trajectory data used in the evaluation of CLR. We then present the methodology used in the evaluation. Then, we evaluate the clustering effectiveness of CADC, the performance of the stay point and location clustering, and the effectiveness of location suggestions.

6.1 Experimental Setup

The GPS trajectory data was collected from 50 different users using the GPS loggers as shown in Figure 11. The users' outdoor trajectories were recorded from September 2009 to May 2010. The GPS loggers were set to record GPS coordinates every second. GPS trajectory data was mainly collected in Hong Kong and China. Figure 12 shows some of the trajectory data collected for the experiments.

6.2 Evaluation Metric

In the experiments, users are asked to evaluate the location recommendations generated by CLR. The top 10 location recommendations generated by CLR are displayed to the users, who are then required to provide ratings to the recommendations according to the criteria as shown in Table 1. Since our top 10 location recommendations are displayed in a ranked list, we employ *normalized discounted cumulative gain (nDCG)* to measure the accuracy of the recommendations. The ratings obtained from a user are used to generate the user's ideal ranking, which are compared against the top 10 location recommendations generated by CLR to obtain the $nDCG$ value of the CLR recommendations. The $nDCG$ of a top 10 recommendation list is computed using the following equations.

Figure 12: Sample GPS trajectory data collected for the experiments.

Table 1: The rating on the location representativeness

Ratings	Meaning
3	I intended to visit the location
2	I would visit the location if I pass by
1	I have a little chance to visit the location
0	This location is obviously not of my interest

$$DCG = rel_1 + \sum_{i=2}^{10} \frac{rel_i}{log_2 i} \quad (9) \qquad nDCG = \frac{DCG}{IDCG} \quad (10)$$

In CLR, we only consider the top 10 recommendations. Thus, the DCG value of a recommendation list can be computed using Equation (9), where rel_i is the relevance rating of the i-th location in the recommendation list. Then, we normalize the DCG value of the recommendation list with respect to the ideal ranking using Equation (10), where DCG is the DCG value computed from a recommendation list, and $IDCG$ is the DCG value computed from the user's ideal ranking. For example, if the ratings on the top 10 recommendations are [3,2,1,3,2,2,1,0,0,1], then the corresponding $nDCG$ is computed as follow.

$$nDCG = \frac{3 + 2/log2 + 1/log3 + 3/log4 + 2/log5 + 2/log6 + 1/log7 + 1/log10}{3 + 3/log2 + 2/log3 + 2/log4 + 2/log5 + 1/log6 + 1/log7 + 1/log8} = 0.92 \quad (11)$$

$nDCG$ measures the accuracy of a ranked recommendation list generated by CLR. Thus, the higher the $nDCG$ is, the more accurate the recommendations are (i.e., relevant locations are ranked higher in the recommendation list).

6.3 CADC Location Recommendations

We compare CADC against a few state-of-the-art methods, including *Distance*, *SimUser*, *Memory-Based Collaborative Filtering (MemCF)*, *Model-Based Collaborative Filtering (ModelCF)*, and *HITS*.[1] The *Distance* method returns the top 10 locations that are closest to the active location as the location recommendations. The *SimUser* method represents each location with a user vector based on the users who visited the location. It recommends the top 10 locations which have user vectors most similar to the active location's user vector. The *Memory-Based Collaborative Filtering (MemCF)* method is based on the method presented in [13]. We first create a User-Activity-Location matrix to model the relationships between users, activities, and locations. For each missing entry in the matrix, we extract a set of top N similar users, top N similar activities, and top N similar locations, and then use the ratings from the top N entities to predict the value of the missing entry probabilistically. After predicting all of the missing entries, it then recommends the

[1]Hereafter, CADC refers to $CADC(U, A, L)$ as discussed in Section 5 unless stated otherwise.

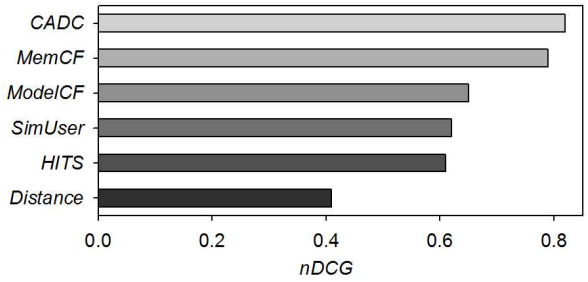


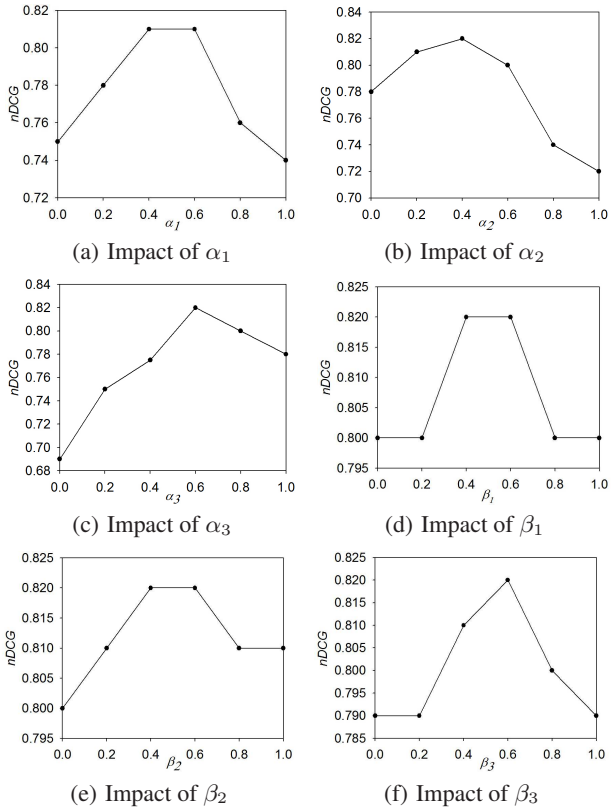
Figure 13: $nDCG$ for Distance, HITS, SimUser, MemCF, ModelCF, and CADC.

top 10 locations that are visited by similar users and where similar activities are performed. In the *Model-Based Collaborative Filtering (ModelCF)* method, we also create a User-Activity-Location matrix but employ a *Higher-Order Singular Value Decomposition (HOSVD)* [12] to predict the missing entries in the matrix. We then return as recommendation the top 10 locations that are visited by similar users and where similar activities are performed. Finally, we also include in the evaluation the *HITS* method proposed in [16]. The HITS method determines a user's authority value using the authority proposition in the hyperlink environment in [8]. The top N users with the highest authority values are picked, and the top 10 locations visited by the top N authority users are returned as location recommendations.

Figure 13 shows the $nDCG$ of CADC and the baseline methods. We observe that the Distance method performs the worst ($nDCG = 0.41$), because the locations that are closest to the active location may not always match the user's spatial interests (i.e., preferences according to the *user* dimension) and the temporal-spatial spatial interests (i.e. preferences according to the *activity* dimension). The HITS method relies only on the recommendations from users with high authority scores. It does not take the user's spatial interests into account. Thus, it performs only slightly better ($nDCG = 0.61$) than the Distance method. The SimUser method considers the user's long term spatial interests. As a result, it yields $nDCG = 0.62$, which is better comparing to the Distance and HITS methods. Both the MemCF and ModelCF methods employ the User-Activity-Location matrix in the collaborative filtering process, thus capturing both the user's long term spatial interests and the temporal-spatial spatial interests. MemCF ($nDCG = 0.79$) performs better than ModelCF ($nDCG = 0.65$), because it performs predictions based on only the truly similar entities (i.e., the top N similar users, top N similar activities, and top N similar locations). Finally, we observe that CADC ($nDCG = 0.82$) performs the best comparing to the other five baseline methods, because the co-clustering method can effectively explore the ties modeled in CLM to iteratively group different types of similar entities simultaneously. The similarity scores are optimized by exploiting similarity fusion to strike a good balance between the similarity scores from different dimensions, thus producing more coherent clusters of similar locations.

6.3.1 CADC Merge Fusion Thresholds α_1, α_2, α_3

We then study the impact of the merge thresholds on CADC's performance by varying the α_1, α_2, and α_3 merge thresholds in Equations (1), (2), and (3), respectively, from 0 to 1 (with 0.1 increments). We obtain the $nDCG$ values under different settings of merge thresholds. When we vary one of the merge thresholds, the other two thresholds are fixed at the optimal values so that only one of the merge thresholds would influence the results. Figure 14(d) shows the $nDCG$ values obtained at different user merge thresholds α_1 from 0 to 1, with α_2 and α_3 fixed at their optimal values (0.4 and 0.6, respectively). We observe that the best $nDCG$ is obtained when $\alpha_1 = 0.5$. This shows that the activity dimension and the location dimension are equally important for identifying similar users. We repeat the experiment on the activity and location merge thresholds (i.e., α_2 and α_3). Similarly, we vary α_2 from 0 to 1, with α_1 and α_3 fixed at their optimal values 0.5 and 0.6, respectively. We observe that the optimal $nDCG$ is obtained when $\alpha_2 = 0.4$. This shows that the sequence of visited locations is more important than the user dimension for identifying similar activities, meaning that two activities performed by two similar users may not always have high similarity to one another (e.g., users u_1 and u_2 are similar because they both live in Clear Water Bay, Hong Kong). However, u_1 can be a housewife, while u_2 can be a student, and they will perform different activities). Finally, we repeat the experiment on α_3 by fixing α_1 and α_2 at 0.5 and 0.4, respectively. The optimal $nDCG$ is achieved when α_3 is at 0.6. This shows that the activity dimension (i.e., the most updated temporal-spatial preferences) is more important than the user dimension (i.e., the long term spatial preferences).



(a) Impact of α_1 (b) Impact of α_2

(c) Impact of α_3 (d) Impact of β_1

(e) Impact of β_2 (f) Impact of β_3

Figure 14: Impact of the merge fusion thresholds α_1, α_2, α_3, and the split fusion thresholds β_1, β_2, and β_3 on CADC.

6.3.2 CADC Split Fusion Thresholds β_1, β_2, β_3

We also study the impact of the split thresholds in CADC by varying the three split thresholds (β_1, β_2, and β_3 in Equations (4), (5), and (6), respectively), from 0 to 1 (with 0.1 increments). As before, when we vary one of the split thresholds, the other two split thresholds are fixed at their optimal values. Figure 14 shows the $nDCG$ values at different split thresholds from 0 to 1. We observe that the $nDCG$ values do not vary much when split thresholds change. This is because we start the CADC algorithm with

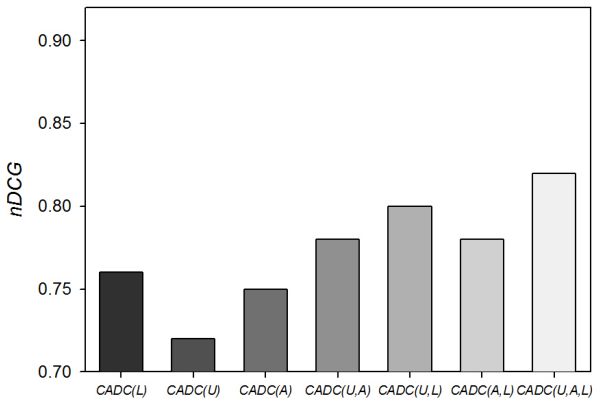


Figure 15: $nDCG$ for the in situ refined communities.

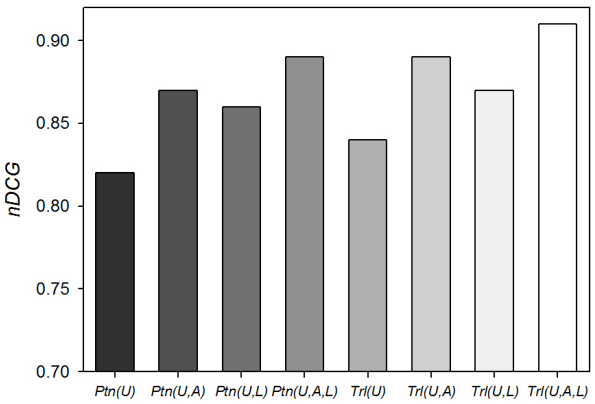


Figure 16: $nDCG$ for the behaviorally refined communities.

the agglomerative phase to group individual objects together, and we require two objects to be highly similar before they are merged into the same node. Thus, in the divisive phase, only a few splits are required to split the oversize clusters into smaller clusters. We observe that the best $nDCG$ value is obtained when $\beta_1 = 0.5$, $\beta_2 = 0.5$, and $\beta_3 = 0.6$.

6.4 Refined Location Recommendations

6.4.1 *In Situ Refined Recommendations*

We also compare the quality of the location recommendations generated from the refined communities. We first evaluate the in situ refined communities, $CADC(U)$, $CADC(A)$, $CADC(L)$, $CADC(U, A)$, $CADC(U, L)$, $CADC(A, L)$, and $CADC(U, A, L)$, described in Section 5. Figure 15 shows the $nDCG$ values obtained for the in situ refined communities. We observe that the location recommendations generated using only one of the dimensions (i.e., $CADC(U)$ based on only the user dimension, $CADC(A)$ based on only the activity dimension, and $CADC(L)$ based on only the location dimension) are the worst. The $nDCG$ for $CADC(U)$, $CADC(A)$, and $CADC(L)$ are only 0.76, 0.72, and 0.75, respectively. The location recommendations generated using two dimensions (i.e., $CADC(U, A)$ based on the user and activity dimensions, $CADC(U, L)$ based on only the user and location dimensions, and $CADC(A, L)$ based on only the activity and location dimensions) are better than those obtained using one dimension only. $CADC(U, A)$, $CADC(U, L)$, and $CADC(A, L)$ yield $nDCG$ of 0.78, 0.80, and 0.78, respectively. Finally, the location recommendations generated using all the three dimensions (i.e., $CADC(U, A, L)$) are the best. $CADC(U, A, L)$ yields $nDCG = 0.82$, which is the highest value among all of the in situ refined user communities.

6.4.2 *Behaviorally Refined Recommendations*

We also evaluate the quality of the location recommendations generated from using the behaviorally refined communities, $PTN(U)$, $PTN(U, A)$, $PTN(U, L)$, $PTN(U, A, L)$, $TRL(U)$, $TRL(U, A)$, $TRL(U, L)$, and $TRL(U, A, L)$, described in Section 5. Figure 16 shows the $nDCG$ values obtained from the behaviorally refined communities. We observe that the location recommendations obtained from Travelers are better than those obtained from Pattern Users. Consistent with the results obtained for the in situ refined communities, location recommendations generated using only one dimension (i.e., $PTN(U)$ and $TRL(U)$) are not as good as those generated using two dimensions (i.e., $PTN(U, A)$, $TRL(U, A)$, $PTN(U, L)$, and $TRL(U, L)$), and the recommendations generated using all three dimensions (i.e., $PTN(U, A, L)$ and $TRL(U, A, L)$) yield the best $nDCG$ (0.89 and 0.91, respectively).

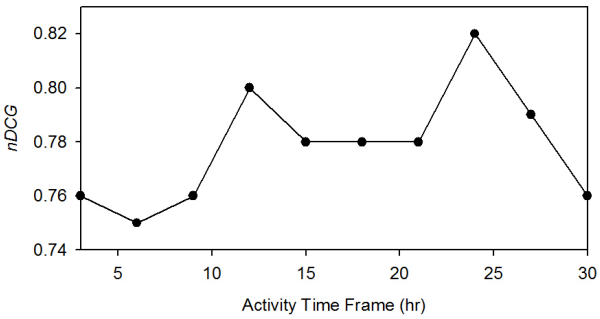


Figure 17: $nDCG$ of CADC with different activity time frames.

Table 2 shows the top 3, 5, and 10 average ratings obtained from the behaviorally refined communities. We observe that the top 5 and 10 average ratings obtained from Travelers are better than those obtained from Pattern Users. This is because Pattern Users usually only visit a few locations in regular patterns. Thus, only a few locations recommended by Pattern Users are useful to other users. On the other hand, Travelers usually travel many different locations. Thus, many of their recommendations are useful to other users, because they are knowledgeable users with experiences in different locations.

6.5 Impact of Activity Length on CLR

As discussed in Section 3.3, an activity is a segment of GPS trajectory within a certain time frame. The length of the time frame can be adjusted to include different numbers of locations associated with an activity (i.e., the longer the time frame, the more locations are associated with an activity). To evaluate the impact of the activity time frame on the accuracy of location recommendations, we repeat the $nDCG$ evaluation on the $CADC(U, A, L)$ method with different time frames. Figure 17 shows the $nDCG$ values of CADC with different activity time frames. We observe that if the time frame is too short (e.g. 3 or 6 hours), the resulting activities will only be associated with a small number of locations. Hence, similar activities cannot be discovered effectively because of the low location overlaps between activities, thus yielding lower $nDCG$ comparing to the optimal value obtained at a 24-hour time frame. On the other hand, if the time frame is too long (e.g. 30 hours), the resulting activities will become too general because of their associations with many different locations. The general activities may bring dissimilar locations together, thus worsening the performance. Finally, we observe that the activity time frame should be either 12 or 24 hours (half-a-day or daily activity), in order to achieve the best performance.

Table 2: The top 3, 5, 10 average ratings for the behaviorally refined communities

	$PTN(U)$	$PTN(U,A)$	$PTN(U,L)$	$PTN(U,A,L)$	$TRL(U)$	$TRL(U,A)$	$TRL(U,L)$	$TRL(U,A,L)$
Top 3	2.34	2.36	2.40	2.43	2.46	2.48	2.48	2.54
Top 5	1.94	1.98	1.96	1.98	2.23	2.25	2.25	2.28
Top 10	1.72	1.72	1.73	1.80	2.08	2.12	2.13	2.18

Table 3: Impact of sp_t on Stay Point Detection

sp_t	# Points	CADC	MemCF	ModelCF	HITS
5mins	2214	0.79	0.75	0.61	0.57
15mins	1297	0.78	0.76	0.62	0.58
30mins	**762**	**0.82**	**0.79**	**0.65**	**0.61**
45mins	497	0.76	0.73	0.64	0.61
60mins	245	0.72	0.71	0.57	0.55

Table 4: Impact of sp_d on Stay Point Detection

sp_t	# Points	CADC	MemCF	ModelCF	HITS
300meters	1863	0.78	0.75	0.63	0.56
500 meters	**762**	**0.82**	**0.79**	**0.65**	**0.61**
800meters	467	0.73	0.71	0.64	0.61

6.6 Impact of sp_t and sp_d on CLR

We also study the influence of the time and distance thresholds (i.e., sp_t and sp_d) on location recommendations. If sp_t and sp_d are too small, we may get locations that lack representativeness (e.g., bus stop, car park, and train station). On the other hand, if sp_t and sp_d are too large, the locations may be too large and too general, leading to difficulty in determining points of interests. To find the optimal time threshold sp_t, we vary sp_t from 5 mins to 60 mins. When we vary sp_t, sp_d is fixed at its optimal value (i.e., sp_d = 500 meters). Table 3 shows the $nDCG$ values at different sp_t. We observe that the larger the sp_t, the fewer the number of stay points detected from the users' GPS data. Moreover, CADC yields the best $nDCG$ when $sp_t = 30$ mins by avoiding locations that are too general or too precise.

We repeat the experiment by varying sp_d from 300 meters to 800 meters. When we vary sp_d, sp_t is fixed at its optimal value (i.e., $sp_t = 30$ mins). Table 4 shows the $nDCG$ values at different sp_d. Again, to avoid locations that are too general or too limited, we need to set $sp_d = 500$ meters in order to achieve the best $nDCG$. We observe that the larger the sp_d, the fewer the number of stay points detected from the users' GPS data. Moreover, CADC yields the best $nDCG$ when $sp_t = 500$ meters by avoiding locations that are too general or too precise.

7. CONCLUSIONS

In this paper, we propose the Collaborative Location Recommendation (CLR) framework, which makes location recommendations based on users' GPS trajectory data. We introduce the community location model *(CLM)*, which represents a User-Activity-Location tripartite graph, and employ the CADC co-clustering algorithm to exploit the rich activity information embedded in GPS trajectory data. The idea of online recommendation refinement is also introduced to refine the clusters of similar locations resulted from CADC for a particular activity issued by a specific user in order to obtain high quality location recommendations. Experimental results confirm that our approach can provide more accurate and refined recommendations comparing to the existing methods based on clustering or collaborative filtering.

As for the future work, we plan to incorporate more semantic information in the clustering process to further improve the quality of location recommendations. For example, on Twitter [4], users may provide textual descriptions on their own POIs on a map. Apart from the GPS coordinates, the textual descriptions can also serve as additional information describing the POIs for clustering.

8. REFERENCES

[1] Geolife. http://research.microsoft.com/en-us/projects/geolife/.

[2] Google map. http://maps.google.com/.

[3] Sherpa. http://www.geodelic.com/sherpa/.

[4] Twitter. http://www.twitter.com/.

[5] G. D. Abowd, C. G. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton. Cyberguide: a mobile context-aware tour guide. *Wireless Networks*, 3(5), 1997.

[6] D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *Proc. of ACM SIGKDD Conference*, 2000.

[7] W. Hoeffding. Probability inequalities for sums of bounded random variables. *JASA*, 58(301), 1963.

[8] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *JACM*, 46(5), 1999.

[9] Q. Li, Y. Zheng, X. Xie, Y. Chen, W. Liu, and W.-Y. Ma. Mining user similarity based on location history. In *Proc. of SIGSPATIAL GIS Conference*, 2008.

[10] M.-H. Park, J.-H. Hong, and S.-B. Cho. Location-based recommendation system using bayesian user's preference model in mobile devices. In *Proc. of UIC Conference*, 2007.

[11] Y. Takeuchi and M. Sugimoto. Cityvoyager: An outdoor recommendation system based on user location history. In *Proc. of UIC Conference*, 2006.

[12] J. tao Sun, H.-J. Zeng, H. Liu, and Y. Lu. Cubesvd: A novel approach to personalized web search. In *Proc. of WWW Conference*, 2005.

[13] J. Wang, A. P. D. Vries, and M. J. T. Reinders. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *Proc. of SIGIR Conference*, 2006.

[14] C.-C. Yu and H.-P. Chang. Personalized location-based recommendation services for tour planning in mobile tourism applications. In *Proc. of EC-Web Conference*, 2009.

[15] V. W. Zheng, Y. Zheng, X. Xie, and Q. Yang. Collaborative location and activity recommendations with gps history data. In *Proc. of WWW Conference*, 2010.

[16] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma. Mining interesting locations and travel sequences from gps trajectories. In *Proc. of WWW Conference*, 2009.