An Approach to Compute User Similarity for GPS Applications

Pramit Mazumdar[1a], Bidyut Kr. Patra[a], Russell Lock[b], Korra Sathya Babu[a]

[a] *National Institute of Technology Rourkela, Odisha, India*
[b] *Loughborough University, Leicestershire, United Kingdom*

Abstract

The proliferation of GPS enabled devices has led people to share locations both consciously and unconsciously. Large spatio-temporal data comprising of shared locations and whereabouts are now being routinely collected for analysis. As user movements are generally driven by their interests, so mining these mobility patterns can reveal commonalities between a pair of users. In this paper, we present a framework for mining the published trajectories to identify patterns in user mobility. In this framework, we extract the locations where a user stays for a period of time popularly known as the stay points. These stay points help to identify the interests of a user. The statistics of pattern and check-in distributions over the GPS data are used to formulate similarity measures for finding K-nearest neighbors of an active user. In this work, we categorize the neighbors into three groups namely strongly similar, closely similar and weakly similar. We introduce three similarity measures to determine them, one for each of the categories. We perform experiments on a real-world GPS log data to find the similarity scores between a pair of users and subsequently find the effective K-neighbors. Experimental results show that our proposed metric outperforms existing metrics in literature.

Keywords:
GPS data, User Similarity, Semantic Location, Trajectory Pattern Mining

1. Introduction

The Twenty-First century is witnessing a meteoric rise in user generated content, supporting a myriad of different interests through both specialists and general online social networks. Popular social networks such as Facebook, Twitter, Whatsapp, etc. are commonly being used as communication and sharing platforms. In addition to this, the advancement of mobile technology has provided handheld devices with powerful and precise GPS functionality to log exact user positions either on user's request or increasingly without their conscious decision. This has changed the way people share their check-in locations along with

[1] Corresponding author

Table 1: Sample trajectory data of five users recorded over a given week. Here any entry L_{72} denotes that user-2 has checked-in at location-7. The symbol '$\rightarrow$' depicts the sequence of locations visited.

Days	Recorded Trajectory
Monday	$L_{11} \rightarrow L_{21} \rightarrow L_{31}; L_{72} \rightarrow L_{82}; L_{43} \rightarrow L_{53}; L_{14} \rightarrow L_{24} \rightarrow L_{34}; L_{15} \rightarrow L_{25} \rightarrow L_{35}$
Tuesday	$L_{41}; L_{72} \rightarrow L_{82}; L_{13} \rightarrow L_{23} \rightarrow L_{33}; L_{44} \rightarrow L_{54} \rightarrow L_{64}; L_{45}$
Wednesday	$L_{11} \rightarrow L_{21} \rightarrow L_{31}; L_{72}; L_{13} \rightarrow L_{23} \rightarrow L_{33}; L_{44}; L_{15} \rightarrow L_{25} \rightarrow L_{35}$
Thursday	$L_{51} \rightarrow L_{61}; L_{12} \rightarrow L_{22}; L_{54} \rightarrow L_{64}$
Friday	$L_{41}; L_{72} \rightarrow L_{82} \rightarrow L_{92}; L_{43}; L_{14} \rightarrow L_{24} \rightarrow L_{34}; L_{45}$
Saturday	$L_{51} \rightarrow L_{61}; L_{12} \rightarrow L_{22}; L_{73} \rightarrow L_{83} \rightarrow L_{93}; L_{55} \rightarrow L_{65}$
Sunday	$L_{51} \rightarrow L_{61}; L_{12} \rightarrow L_{22}; L_{73} \rightarrow L_{83}; L_{55} \rightarrow L_{65}$

their real time experiences.

Irrespective of the way check-ins are performed in social networks, a user's movement can also be identified from the GPS logs of their handheld device and other navigation systems. For the remainder of this paper, the term 'check-in' is exclusively used to represent a user's precise location of presence at a defined timestamp. These check-ins bridge the gap between the real world and the online world. The locations recorded over a period of time for any user is usually termed as his/her historical trajectory. Check-in locations can be categorized into a number of pre-defined classes or categories such as Restaurant, Stadium, Hospital, etc. The category of locations visited by users have been used by researchers to reveal their interests and behavior [1, 2, 3]. The mining of historical data can also be used to provide customized services including location prediction [4], location recommendations [5], identifying hidden social links [6], friend recommendations [7], community detection [8], etc. For all of the above services, finding neighbors of a user who exhibit common interests and behavior is an important step. Two users are said to be neighbors if they have common mobility profiles. While analyzing user mobility, instead of using physical locations, their corresponding categories are used to identify actual user's interests. This helps to trace the movement of users who are geographically apart, but share similar interests. Similarity metrics available in literature only consider the maximum length of common patterns along with their frequency of occurrences [9, 10, 11]. However, existing similarity metrics fail to compute effective neighbors in a few scenarios as depicted in Table 1.

The example in Table 1 shows the trajectory information of five users (u_1, u_2, u_3, u_4 and u_5) over a week in a city. For instance on Thursday, u_1, u_2 and u_4 visit locations $\langle L_5 \rightarrow L_6 \rangle$, $\langle L_1 \rightarrow L_2 \rangle$ and $\langle L_5 \rightarrow L_6 \rangle$ in sequence, respectively. From the available data many common patterns of movement can be identified. The goal is to find the nearest neighbor of u_1 on the basis of the mobility patterns. Initially the common mobility patterns between users are gathered. The user pair (u_1, u_2) has only $\langle L_1 \rightarrow L_2 \rangle$ as the common mobility pattern. Similarly, (u_1, u_3) has two, (u_1, u_4) has three and (u_1, u_5) has three common mobility patterns. It is observed that users with a higher number of common patterns always have similar (common) interests and are thus closer to each

other. Thus, neither u_2 or u_3 can be identified as the nearest neighbor to u_1. The patterns $\langle L_1 \rightarrow L_2 \rightarrow L_3 \rangle$, $\langle L_4 \rangle$ and $\langle L_5 \rightarrow L_6 \rangle$ are common to u_1, u_4 and u_5. Existing works [9, 10, 11] consider the frequency for each of these common patterns. We argue that the number of days on which check-ins are performed is an important criteria for similarity analysis. We further explore the example to identify the distribution of check-ins by the users. In the given example, the pattern $\langle L_1 \rightarrow L_2 \rightarrow L_3 \rangle$ has been followed by u_1 and u_5 on {Monday, Wednesday} and by u_4 on {Monday, Friday}. The pattern $\langle L_4 \rangle$ has been followed by u_1 and u_5 on {Tuesday, Friday}, and by u_4 on {Tuesday, Wednesday}. Lastly, the pattern $\langle L_5 \rightarrow L_6 \rangle$ has been followed by u_1 on {Thursday, Saturday, Sunday}, u_4 on {Tuesday, Thursday} and by u_5 on {Saturday, Sunday}. Thus, we determine that the pair (u_1, u_4) has common mobility patterns only on three days namely, Monday, Tuesday and Thursday. Whereas, pair (u_1, u_5) has common mobility patterns on all days of the week except on Thursday. Therefore, u_5 can be considered as the nearest neighbor of u_1. Thus, for GPS applications with spatio-temporal data, the count of common mobility patterns and their frequencies are not sufficient to compute the similarity between users.

In this work, we extract stay points (locations) where users have stayed for a period of time. A stay point consists of the latitude/longitude information and the time of visit. Next, each of these physical locations are classified into effective location categories. From the obtained sequence of categories, we perform sequential pattern mining to gather the most frequently occurring check-in patterns for each user. Using the obtained patterns and the every day distribution of check-ins, we formalize three similarity metrics to categorize neighbors into three groups. Common mobility patterns and their frequency of occurrence have been used to compute the similarity score. Our first measure, which is intended to find the weakly similar category of neighbors considers these two parameters, however, it computes the score in a different way. In the second intuition for finding similar neighbors, we add the knowledge of how frequently the users are travelling. Our final intuition for finding the strongly similar category of neighbors deals with the category of locations visited by the users on each day in a week.

The contributions are summarized as follow.

1. A Stay Point Extraction (SPE) algorithm is proposed to select locations where a user stays for a period of time. The proposed SPE overcomes the drawbacks of using threshold by introducing one more term 'significance score'.

2. Existing approaches using the raw trajectory data fail to identify a user's interest in visiting them. Thus, we use nearest neighbor approach to identify the category of a stay point.

3. A seminal sequential pattern mining algorithm [27] is exploited to find the frequent mobility patterns of a user.

4. Three similarity measures have been proposed highlighting their need in different scenarios. The length of common patterns along with their frequency of occurrence are considered in the *Relative Importance-based Similarity* (RIS) metric. We extend this *RIS* measure by estimating the distribution of patterns over the number of days on which check-ins are recorded. We term this as a common *Pattern Distribution-based Similarity* (PDS) metric. Finally, we propose the *Check-in Distribution-based Similarity* (CDS) metric. It considers the distribution of visited locations on the particular days of a week, on which the common patterns are identified. As it adds more granular facts, including the check-in distributions along with the *RIS* measure, hence we treat this *CDS* based measure as the most effective and accurate. Further, we group the neighbors into three categories namely strongly similar, closely similar and weakly similar on the basis of the three proposed similarity metrics. These categories are utilized to rank the K-nearest neighbors.

5. Experiments were performed on a real-world Geolife dataset to evaluate the proposed framework. We compare the proposed metrics with other similarity metrics in literature. Results show that our metrics outperform the existing measures.

The rest of the paper is organized as follows. In Section 2, we survey related works on modeling user behavior and also the different approaches for finding nearest neighbors. Motivation for our current research, along with the problem of finding stay points are discussed in Section 3. The proposed framework for finding nearest neighbors, along with algorithms and their explanations are described in Section 4. In Section 5, the performance of the proposed framework is evaluated using a GPS trajectory dataset. The evaluation approach along with the obtained results and their detailed discussions are also provided in this section. Finally, we conclude our work in Section 6.

2. Related Work

The task of finding nearest neighbors has two parts, modeling the behavior of users and then proposing a metric to compare their pairwise similarity. In this section we explore recent works which model user behavior from visited locations, and subsequently highlight the existing similarity measures along with their limitations.

2.1. Modeling User Behavior

There have been numerous efforts to model the behavior of users from their location histories either recorded from GPS trajectories or shared through social networks. User behavior analysis is performed in three phases [1, 2, 12]. First, the semantics of each visited location is revealed. Second, they predict the path of future movement and finally analyze the user's activities. These works consider the visited locations independently, instead of the patterns within it.

Identifying patterns from existing trajectories can provide a better understanding of mobility behavior. Giannotti *et al.* [13] propose sequential pattern mining algorithms for mining trajectories of moving objects. Frequently occurring sequential patterns have been analyzed in [9, 10, 14, 15]. However, the short and frequent movements of users identify more important behaviors. Existing works in [16, 17, 18] consider routine patterns of movement for analyzing user mobility.

Zheng *et al.* [19] propose a personalized friend and location recommender system for the Geographical Information Systems (GIS). In their HGSM framework, to model the recorded GPS points of user mobility they introduce the concept of stay point. A stay point is any location within a trajectory which represents a region or a geographical area where a user has potentially stayed for a period of time. After identifying these stay points, they are clustered using the standard density based clustering approach. Each such clusters is named as a stay region or stay location. For each user the framework develops a personal hierarchical graph. Each level of this hierarchical graph consists of stay regions. The common stay regions are identified to analyze user mobility. However, it considers each stay region independently, instead of considering them as a sequence.

Lv *et al.* [20] propose a three phase method to model user behavior and then compute similarity. Firstly, the locations in a trajectory are grouped into a confined region. Each of these regions is termed as a stay region. Every stay region consists of more than one check-ins. However, the approach does not consider the distance travelled during stay region formation. Selection of a stay region also depends upon the speed at which a user is moving. Hence, the distance along with time can be used as the constraints for identifying a stay region. Mobility of a user is mostly driven by their interests. Thus, analysis of frequent patterns can lead to effective similarity computation. Moreover, the above mentioned approaches do not consider the semantics of visited locations. We explore the existing similarity computation techniques next.

2.2. Finding Nearest Neighbors

Many approaches have been proposed in literature to measure user similarity [9, 10, 11, 19, 21, 22, 23]. These measures are generally based upon different factors of user behavior. Some of these aspects are the set of common locations visited, the patterns in user movements and frequency with which these patterns occur.

Li *et al.* [22] propose a hierarchical graph-based similarity measurement model, which clusters the visited locations of a user into a hierarchical graph. They measure similarity as the maximum common length of a path traversed sequentially. However, the geographic distance between locations and the semantics of popular locations are not considered. Xiao *et al.* [23] estimate the similarity score between a pair of users from their GPS trajectories. The trajectories are first modeled using the semantics of the visited locations. Subsequently, the similarity score is computed using the common maximal travel sequences obtained from the published trajectories. Zheng *et al.* [19] propose

a new sequence mining algorithm for finding frequent sequential patterns from GPS trajectories. The series of locations visited by a user is represented using a hierarchical graph structure, which is further used in estimating the similarity score between a pair of users. During similarity computation more emphasis is given to the maximum length sequences and common locations travelled by users.

Ying *et al.* [9] propose Maximal Semantic Trajectory Pattern (*MSTP*) similarity which computes similarity between a pair of users on the basis of frequently occurring check-in patterns. It first identifies the maximal patterns from a trajectory. Similarity between these maximal patterns is computed on the basis of the longest common sequence. Next, the participation ratio of the longest common part to the sequential pattern is computed. The participation ratio is computed as the function of the number of sequential patterns having the longest common sequence in it and the total number of sequential patterns obtained. Similarity between two sequential patterns is finally obtained from the average participation ratio of their longest common parts. Similarity between the common patterns is then weighted by either their support count or the TFIDF weight. Here support is the count of the number of times a pattern occurred in each sequence with respect to the total number of sequences. A similarity metric generally produces a maximum score between two identical users. However, the *MSTP* metric fails to produce maximum similarity between two identical users [10].

Chen *et al.* [10] argue that users' similarity computation based on the weighted average of pattern similarity fails to identify the nearest neighbor. While computing the similarity score between a pair of users (u_1, u_2) with respect to u_1, they consider only the most similar maximal pattern out of all the maximal sequence patterns of u_2. They introduce a concept of relative similarity between two users. The relative similarity is computed using the length of the longest common sequences between two users and the support of these patterns to the users. The similarity score between two users u_1 and u_2 is computed in three steps. Firstly, the relative similarity of (u_1,u_2) with respect to u_1 is computed. Second, the relative similarity between (u_1,u_2) with respect to u_2 is computed. Finally, the average of the two relative similarity values are used as the final similarity score. Thus, the proposed similarity metric produces a maximum similarity score between two identical users. However, two users can never be considered as identical if they have similar patterns with fairly distinguishing frequencies. While computing similarity between two users, this measure fails to consider the difference between the support values of the common patterns. Therefore, if two users have a similar mobility pattern with different frequency of visits, still this measure identifies them as identical with maximum similarity score. In the rest of the paper, we term this metric as *MTP* (Maximal Trajectory Pattern).

The problem stated above is addressed by Chen *et al.* in [11]. They propose a Check-in Pattern based Similarity (*CPS*) measure for computing similarity between a pair of users. It first identifies the common patterns between trajectories. Subsequently, it finds the relative importance of the common patterns to

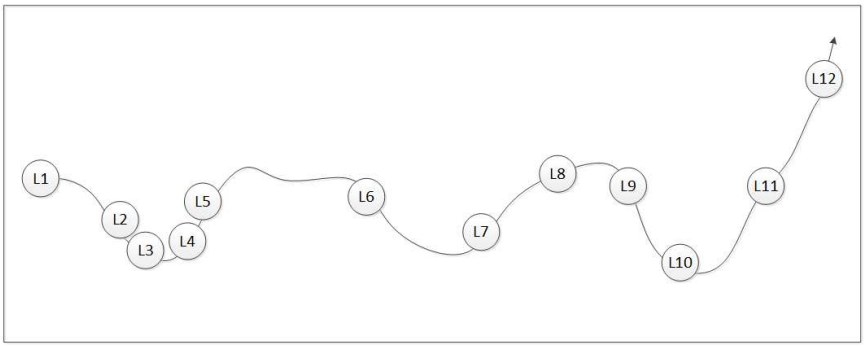

Figure 1: Sample trajectory data of a user u_1. It consists of 12 check-ins at different locations on the same day. The check-ins are depicted with circles, along with the location names.

a user. To estimate the distance between the support values of their common patterns, a standard similarity metric *Bray-Curtis similarity* [24] is used. Both the relative importance and frequency of occurrence of common patterns are used to finally compute the similarity score between two users. The relative similarity computation techniques used in [10] and [11] treat the user similarity score as symmetric. However, similarity score between a pair of users may not be symmetric always. In addition to that, these measures fail to consider the distribution of check-ins recorded on each day on which a common pattern is obtained.

The existing user mobility modeling techniques and metrics for nearest neighbor identification fail in many scenarios. Detailed descriptions of these practical scenarios are depicted in the following section.

3. Motivation and Problem Statement

Trajectories obtained from GPS devices consist of locations where a user visits and stays for a period of time, as well as places through which a user passes by. Moreover, users passing by common or renowned POI often shares them with friends for fun. Identifying the locations where a user has visited and stayed for a specific period of time is therefore an important step for trajectory modeling.

Many researchers in recent years have successfully used the concept of stay point identification from trajectory data. Figure 1 shows a sample trajectory data of a user u_1. The existing stay point identification technique applied on the trajectory data of u_1 is shown in Figure 2. Although the technique shortens a trajectory, it does face some critical issues as well. Below we discuss prominent scenarios that are not addressed by the existing stay point identification technique in [10, 11, 20], subsequently we mention how the proposed approach provides solution to these limitations.

1. Suppose every day a user drives λD distance from his office to reach an indoor stadium before returning home. The distance from the stadium to home is σD. Let us consider the distance threshold selected for defining a

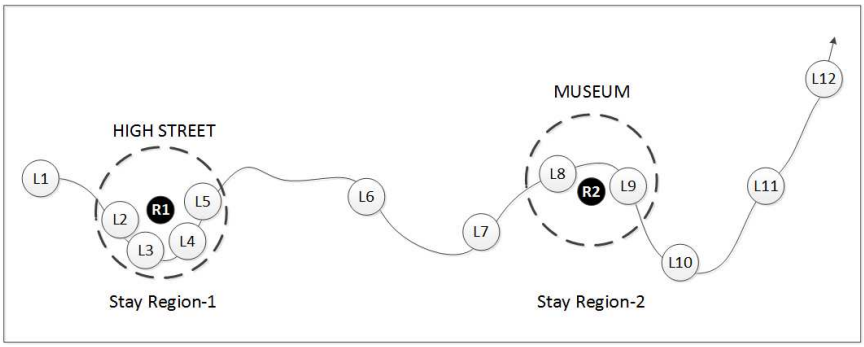

Figure 2: The existing stay point identification technique applied on the trajectory of u_1 (Figure 1) is shown here. It first marks all the stay regions. Here, we represent the stay regions as dotted circles. The two stay regions have $\{L2, L3, L4, L5\}$ and $\{L8, L9\}$ as the set of locations, respectively. Subsequently, the stay point for each stay region is identified by computing the centroid location in it. In this example, the locations $R1$ and $R2$ are the two stay points for stay regions 1 and 2, respectively. We use a dark circle in each stay region to represent the stay point in it. Thus, the trajectory can be reduced to a sequence of stay points visited, i.e. $R1 \rightarrow R2$. Further, the category for each of the identified stay points are found. In this example, the user follows the sequence, *HIGH STREET* $\rightarrow$ *MUSEUM*.

stay region is δD, where $\lambda D > \sigma D > \delta D$. The existing approach creates a stay region if it contains more than one locations in it and the distance from the first location of the stay region to all other locations within it is less than the threshold δD. Therefore, the indoor stadium can never be a member of any stay region according to the existing approach. Thus, the indoor stadium is not considered during the stay point identification process. However, visiting the indoor stadium is a routine activity, and also very important in terms of his/her behavior analysis. In our approach of stay region formation, the significance of a location to a user is taken into consideration. This allows our approach to select a frequently visited location (potential stay point) to be in a stay region even though it fails to meet the thresholds.

2. The stay point in a stay region is computed by taking the mean of its member location's co-ordinates. We argue that the centroid of a region does not identify how important the member location is to a user. In this work, the location with the highest significance score with respect to a user and having the shortest distance from a location category is selected as the stay point in a stay region.

3. The existing stay point identification technique considers only a user's precise location where they actually reside. Incidentally, every check-in does not belong to the commonly known Point of Interests (POI) set. Thus, correlating the categories in POIs of a city to these stay points can lead to some unknown locations. Analyzing a trajectory with such unknown categories does not help in modeling user mobility. In the proposed work,

a score for each location in a stay region is computed for determining the stay point. This reduces the chance of getting unknown locations.

The task of semantic location identification from the stay points is a critical process, as this finally reveals behavior and interest of a user. Researchers have used various techniques to acquire the category or semantic information for each location in a trajectory. Ying *et al.* [9] uses the MIT reality mining mobile phone dataset which has user annotated cell names. These cell names are used as semantic trajectories. Xiao *et al.* [25] creates a database with 6,828,951 POIs of 13 different categories which includes restaurant, museum, etc. Lv *et al.* [20] develops a website using the Google Maps API, where users can manually record their trajectory and also name the category of location for the referred places. Ma *et al.* [26] in their approach for finding user similarity utilize the Nokia Ovi Store for mapping the raw trajectories to semantic locations. However, geographical position of the retrieved category and the point from which the query is made, is not always the same geographical location. This distance has been exploited in this work.

Similarity between a pair of users depends upon many factors, including the mobility patterns, frequency of common behaviors, similar interests, etc. Next, we point out the limitations of the existing similarity measures and subsequently mention how they are addressed in this work.

1. Similarity measure in [9] fails to ensure a maximum similarity score for two identical users. The proposed similarity measures RIS, PDS and CDS ensure a maximum similarity score for two identical users.

2. Similarity measure in [9] does not consider the support of the common sequential patterns. The proposed RIS measure considers the length of all the common frequent patterns along with the support of them. The PDS and CDS measure is weighted by the RIS measure, and hence the similarity score obtained from them is also influenced by the support value of the common frequent patterns.

3. Similarity measures in [10, 11] do not consider the day on which the common pattern is identified. However, it is noteworthy that users' movement during weekends tend to be different compared to a weekday. Therefore, the day on which a frequent pattern is identified is an important criteria for similarity score computation. The proposed CDS measure considers the day on which a common frequent pattern is observed between two concerned users. Users are believed to be closer to each other if they follow similar movement of patterns on a particular day.

4. Existing similarity measures [9, 10, 11] consider similarity between two users as symmetric. Similarity depends upon behavior of an individual and commonality between a pair of users. The proposed similarity measures RIS, PDS and CDS are not symmetric as these two factors are incorporated.

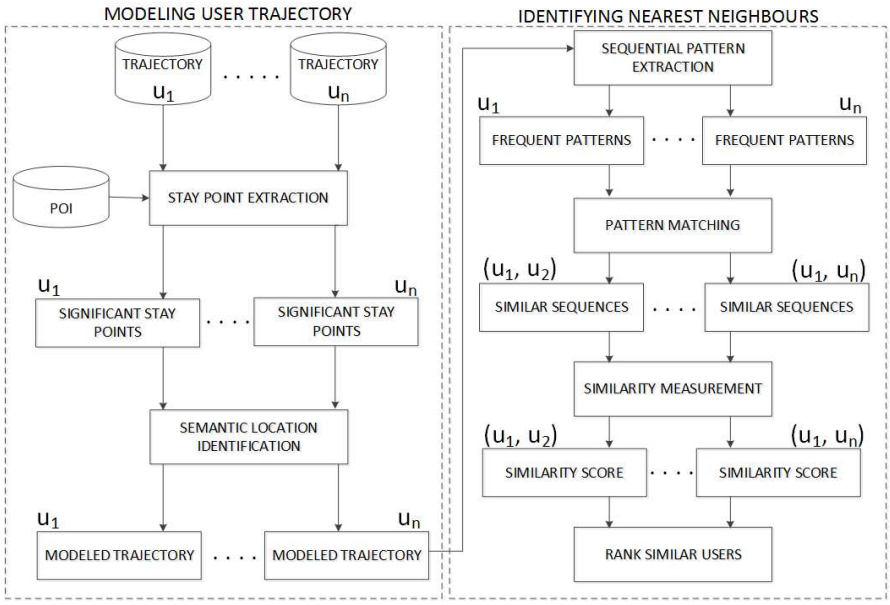

Figure 3: Framework for finding nearest neighbors of an active user u_1.

4. Proposed Framework

The framework for finding nearest neighbors of an active user is depicted in Figure 3. The proposed framework has two major phases, modeling user trajectory and identifying top-K nearest neighbors.

4.1. Modeling User Trajectory

In this phase, we model the raw trajectory data into a sequence of predefined categories. We divide this phase into two modules, stay point extraction and semantic location identification. In the first module, the stay points of each user are identified from its trajectory. Subsequently, the location category of these stay points are identified in the second module.

4.1.1. Stay Point Extraction

The standard approach towards extracting stay points is to first identify the stay regions and then select the stay points from it. For extracting the stay regions one can use the density based clustering approach [19]. However, for behavior analysis generally each cluster is confined by a fixed distance and a time constraint. Therefore, it is necessary to identify how a user moves across the clusters. Using a density based clustering approach it is difficult to find the time when a user moves into a cluster and leaves a cluster. Hence, in this work we propose an algorithm that uses distance and time thresholds to identify

geographic regions where a user stays for a period of time. The thresholds thus help to identify movement of a user across regions.

User mobility is generally driven by either routine activities or interests. Staying at home or at a workplace can be identified as routine activities which a user follows in sequence. In addition to it, the remainder of the places visited are mostly influenced by their interests. Therefore, we consider identifying the locations where a user has actually stayed from the available trajectory. For the sake of readability, we mention a few popularly known terms used to model user mobility.

Definition 1 (*Trajectory*)**.** A Trajectory is a spatio-temporal information of sequential check-ins performed by a user. It can be represented as, $T = \langle T_1, T_2, ..., T_n \rangle$, where each check-in $T_i = (l_i, t_i), 1 \leq i \leq n$ is a doublet having location l and time t details.

The trajectory of a user consists of granular locations where a user checks-in. Behavior analysis considering all these individual locations would be costly and time consuming. Thus, we need to find more generalized locations that can effectively reduce the number of candidate locations, as well keep the underlying interest of users intact. Therefore, the sequential check-ins in a trajectory are grouped into a confined geographical area termed as a stay region. We reproduce the definition of stay region as stated in [25].

Definition 2 (*Stay Region*)**.** A Stay Region (sr) is the geographical cluster where a user stays for a period of δT bounded by a distance of δD. It may consist of a series of locations, $sr = \langle l_i, l_{i+1}, ..., l_j \rangle$ where $distance(l_i, l_z) \leq \delta D$, $\forall i < z \leq j$ and $time(l_i, l_j) \geq \delta T$. Here, $time(l_i, l_j)$ is the time taken by a user to move from location l_i to l_j.

Once the stay regions are identified, we need to find the representative points for each of these regions. By the term representative point, we aim to choose a location from each stay region that has more chance of getting visited whenever the user moves within the stay region. This allows us to identify and work with the significant locations for a user. We thus introduce the term significance score for locations to depict their importance to a user.

Definition 3 (*Significance Score*)**.** The significance score of any location l denotes the importance of the location to a user u_1. It is computed from the frequency of visiting l in a stay region to the fraction of trajectories of u_1 containing the location l. It can be denoted as:

$$Sig(l) = \frac{|l|}{|sr|} * \frac{|T_l|}{|T|} \tag{1}$$

Here, $|l|$ is the number of times u_1 has visited location l ($l \in sr$), and $|sr|$ is the number of locations in the stay region. The total number of trajectories is $|T|$ and the trajectories with location l is represented as $|T_l|$. The significance score of any location l is thus computed from two parameters. The location

frequency is weighted by the ratio of trajectories having the location l in them to the total number of trajectories. Due to this, the effective significance score of the most relevant location to a user is maximised. This significance score is finally used as an important parameter to find the stay points. We approach the problem of finding a stay point in a different way. First, the locations which do not fall within any stay region are not directly discarded. If these individual locations are found to be significant with respect to a user then they are considered as a stay point. Secondly, for identifying a stay point from a stay region having multiple locations we compute a *Score*. The *Score* of each location in a stay region is computed as $\frac{Sig(l)}{D(l)}$. Here, l is a location in a stay region sr and $Sig(l)$ is the significance score of l to an active user. The term $D(l)$ denotes the distance between the location l and the nearest POI category. The location with maximum *Score* is selected as the stay point.

The process of extracting stay points from a trajectory is further illustrated with an example using Figure 4 and Figure 5. Let us consider, trajectory T of an active user u_1 consists of 12 locations ($L1$ to $L12$) along with their time of check-in. Initially, we set $L1$ as a member of the first stay region. We keep on adding locations sequentially to $L1$ till they satisfy the distance and time thresholds (δD and δT). We observe that the distance between $L1$ and $L2$ exceeds δD and the time taken by u_1 to move from $L1$ to $L2$ is also less than the time threshold δT. Thus, $\{L1\}$ forms the first stay region for user u_1. Next, we consider location $L2$ as our second stay region and start adding locations to it till the thresholds are satisfied. The second stay region is formed with locations $\{L2, L3, L4, L5\}$. Similarly, we cluster the rest check-ins into effective stay regions. Stay regions 3 to 8 consists of $\{L6\}$, $\{L7\}$, $\{L8, L9\}$, $\{L10\}$, $\{L11\}$ and $\{L12\}$ set of locations, respectively. This is the first step in identifying stay regions, and the procedure is clearly depicted in Figure 4. Next, we identify the representative points or the stay points for each stay region in T. For selecting stay points we consider significance or importance of a location to a user and its geographic distance from the nearest POI category. The significance score for each location in a stay region is computed using Equation (1). Locations that do not satisfy the significance score threshold are removed from respective stay regions. Therefore, a stay region found from the first step may get discarded if none of the member locations satisfy the acceptable significance score. Figure 5 depicts such a situation where the stay regions 1, 3, 6 and 8 of Figure 4 are not considered for further analysis. Thus, from the trajectory of u_1 only four stay regions are obtained. Therefore, the obtained sequence of movement for the user can be depicted as $L3 \rightarrow L7 \rightarrow L8 \rightarrow L11$. Next, we discuss the technique employed for finding categories of the stay points.

4.1.2. Semantic Location Identification

The stay points extracted from a trajectory are a collection of raw physical locations. Incidentally, finding similarity between users using these raw stay

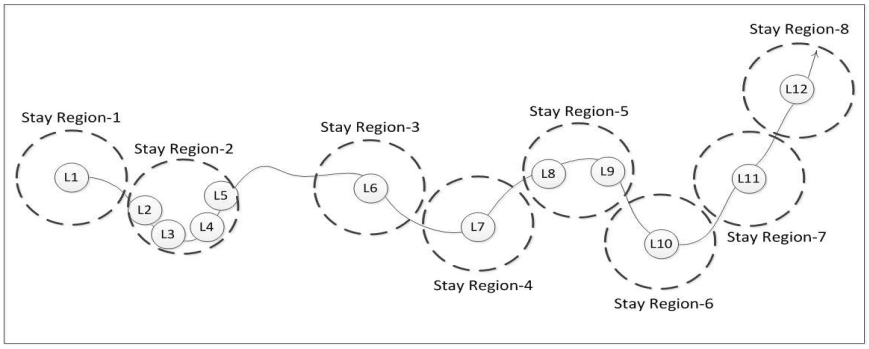

Figure 4: Our proposed stay point identification technique is applied over the trajectory of u_1 (Figure 1). The first step of forming the stay regions are shown in this figure. The stay regions marked with dotted circles are bounded by two thresholds δD and δT for the geographic distance and time of travel, respectively.

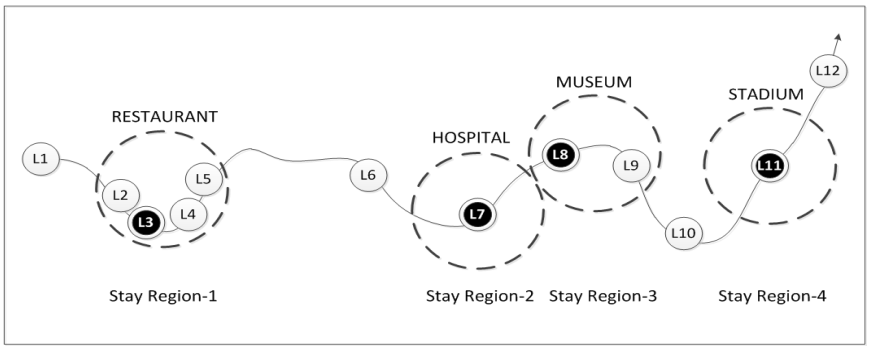

Figure 5: Continuing with the same example in Figure 4, the stay regions which do not have any location which satisfies the acceptable significance score are discarded. For each stay region, the stay points are identified from their significance score and distance from the retrieved location category. The stay points are highlighted with dark circles, one for each stay region.

points may not be sufficient to identify the intention or interest of users. Let us consider two users u_1 and u_2, moving around two different cities, Leicester and London. From their trajectory details, it is found that both of them visited a Book Store followed by a Restaurant. However, the physical location of the stay points namely, Book Store and Restaurant are geographically spread apart by a large distance. Thus, similarity measures which consider only the raw stay points do not identify u_1 and u_2 as neighbors. However, it is evident that even though the physical locations of their check-ins are different, they still share common interests of visiting Book Store and Restaurant. Therefore, in our proposed framework we first find the categories of stay points and then use these categories for similarity computation.

We use the Foursquare API to find categories of the stay points. The API

ALGORITHM 1: Stay Point Extraction (SPE) algorithm

Input: T: Trajectory of a user u
Result: Stay_Points: Series of categories visited by u
Data:
DT, TT, ST is the Distance, Time and Significance Score Threshold;
l_s indicates first location of a temporary Stay_Region;
l_e indicates last location of a temporary Stay_Region

```
l_s = l_e = First location in T;
Start a temporary Stay_Region with l_s;
for each location l in T except the first location do
    if dist(l_s, l) ≤ DT then
        Add l to the temporary Stay_Region;
        l_e = l;
    else
        if time(l_s, l_e) ≥ TT then
            A permanent Stay_Region is formed with all locations from l_s to l_e in series;
        else
            Permanent Stay_Regions are formed for each location l_s to l_e in series;
        end
        l_s = l_e = l;
        Start a new temporary Stay_Region with l_s;
    end
end
for each Stay_Region do
    Compute Sig for each location in current Stay_Region;
    Remove locations from current Stay_Region having Sig ≤ ST;
    if any location exists in current Stay_Region then
        Compute Score of each location in current Stay_Region;
        catg = Category of the location with maximum Score;
        Add catg to Stay_Points;
    else
        Continue with the next Stay_Region;
    end
end
return Stay_Points
```

stores the experiences shared by its users throughout the world starting from the locations visited, ratings and tips given to a POI, etc. From May 2014, the check-ins have been exclusively published from the Swarm platform. It has a predefined set of categories of the locations that can be explored using the two endpoints called Venue Search and Venue Explore. Venue Explore is used to provide details of a location or a defined region. For any physical location, the Venue Explore method in Foursquare API returns a collection of probable categories. The API provides the category based on their proximity to the query location. Our aim is to find a specific category to a given physical location. In this regard, we apply the nearest neighbor approach for finding category.

Details of stay point extraction from a trajectory is depicted in ALGORITHM 1. In the current example (Figure 5), $L3$, $L7$, $L8$ and $L11$ are the representative points for each of the four stay regions (represented with dark circles). Therefore, the obtained sequence of movement for the user can be represented as, $RESTAURANT \rightarrow HOSPITAL \rightarrow MUSEUM \rightarrow STADIUM$. It may be noted here that, this approach selects a visited location and not any arbitrary location like the previous approaches (Figure 2). Our approach towards identifying stay points from trajectories address all the problems mentioned in Section 3. The trajectories of users are thus reduced and converted from raw physical locations into meaningful categories. Further, we analyze these semantic trajectories to identify the nearest neighbors.

4.2. Identifying Nearest Neighbors

In this phase, we use the modeled trajectories of each user to identify their neighbors. We divide this phase into four modules, sequential pattern extraction, matching common patterns, similarity measurement and ranking the neighbors. In the first module, frequently occurring sequential patterns are extracted. The common patterns among them are selected on the basis of certain conditions depicted in the second module. We propose three similarity measures to categorize the neighbors into three groups in the third module. Finally, the last module ranks the top-K nearest neighbors by combining members from each group.

4.2.1. Sequential Pattern Extraction

We apply a sequential pattern mining technique as it can effectively identify the statistically relevant patterns in user movements. First we present a sample user trajectory data and describe how it is converted to a sequence database for input to a sequence mining algorithm. Table 2 contains the trajectory of a user u_1. From the dataset (Table 2), we first find the day on which the check-ins were recorded and mark them as separate weeks along with the days. LocId is the unique name given to each location visited by u_1. Table 3 shows the Week, Day and LocId details from the existing dataset in Table 2. Table 4 shows the sequence database for user u_1 generated from Table 3. This sequence database is presented as input to a sequential pattern mining algorithm for identifying the frequently occurring movement patterns for a user.

We adopt the basic terms in sequence mining like the item, itemset and sequence into our problem as follows.

Definition 4 (*Locationset*)**.** The set of all items in sequential pattern mining is coined as Locationset in this work. It is the set of locations visited by all users. It is represented as $L = \{l_1, l_2, ..., l_n\}$.

Definition 5 (*Event*)**.** The Itemset in sequential pattern mining is coined as Event in this work. An event is a subset of Locationset which contains the set of locations visited by a user on a day in a particular week. An event E is denoted by $(x_1 x_2 \ ... \ x_m)$ where, $x_k \in L, 1 \leq k \leq m$.

Table 2: Trajectory dataset of a user u_1.

Date	Time	Latitude	Longitude
03.05.2016	10:12:45	39.98468	116.3185
07.05.2016	14:25:50	39.98461	116.318
08.05.2016	07:23:15	39.98468	116.3185
10.05.2016	13:37:23	39.98461	116.318
14.05.2016	21:07:10	39.98456	116.3175
17.05.2016	16:25:08	39.98456	116.3175
21.05.2016	19:48:17	39.98452	116.3162
24.05.2016	09:11:39	39.98457	116.3156

Table 3: Processed trajectory dataset of user u_1.

Date	Time	Week	Day	Latitude	Longitude	LocId
03.05.2016	10:12:45	1	Tuesday	39.98468	116.3185	1
07.05.2016	14:25:50	1	Saturday	39.98461	116.318	2
08.05.2016	07:23:15	1	Sunday	39.98468	116.3185	1
10.05.2016	13:37:23	2	Tuesday	39.98461	116.318	2
14.05.2016	21:07:10	2	Saturday	39.98456	116.3175	3
17.05.2016	16:25:08	3	Tuesday	39.98456	116.3175	3
21.05.2016	19:48:17	3	Saturday	39.98452	116.3162	4
24.05.2016	09:11:39	4	Tuesday	39.98457	116.3156	5

Definition 6 (*Sequence*)**.** It is an ordered list of events. It is denoted as $\langle s_1 s_2 \ ... \ s_g \rangle$ where, s_j is an event and $s_j \subseteq L, 1 \leq j \leq g$. A sequence in this work represents the mobility data of a user on a particular day of every week spanning over the entire trajectory.

Table 5 shows the mapping between the terms used in this work and the terms used in sequence mining [27]. A sequence database for a user consists of at most 7 tuples or sequences. Each tuple represents the sequence followed by the user on a particular day over all the weeks spanning over its entire trajectory. For example, a user visits $(L1, L2)$ on Monday of week-1 and then $L3$ on Monday of week-2. Therefore, the user has recorded two events $(L1, L2)$ and $(L3)$. The sequence for Monday is represented as $\langle \ (L1, L2) \ L3 \ \rangle$. Similarly, separate sequences are generated for other days in which check-ins are recorded. Therefore, a sequence database can have at most 7 tuples or sequences.

Definition 7 (*Support*)**.** The Support of a pattern is the number of sequences in the sequence database containing the pattern.

Next we briefly review the available sequence mining algorithms. The GSP [28] algorithm creates a large number of candidate sets and also involves multiple database scans as it is an Apriori based approach. It is therefore very inefficient for mining large datasets. In SPADE [29], although the number of database scans are reduced, the generation of a large number of candidate itemsets makes

Table 4: Sequence database of user u_1. In each sequence we use the LocId instead of the actual latitude and longitude.

Day	Sequence-Id	Sequence
Tuesday	1	$\langle$ 1 2 3 5 $\rangle$
Saturday	2	$\langle$ 2 3 4 $\rangle$
Sunday	3	$\langle$ 1 $\rangle$

Table 5: We represent the mapping of the terms used in sequence mining with the terms used in this work.

Pei *et al.* [27]	Our Work	Representation
Items	Locationset	$L = \{l_1, l_2, ..., l_n\}$
Itemset	Event	$E = (x_1 x_2 \ ... \ x_m)$, $x_k \in L, 1 \leq k \leq m$
Sequence	Sequence	$S = \langle s_1 s_2 \ ... \ s_g \rangle$, s_j is an Event, $s_j \subseteq L, 1 \leq j \leq g$

it inappropriate for large datasets. FreeSpan [30] and PrefixSpan [27] both are based on candidate itemset projection technique. FreeSpan recursively projects the frequent items into smaller datasets. This partitions both the data and the frequent patterns. However, for projecting a pattern in FreeSpan, it needs to be kept in the dataset, and hence it becomes costly. PrefixSpan only deals with the prefix sub-sequences and projects their corresponding postfix sub-sequences. This reduces both execution time as well as the number of generated candidate itemsets. Therefore, in this work we use the standard PrefixSpan algorithm to find the frequently occurring patterns in user mobility.

4.2.2. Matching Common Patterns Between Users

Once the frequently occurring sequential patterns are obtained, the next task is to select common patterns that are effective for similarity computation. Unlike the existing techniques, we consider all the frequent patterns for matching the common patterns. Let a pair of users u_1 and u_2 have two frequent patterns p_1 and p_2 with the same length obtained from their trajectories, respectively. They can be represented as:

$$p_1 = \left\langle x_1 \overset{\Delta t_1}{\rightarrow} x_2 \overset{\Delta t_2}{\rightarrow} ..... \overset{\Delta t_{n-1}}{\rightarrow} x_n \right\rangle$$
$$p_2 = \left\langle y_1 \overset{\Delta \bar{t}_1}{\rightarrow} y_2 \overset{\Delta \bar{t}_2}{\rightarrow} ..... \overset{\Delta \bar{t}_{n-1}}{\rightarrow} y_n \right\rangle$$

A pair of patterns can only be considered common if, $x_k = y_k$ where, $1 \leq k \leq n$, i.e. the locations at each position of the two patterns are the same, and $|\Delta t_k - \Delta \bar{t}_k| \leq t_{th},\ 1 \leq k \leq n$ where, t_{th} is the pattern duration threshold. The common patterns are further used to compute similarity score between a pair of users.

4.2.3. Similarity Measurement

As mentioned in Section 2.2, the similarity computation depends upon the visited locations, patterns in movement and their frequency of occurrence. Along with them, there are certain factors such as the number of days having check-in information and the locations checked-in on a day can also be exploited to compute similarity score between a pair of users. Based on those factors we propose three similarity measures for finding neighbors of a user in various GPS applications and Location Based Social Networks (LBSN).

I. **Relative Importance Based Similarity Measure (*RIS*)**

A pattern having high support can always be treated as the most travelled path for a user. So a similarity calculation on the basis of quantified frequency of common patterns play an important role. A pattern depicts a user's routine movements, and its support assesses their frequencies. Hence, we use both these elements to represent the mobility of a user. If P is the set of all frequent patterns, then the mobility of any user u_1 can be written as $M_{u_1} = \{(p_1, \sup_{u_1}(p_1)), (p_2, \sup_{u_1}(p_2)), ..., (p_z, \sup_{u_1}(p_z))\}$, $\sup_{u_1}(p_k) \geq \alpha$ and $p_k \in P$ where, $1 \leq k \leq z$, α=support threshold. It can be noted that, each of the locations mentioned in P indicates its category, which refers to the interest of users and not the physical locations. The importance of a pattern varies from user to user. Therefore, a common pattern may not always be equally important to both pairs of users. In this regard, we introduce the term *relative*, which first determines the importance of a pattern to a user, and then computes similarity on the basis of that.

Definition 8 (*Relative Importance*)**.** The relative importance of the set of common patterns (CP) between two users u_1 and u_2 with respect to mobility of u_1 is computed as the factor of the number of locations covered by patterns in CP to the total number of patterns obtained, both weighted by their support values in the mobility of u_1.

The Relative Importance (τ) of common patterns CP with respect to the mobility of u_1 can thus be written as,

$$\tau_{u_2 u_1} = \frac{\sum_{p \in CP} length(p) * \sup_{u_1}(p)}{\sum_{p \in M_{u_1}} length(p) * \sup_{u_1}(p)} \tag{2}$$

Users who follow similar mobility patterns over more number of days are believed to be close neighbors. Incidentally, *RIS* considers only the common patterns and their frequency between a pair of users, and hence, fails to consider this aspect. Therefore, the neighbors identified from *RIS* metric are termed weakly similar. To address this problem, we extend the

RIS metric and hence propose a new similarity metric which considers the distribution of common patterns throughout a week.

II. **Common Patterns Distribution Based Similarity Measure (*PDS*)**

Two users which have common patterns covering maximum number of visited locations with higher frequencies can intuitively be considered as neighbors. As people tend to move between places frequently, the obtained patterns identify the noted routines of their movements. In reality, the number of days recognized for users' mobility is scattered. In this regard, we categorize two types of users on the basis of their mobility, frequent travellers and occasional travellers. For frequent travellers, we obtain a considerably higher number of check-ins distributed over number of days compared to an occasional traveller. We provide an example in Table 6 for better understanding of the scenario. Here each column represents the number of check-ins performed by a user on each day in a week.

It is observed that Bob has a fewer number of check-ins and mostly they are concentrated in the weekends. Whereas, Eve is a very frequent traveller with check-ins on all days of a week. Now if all the users have Gym as their common place of interest, then it is obvious that Alice should be a neighbor to Bob. As their distribution of check-ins over the number of days suggests that they most likely visit the Gym on weekends. Hence, a similarity metric should select Alice as a neighbor to Bob over all other users. Hereby, we define and represent the metric for user similarity computation on the basis of pattern distribution over the number of days in which check-ins are observed.

Definition 9 (*Pattern Distribution*)**.** Common Patterns Distribution based Similarity (γ) between users u_1 and u_2 is the relative importance weighted by the ratio of the difference between number of days checked-in by the users and the maximum number of days checked-in by them. The similarity score between the pair on basis of their distribution of common patterns with respect to the mobility of u_1 can thus be computed as,

$$\gamma_{u_2u_1} = \tau_{u_2u_1} * (1 - \frac{|days_{u_1} - days_{u_2}|}{max(days_{u_1}, days_{u_2})}) \tag{3}$$

Here, $\tau_{u_2u_1}$ is the relative importance of common patterns between u_1 and u_2. The number of days in which check-ins of u_1 and u_2 are observed is represented as $days_{u_1}$ and $days_{u_2}$.

The similarity score computation using the *PDS* metric covers many behavioral aspects including the common patterns followed, their frequencies and also the number of days in which they are distributed. However, we believe that maximum closeness between a pair of users can be obtained if they visit more common locations every day. This requires an estimate of

Table 6: Example for describing the different scenarios which the proposed metric considers.

User	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Bob	0	0	0	0	0	12	15
Eve	5	2	9	1	12	4	3
Alice	0	0	0	0	0	2	1
Maria	2	0	1	0	0	14	13

the check-ins performed on each day in a week. The *PDS* metric fails to consider this aspect of user mobility in similarity computation. Therefore, we term the neighbors obtained from *PDS* metric as closely similar. To address this problem we extend the *PDS* metric and hence propose the final similarity metric to identify the nearest neighbors.

III. **Check-in Distribution Based Similarity Measure (*CDS*)**

Continuing with the same example in Table 6, the number of days in which check-ins are available makes Bob and Alice neighbors. However, the weekend check-ins of Bob are much denser than those of Alice. Thus, along with the common locations, Bob also visits places that are not visited by Alice. So selecting Alice as the nearest neighbor to Bob may be inappropriate. Incidentally, if the common patterns between two users cover the maximum length of their mobility sequence for a given day then it is highly likely that they did not visit diverse locations on the same day. Thus, from the given example Maria can be considered as the nearest neighbor to Bob. Next we define the term check-in distribution along with the similarity measure.

Definition 10 (*Check-in Distribution*)**.** The Check-in Distribution based Similarity (ϕ) between users u_1 and u_2 is computed as the weighted average of the relative importance with the set difference of the number of common locations visited on the same day on which the check-ins are performed.

$$\phi_{u_2u_1} = \tau_{u_2u_1} * (1 - \frac{\sum\limits_{d \in D} |L_{u_{1_d}} \backslash L_{u_{2_d}}|}{\sum\limits_{d \in D} |L_{u_{1_d}}|}) \tag{4}$$

Here, D is the set of days in which check-ins are recorded and $L_{u_{1_d}}$ is the set of locations visited by user u_1 on day d.

The *CDS* metric considers all the mobility aspects required for similarity computation. Therefore, the neighbors identified using *CDS* are termed strongly similar. Thus, the three proposed similarity metrics categorizes the neighbors into strongly, closely and weakly similar. Next, we depict a ranking technique for ranking the predicted neighbors.

Table 7: The neighbors of a user u_1 identified using the proposed measures are shown here. The neighbors are arranged in descending order of their similarity score with respect to u_1.

Similarity Level	Similarity Measure	Selected Users	Category
3	*CDS*	u_2, u_3	Strongly Similar
2	*PDS*	u_4, u_5, u_6, u_7	Closely Similar
1	*RIS*	$u_8, u_9, u_{10}, u_{11}, u_{12}, u_{13}, u_{14}, u_{15}$	Weakly Similar

4.2.4. Ranking Nearest Neighbors

As mentioned earlier the *CDS*-metric is the strongest similarity metric among the three proposed metrics. This is due to the fact that it considers check-ins on individual days and not over a week as a whole. Therefore, the similarity score between a pair of users in *CDS*-metric is generally less than others. This leads to a situation where a GPS application using this *CDS*-metric may fail to identify the complete set of K-nearest neighbors. However, using other proposed metrics like *PDS* and *RIS*, the K-nearest neighbors can be identified. We also categorize these users satisfying the similarity measures into three distinct groups. Therefore, any GPS application which uses our similarity measures can effectively select K-nearest neighbors, where the value of K may vary as per requirement. Next we explain such a scenario with one specific example, as depicted in Table 7.

Suppose a GPS application requires the ten (10) nearest neighbors of a user u_1. As *CDS* is the strongest measure with many factors considered in it, we try to select all neighbors obtained from this metric. However, in this example it is impossible to identify ten neighbors satisfying *CDS*. In such a scenario, we employ a ranking technique. Subsequently, we start considering users with the highest level of similarity until the required number of nearest neighbors for u_1 are identified. Therefore, the application identifies a number of nearest neighbors using CDS. The neighbors with more than median similarity scores are selected from any similarity level. Thus, users $\{u_2, u_3\}$ from CDS, $\{u_4, u_5, u_6, u_7\}$ from PDS and $\{u_8, u_9, u_{10}, u_{11}\}$ from RIS are selected to complete the set of K-nearest neighbors (K=10). This explains how the ranking technique can be employed by a GPS application for identifying K-nearest neighbors.

5. Experiments

We conducted a series of experiments over a real-world dataset Geolife [31, 32, 33]. All the experiments are implemented using Matlab. Here, the proposed similarity measures are compared with the three existing works in literature, namely *MSTP* [9], *MTP* [10] and *CPS* [11]. In addition to the Geolife dataset, we also use a synthetic trajectory data for comparison.

5.1. Dataset Description

Geolife is a GPS Trajectory dataset collected by Microsoft Research Asia for their Geolife project. This dataset records a broad range of users' outdoor movements such as shopping, sightseeing, dining, hiking, cycling, etc.

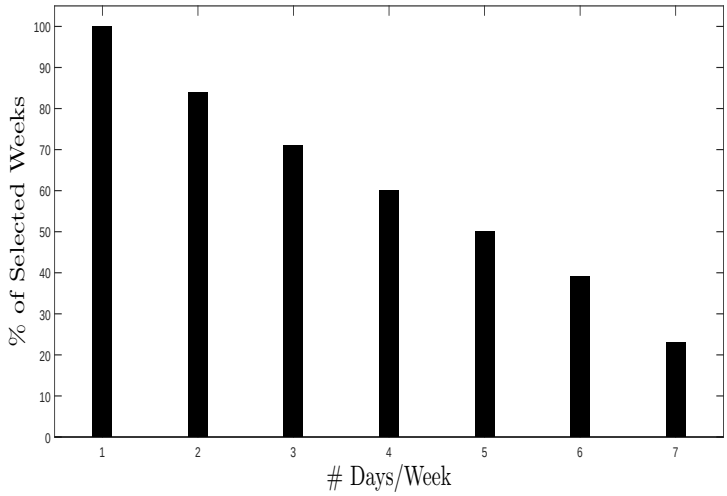

Figure 6: The % of selected weeks are plotted by varying the minimum number of days per week in which check-ins were recorded. The horizontal axis shows the minimum number of days considered starting from 1-day to a maximum of 7-days in a week. The vertical axis shows the % of weeks selected from the dataset satisfying the condition of a minimum number of days in the horizontal axis.

Our proposed framework finds the similarity between users on the basis of the check-ins performed every day. We select weeks for a user with a significant number of days in which check-ins are reported. In this regard, we analyze the percentage of selected weeks, by varying the number of days in a week on which check-ins were recorded (Figure 6). We observe that the percentage of selected weeks decreases as we consider a higher number of days per week. For experiments, we selected a week only if it has at least four days of recorded check-in. However, with this constraint we found that the standard deviation of the number of weeks per user is 22.62. This indicates that the distribution of selected weeks for each user is skewed. Next, we aim to reduce the standard deviation of the number of weeks per user. In this regard, we compute the median of this distribution of weeks per user. Those users who have visited (checked-in) a number of weeks more than the obtained median are selected for analysis. Figure 7 shows the median value of the number of weeks per user, as we vary the minimum number of days in a week. Along with it, we also plot the selected percentage of users having number of weeks more than the corresponding median value. We observed that 45% of the total users having more than 4 weeks of check-in data are selected when a week with minimum four days of check-in records are considered. These users are considered as active and experiments are performed on them. A detailed description of the working dataset is shown in Table 8.

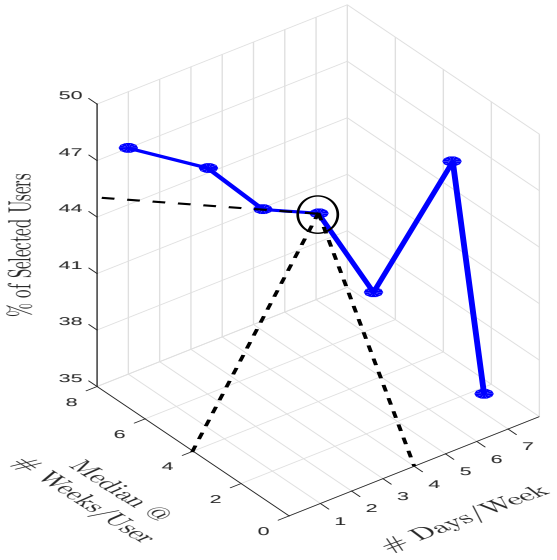

Figure 7: The 3D-plot shows the % of users selected by varying the minimum number of days per week in which check-ins were recorded. First, we compute the median value of the number of weeks per user. Subsequently, the percentage of users having number of weeks more than the obtained median value are plotted in the graph. We work with 45% of all users having more than 4 weeks of check-in data, where each week has atleast 4 days of check-ins.

The synthetic dataset used for comparing the similarity measures are shown in Table 9. Each tuple in the dataset corresponds to the sequences of a user. For example, user u_2 has three sequences $\langle l_1\ l_2\ l_4\ l_8\rangle$, $\langle l_2\ l_4\ l_5\rangle$ and $\langle l_1\rangle$ on Tuesday, Saturday and Sunday, respectively. This dataset is used to find the nearest neighbor of u_1 using the existing CPS measure and our proposed measures (RIS, PDS, CDS).

5.2. Parameter Selection

To evaluate the proposed framework we need to set certain parameters. In this section, we specify each of those parameters used in the algorithms.

5.2.1. Stay Point Extraction

The proposed SPE-algorithm requires three user defined threshold values for distance, time and significance score. In performing experiments on the Geolife dataset we set the distance threshold (δD) as 200 meters, the time threshold (δT) as 30 minutes and the significance score threshold (ST) as 0.4. To convert the raw stay point data into semantic locations, we use the Foursquare API. The response from Foursquare API is further explored to extract the nearest category along with its distance from the requested physical location.

Table 8: Detail description of the working dataset.

Active Users	Trajectories	Check-ins	Weeks
81	17182	17824263	1440
Effective Days	Distance(km)	Duration(hr)	Trajectory/User
8551	1196898	46459	212
Check-ins/User	Check-ins/Week	Effective Days/User	Effective Days/Week
220053	12378	106	6
Distance(km)/User	Distance(km)/Week	Duration(hr)/User	Duration(hr)/Week
14777	831	1013	57

Table 9: A synthetic dataset with check-in details of five users is shown here. Each sequence corresponds to the day on which a check-in activity has been recorded. This dataset is used to identify the nearest neighbor of u_1.

User	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
u_1	$\langle l_1\ l_2\ l_3\rangle$	$\langle l_1\rangle$	$\langle l_2\ l_3\rangle$	$\langle l_1\ l_2\ l_3\ l_4\ l_5\rangle$	$\langle l_2\ l_3\ l_4\ l_5\rangle$		$\langle l_4\rangle$
u_2		$\langle l_1\ l_2\ l_4\ l_8\rangle$				$\langle l_2\ l_4\ l_5\rangle$	$\langle l_1\rangle$
u_3		$\langle l_1\ l_4\ l_9\rangle$			$\langle l_1\rangle$	$\langle l_2\ l_3\rangle$	$\langle l_2\ l_3\ l_9\rangle$
u_4	$\langle l_8\rangle$	$\langle l_1\ l_2\ l_3\ l_8\ l_9\rangle$	$\langle l_1\ l_2\ l_3\ l_8\rangle$		$\langle l_9\rangle$	$\langle l_1\ l_8\rangle$	$\langle l_2\ l_3\rangle$
u_5	$\langle l_2\ l_3\ l_9\rangle$	$\langle l_6\ l_9\rangle$	$\langle l_6\rangle$	$\langle l_2\ l_3\ l_6\ l_7\ l_9\rangle$	$\langle l_2\ l_3\ l_6\rangle$		

5.2.2. Sequential Pattern Extraction

In order to extract the sequential patterns from modeled trajectories of stay points, we use the standard PrefixSpan algorithm. This sequence mining approach requires a support value that is computed for each candidate frequent locationset. We set the support threshold for PrefixSpan algorithm as 0.3.

5.2.3. Matching Common Patterns Between Users

The similarity between the recovered sequential patterns of two users depend upon a pattern duration threshold. As already mentioned in Section 4.2.2, two patterns are identified as common if they satisfy certain conditions. The time taken by people to move around places depends upon the purpose of the visit. For example, if a user visits a work place we may expect their next movement to be recorded after eight hours. Similarly, the same person is expected to move from a coffee shop within 30 minutes, if it is not their workplace. So we understand that a single parameter is not sufficient to capture the time of stay for a user at a POI. Hence, to find the optimal threshold for pattern duration (t_{th}), we experimented with various time intervals over the maximal patterns between users (Figure 8). We observed that a 15 min threshold selects a maximum 30% of the common maximal patterns. As we increase this time span, the percentage of acceptance for patterns gradually increases. During experiments, the time threshold was set to 180 min. Thus, if the time difference between any two consecutive sequential locations (l_1, l_2) of two patterns (p_1, p_2), respectively is more than 180 min, then the patterns are not considered as common.

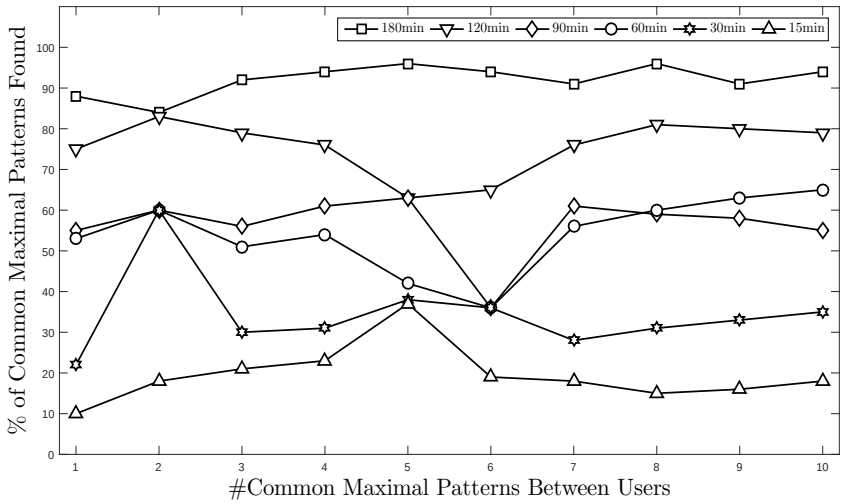

Figure 8: The horizontal axis shows the number of common maximal patterns between a pair of users. Whereas, the vertical axis shows the percentage of common patterns found using the condition mentioned in Section 4.2.2. We vary the pattern duration threshold t_{th} over the obtained number of common patterns. This graph depicts that a pattern duration threshold of around 180 mins selects over 80% of the common maximal patterns.

5.3. Results and Analysis

In this section, we provide the results and their detailed analysis after extensive experimentation on the Geolife dataset. The proposed similarity metrics are compared with the existing metrics in literature. Subsequently, we also explore the impact of these similarity metrics on the location prediction problem.

5.3.1. Performance of Similarity Metrics

We evaluated our proposed framework as depicted in Figure 9. Here, we select each user and compute its similarity score with all other users. For performance evaluation of the similarity measures in nearest neighbor prediction, the ground truths about actual neighbors are required. To achieve this, we divide the trajectory of a user into two equal parts. We consider each of these two parts as trajectories of two different users. Suppose, user u_1 has x number of selected weeks having more than four days of recorded check-ins. So we generate two users ${u_1}^*$ and ${u_1}^\#$, where we assign all the trajectories in week numbers $\{1, 2, ..., x/2\}$ to ${u_1}^*$ and in a similar way all trajectories in week numbers $\{(x/2) + 1, (x/2) + 2, ..., x\}$ to ${u_1}^\#$. It should be noted here that, since users ${u_1}^*$ and ${u_1}^\#$ are actually generated from the trajectories of u_1, hence they are the nearest neighbors to each other. This procedure helped us to get the ground truths (nearest neighbor for each user). To evaluate the proposed approach with the existing approaches, we predict a set of neighbors, and subsequently two evaluation metrics are introduced, namely *Perfect Prediction* and

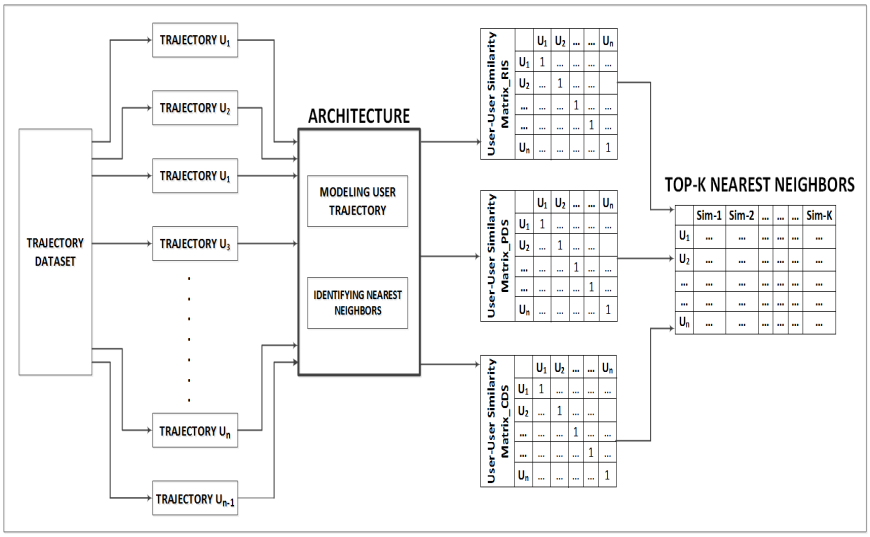

Figure 9: The framework for evaluating our proposed approach.

Table 10: Percentage of *Perfect Prediction* obtained using proposed similarity metrics.

Performance (%)	Existing			Proposed		
	MSTP	*MTP*	*CPS*	*RIS*	*PDS*	*CDS*
Perfect Prediction	58.352	60.361	61.254	62.248	64.894	**67.257**

Successful Prediction. The *Perfect Prediction* is the measure of the number of times a similarity metric correctly identified the nearest neighbor of an active user at the 1^{st} position in the predicted set. Whereas, *Successful Prediction* is the measure of the number of times the nearest neighbor is found in the predicted set of neighbors (length of the set > 1) identified by a similarity metric. It can be represented as:

$$Successful\ Prediction = \begin{cases} 1; Actual\ nearest\ neighbor \in Set\ of\ predicted\ neighbors \\ 0; Actual\ nearest\ neighbor \notin Set\ of\ predicted\ neighbors \end{cases} \tag{5}$$

Table 10 shows the percentage of *Perfect Prediction* obtained using *MSTP*, *MTP*, *CPS*, *RIS*, *PDS* and *CDS* similarity metrics. It is observed that our proposed similarity measures have higher *Perfect Prediction* rate than the existing metrics in literature. Further, the *CDS* metric has the best percentage of *Perfect Prediction*. We analyze the results of *Successful Prediction* when a set of neighbors are identified. Figure 10 shows the observed results, where the horizontal axis is the length of the predicted set of neighbors. Here, we vary the number of neighbors predicted from 2 to 10. More number of *Successful Predictions* are found as we increase the length of the prediction set (number of

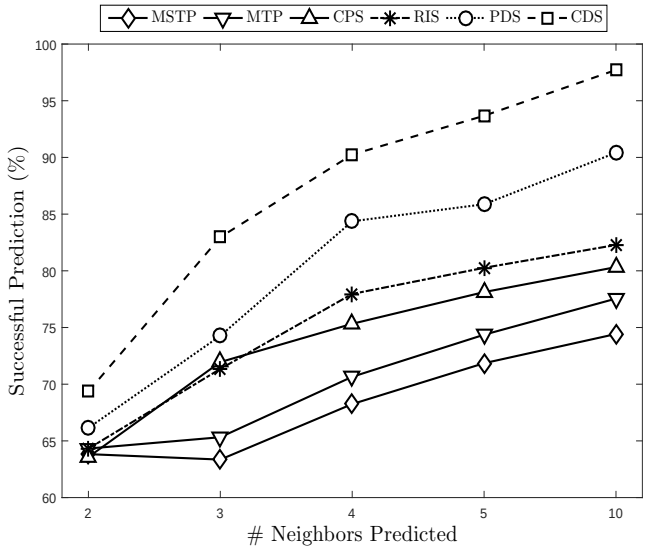

Figure 10: *Successful Prediction* vs Number of neighbors predicted using various similarity metrics on Geolife dataset.

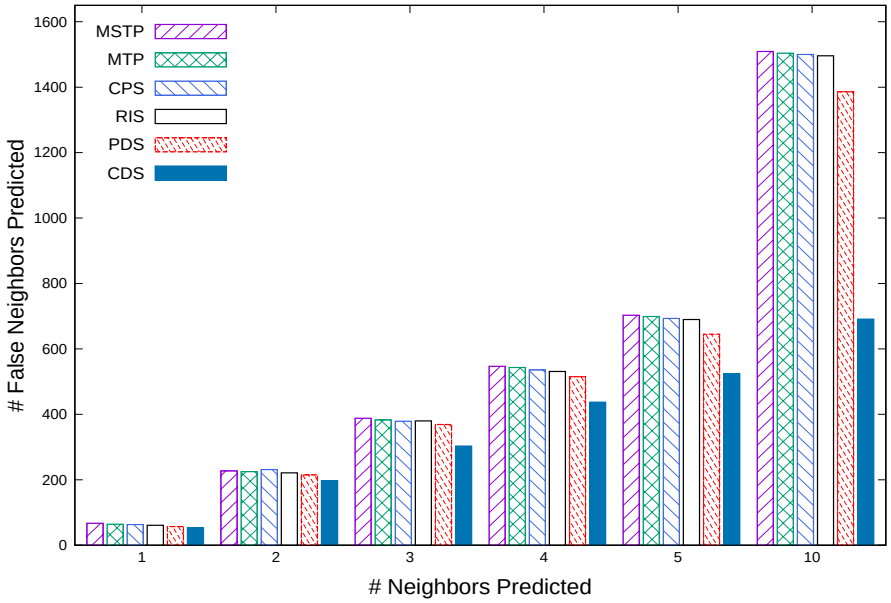

Figure 11: Number of False Neighbors Predicted vs Length of the Predicted list for each similarity metric on Geolife dataset.

predicted neighbors). *CDS* metric shows more consistency and a better success rate in predicting neighbors than others. We also computed the number of false neighbors predicted by the similarity approaches (Figure 11). Strict measures like PDS and CDS often fail to determine the required K neighbors. In those cases, the number of false predictions are significantly lowered. The PDS and CDS measure is observed to have a higher rate of successful prediction than the others.

The neighbors thus obtained from the similarity metrics are further ranked to evaluate the proposed similarity metrics. In this regard, we describe a metric termed as *Positional Rank*. If two similarity metrics S_1 and S_2 ranks the ground truth at position 3 and 7, respectively in the predicted set, then S_1 is favoured as the better performing similarity metric over S_2. The *Positional Rank* indicates how the rank of the predicted neighbors are close to the ground truths. The *Positional Rank* (PR) is computed as:

$$PR = \frac{1}{|U|}\sum_{i\in U}\frac{1}{|G|}\sum_{j\in G}\begin{bmatrix} OR^i_j/_{PR^i_j}; OR^i_j \leq PR^i_j \; and \; \{j \in G\} \cap PL_i \neq \emptyset \\ PR^i_j/_{OR^i_j}; PR^i_j < OR^i_j \; and \; \{j \in G\} \cap PL_i \neq \emptyset \\ 0 \quad ; \{j \in G\} \cap PL_i = \emptyset \end{bmatrix} \tag{6}$$

where,
U = set of all the active users,
G = set of all neighbors of a user,
PL_i = predicted set of neighbors,
OR^i_j = original rank of the neighbor j with respect to user i,
PR^i_j = predicted rank of the neighbor j in PL_i.

It can be noted here that, as per our experimentation strategy the ground truth consists of only the nearest neighbor for each user. Therefore, it is desired that the ground truth should appear in first position in the predicted list. Figure 12 shows the average *Positional Rank* of the nearest neighbor in the set of neighbors predicted for each user. During experimentation, we found that for many instances the CDS measure alone was unable to identify the required number of neighbors to be predicted. In those scenarios, we combine the neighbors obtained from PDS and if required from RIS to complete the set of required neighbors. Combining the neighbors obtained from PDS and RIS with the neighbors of CDS enhances the chance of finding the required number of nearest neighbors. Therefore, the combined approach yields a better result than the individual measures. For the combined approach, if a set of 10 neighbors are predicted for each user, then the average *Positional Rank* of their nearest neighbor in the predicted set is found to be nearly 0.94.

Next we describe the performance of various similarity measures over the synthetic dataset provided in Table 9. First, we perform the PrefixSpan sequential

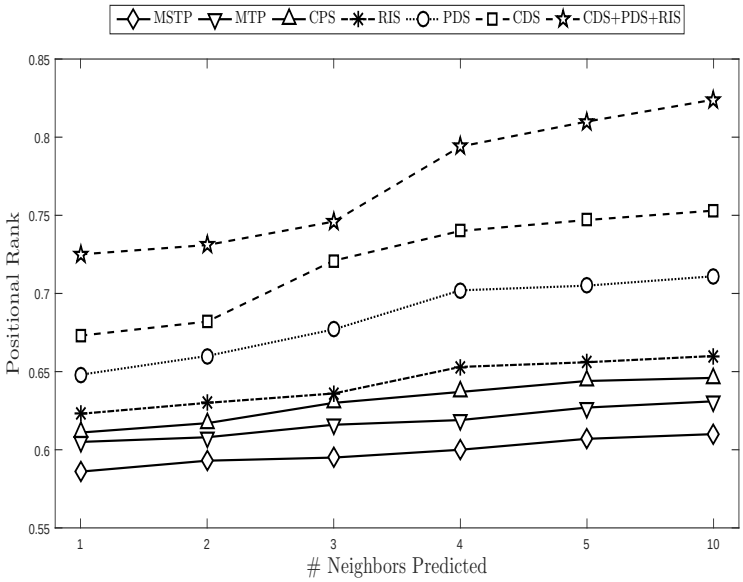

Figure 12: *Positional Rank* vs Number of neighbors predicted using various similarity metrics on Geolife dataset.

pattern mining algorithm over the travelled sequences of every user to extract their frequent patterns of movement. Subsequently, the common patterns between the concerned pair of users are selected. Finally, these frequent patterns and the common patterns in them are used to compute the similarity score between two users. Table 11 shows the results obtained after computing the similarity score between user u_1 and others. The user with highest similarity score for each measure is highlighted in bold. Working with this synthetic dataset helps us elaborate the advantages of our proposed measure over existing works. The CPS [11] similarity measure depends upon the frequency of pattern occurrence and the difference between the support count. Results in Table 11 show that the CPS measure selects u_3 as the nearest neighbor to u_1. Incidentally, our first metric RIS also considers the frequency of pattern occurrence. Thus, it is observed that our RIS measure also selects u_3 as a neighbor along with u_4, with the same similarity score. The user u_1 has six days of check-in, which marks him as a frequent traveller. Whereas, users u_2 and u_3 do not have dense check-ins, which suggests that they are not frequent travellers. The results from all the measures rightly discards u_2 as the probable neighbor. However, CPS and RIS identifies u_3 as the nearest neighbor to u_1, which is found to be inaccurate on the basis of their sequence of movement. The RIS measure does not consider the number of days in which the check-ins have been observed. From the sequence database mentioned in Table 9, it can be observed that both u_1 and u_4 are frequent travellers. Therefore, the metric PDS

Table 11: The similarity scores between u_1 and other users using the existing CPS-based metric and our proposed similarity metrics are shown here. Similarity scores highlighted in bold indicate the nearest neighbor found using the corresponding similarity measure.

	$sim(u_1, u_2)$	$sim(u_1, u_3)$	$sim(u_1, u_4)$	$sim(u_1, u_5)$
CPS	0.464	**0.74**	0.625	0.384
RIS	0.447	**0.868**	**0.868**	0.737
PDS	0.224	0.579	**0.868**	0.614
CDS	0.028	0.054	0.163	**0.276**

is found to perform better than both the existing CPS and the proposed RIS. However, from Table 9 we also observe that user u_1 prefers movement during weekdays in contrast to u_4 who has dense check-in data during the weekends. A similarity measure should consider the set of locations visited by two users on a given day. This aspect is considered by our proposed CDS measure. The proposed CDS measure punishes the similarity score if two users have a higher number of dissimilar locations visited on a common day. For example, during similarity score computation between u_1 and u_4 using CDS, their sequences on Monday, Thursday, Friday, Saturday and Sunday effect the similarity score to a fairly large amount. Incidentally, sequences of u_1 and u_5 are found to be almost equal and also on nearly similar set of days. Sequence of movement on Monday, Thursday and Friday by users u_1 and u_5 contributes to the CDS similarity score. Therefore, CDS rightly identifies u_5 as the nearest neighbor to u_1. It can be noted here that, the similarity score between (u_1, u_5) using CDS is less than the similarity score between (u_1, u_3) using CPS. However, considering the CDS similarity scores between u_1 and other users we find that u_5 is rightly identified as the nearest neighbor. Though the similarity score using CPS is higher, it identifies u_3 as the nearest neighbor to u_1, which is not true from the existing trajectory data.

5.3.2. Impact of Similarity Measures

Identifying the nearest neighbors help to solve many real world problems. Predicting the next location of visit for an active user is one such problem which is well studied in literature. We select this location prediction problem for evaluating the effectiveness of the proposed similarity metrics in real world scenarios. The location prediction problem can be stated as follows, for an active user u_1 having historical check-ins of $\langle (l_1, t_1), (l_2, t_2), ..., (l_w, t_w) \rangle$, we need to predict the location at which it may check-in next at time t_{w+1}. The standard User-based Collaborative Filtering (UCF) technique [34, 35] is one of the traditional method in predicting user ratings for an item in a recommender system. This technique has also been successfully used in location prediction problem by Huo *et al.* in [36]. It involves finding similarity scores between a pair of active users. In this regard, we utilize the proposed metrics and the existing *MSTP*, *MTP* and *CPS* metrics in *UCF* technique for location prediction. For performance evaluation, we compute the *Mean Average Precision (MAP)* and

Successful Prediction of the predicted locations using all the similarity metrics. Moreover, the absolute errors in location prediction is further compared using the Mean Absolute Error (MAE). The above mentioned metrics are mostly used to evaluate the performance of a prediction model. As this work mainly focuses on the aspect of finding K-nearest neighbors for a given user, we frame the above scenarios in a different way. We deliberately hide 95%, 90%, 85%, 75%, 50%, and 25% random locations from the published trajectory of each user. In this way, we generate six different training sets having 5%, 10%, 15%, 25%, 50% and 75% of published check-ins. Our goal is to predict the deliberately hidden locations using the UCF technique. For performance evaluation these hidden locations are considered as the ground truth. Next, we adopt the basic terms like *UCF*, *MAP* and *MAE* into our problem as follows.

(a) **User-based Collaborative Filtering (*UCF*)**: It states that if a set of selected neighbors (su) have followed the same sequence of movement as user u_1 for first c check-ins, then the next sequence of movement $(c+1)$ for u_1 may be analyzed using check-ins of su. As depicted in [36], if $U = \{u_1, u_2, u_3, \ldots, u_n\}$ is the set of users and $L = \{l_1, l_2, l_3, \ldots, l_q\}$ is the set of locations checked-in by them, then the probability of a user u_i to visit a location l_j is given by,

$$UCF_{u_i,l_j} = \frac{\sum_{z \in U} (SI_{u_z u_i} * T_{u_z,l_j})}{\sum_{z \in U} SI_{u_z u_i}} \tag{7}$$

where,
$SI_{u_z u_i}$ = similarity measure between users u_i and u_z, and
$T_{u_z,l_j} = \begin{cases} 1; \text{if } \mathrm{u_z} \text{ has checked-in at } \mathrm{l_j} \\ 0; \text{if } \mathrm{u_z} \text{ has not checked-in at } \mathrm{l_j} \end{cases}$

(b) **Mean Average Precision (MAP)**: The Mean Average Precision (MAP) is a popular single score performance measure used for evaluating the quality of a prediction model. It is the mean of the average precision of the predicted locations for each active user. The *Precision* or Positive Predictive Value is computed as the fraction of the retrieved locations that are relevant.

$$Precision = \frac{\#\, Recovered\ ground\ truths}{\#\, Predictions} \tag{8}$$

(c) **Mean Absolute Error (*MAE*)**: A statistical accuracy metric widely used to measure quality of a recommender system [37, 38, 39, 40]. It is measured by finding the average deviation of the predicted rating with the actual user rating, in a user-item rating dataset. In our case the actual and predicted

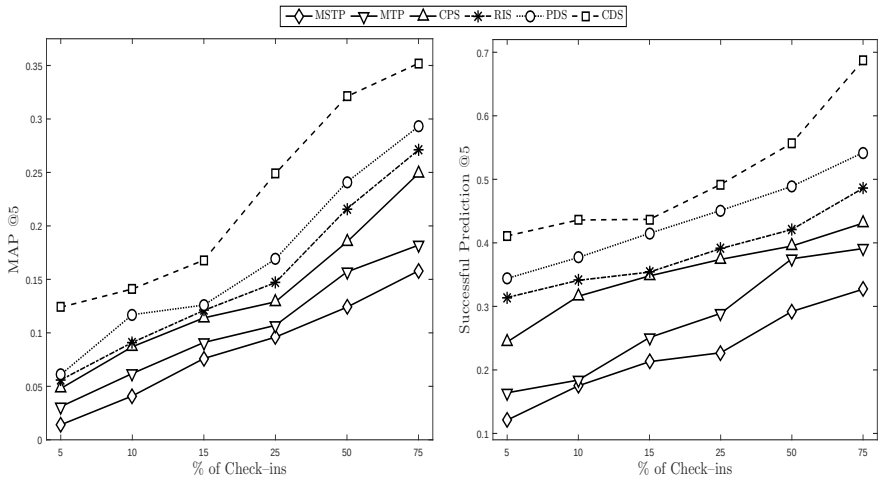

Figure 13: *MAP*, *Successful Prediction* vs Percentage of check-ins for 5 locations predicted.

user rating used in *MAE* calculation for recommender system is replaced by the probability with which a user visits a location in the original dataset and the probability of its visit predicted by *UCF* using the proposed similarity measure, respectively. Mean Absolute Error (*MAE*) in predicting locations for an active user u_1 can thus be formulated as:

$$MAE_{u_1} = \frac{1}{|N|}\sum_{i \in N} |prb_i - a_i| \tag{9}$$

where, N is the set of all ground truths from the original dataset. The probability with which an active user u_1 visits a location l_1 in the original dataset is represented as prb_{l_1}.

The *UCF* technique identifies the probability with which a user visits a location. Locations with higher probability are generally selected for prediction. Hence, for every instance we predict a collection of locations. The term a_i in Equation (9) is the mean of the probabilities with which the predicted locations are visited by u_1. The *MAE* for all the active users U is the average of *MAE* of the active users.

$$MAE = \frac{1}{|U|}\sum_{i \in U} MAE_{u_i} \tag{10}$$

Next we discuss the results obtained after using the similarity metrics in the location prediction problem. Figure 13, 14 and 15 show the performance of the predicted locations using various similarity metrics in terms of *MAP* and *Successful Prediction*. From our experiments, it was observed that the *MAP* increases to a considerable amount as the number of predicted locations are

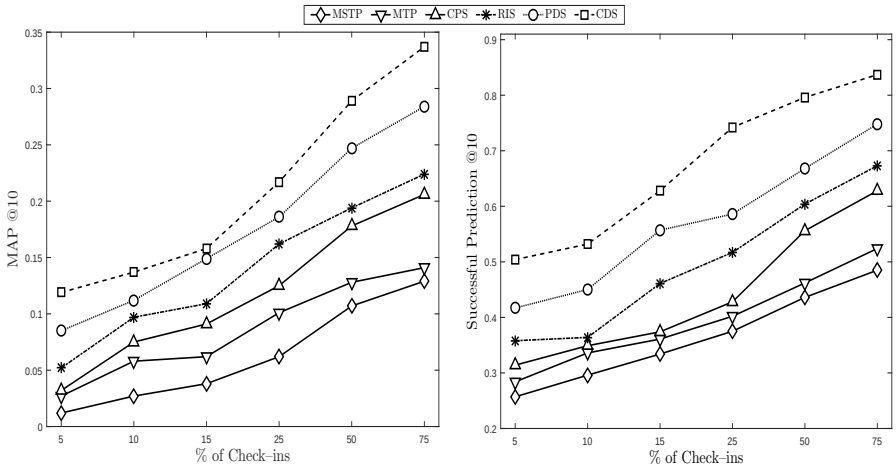

Figure 14: *MAP*, *Successful Prediction* vs Percentage of check-ins for 10 locations predicted.

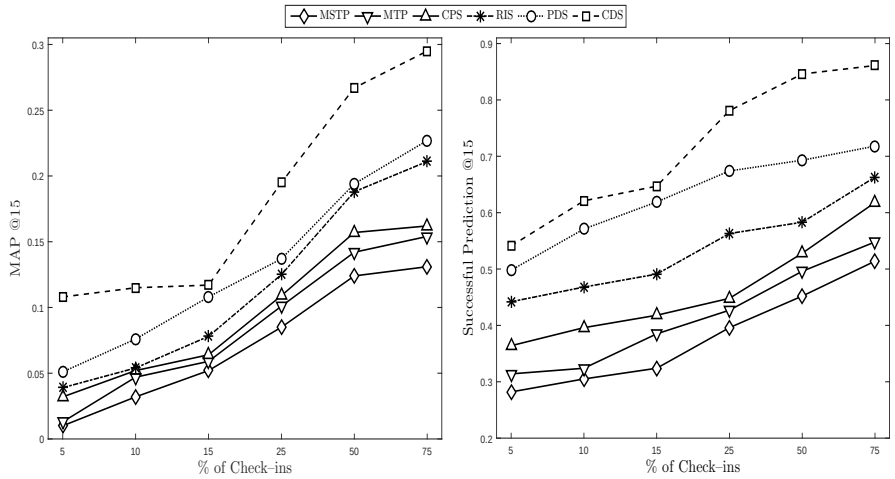

Figure 15: *MAP*, *Successful Prediction* vs Percentage of check-ins for 15 locations predicted.

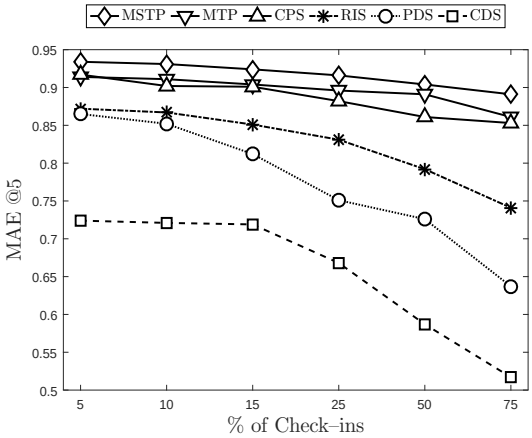

Figure 16: *MAE* vs Percentage of check-ins for 5 locations predicted.

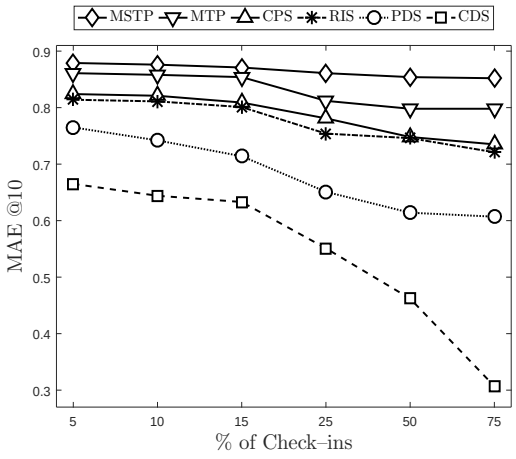

Figure 17: *MAE* vs Percentage of check-ins for 10 locations predicted.

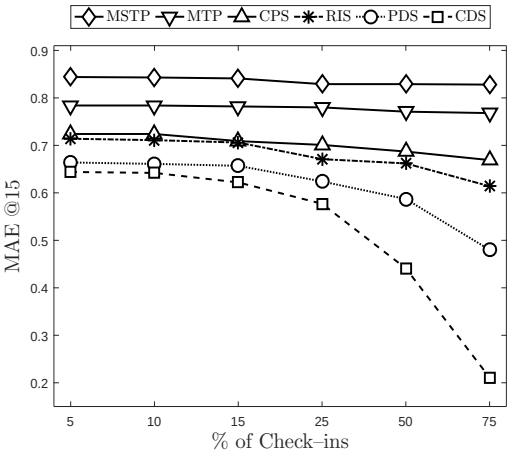

Figure 18: *MAE* vs Percentage of check-ins for 15 locations predicted.

increased from 5 to 15. Our proposed *CDS* metric produces consistently high *MAP* for 5, 10 and 15 predictions.

The *Successful Prediction* is computed in a similar way as depicted in Equation (5). In this case, if the ground truth is found in the predicted set of locations then the *Successful Prediction* is 1 else 0. The rate of *Successful Prediction* for all the metrics increase as we use more number of check-ins for predicting the next location. This shows that higher percentage of historical data carries more information on user mobility. The *CPS* and *RIS* methods utilize the common patterns and their frequencies for similarity computation. Therefore, their performance is found to be close for all the three evaluation metrics. As we increase the number of predictions with the number of historical check-ins, the rate of *Successful Prediction* produced by *CDS* outperforms others.

Figure 16, 17 and 18 show the error in location prediction in terms of *MAE*. With increase in the percentage of check-ins, the absolute error (*MAE*) is reduced considerably for all the similarity metrics. However, for the *MSTP* and *MTP* metrics, we observed less difference in the error metric as the percentage of check-ins are increased. In *CPS* metric, the *MAE* values are significantly reduced as the number of predictions are increased. Incidentally, noticeable changes are observed for our proposed *CDS* metric as we increase the number of predictions. It can be noted here that the *CDS* approach considers the check-in distribution on each day of a week.

6. Conclusion

In this work, we studied the identification of the nearest neighbors by modeling user mobility patterns. The proposed framework first identifies significant locations in a trajectory where a user have stayed instead of passing by.

Subsequently, the neighbors are identified using three proposed similarity metrics. Finally, a ranking technique is used to rank the neighbors on the basis of their similarity scores. The obtained results justify that the proposed metrics can effectively be used in user similarity computation. The proposed approach towards identifying similarity between a pair of users can be used in various real-world problems like recommender services, behavior analysis from spatio-temporal data, community detection in a social network, identification of hidden social links in a network, etc. In future, we plan to extend this work on Geosocial network dataset where users share locations along with their real-time experiences.

7. References

[1] F. Hopfgartner, J. M. Jose, Semantic user profiling techniques for personalised multimedia recommendation, Multimedia systems 16 (4) (2010) 255–274.

[2] L. Backstrom, E. Sun, C. Marlow, Find me if you can: improving geographical prediction with social and spatial proximity, in: Proceedings of the 19th International Conference on World Wide Web, 2010, pp. 61–70.

[3] F. Giannotti, M. Nanni, F. Pinelli, D. Pedreschi, Trajectory pattern mining, in: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2007, pp. 330–339.

[4] J. J.-C. Ying, W.-C. Lee, V. S. Tseng, Mining geographic-temporal-semantic patterns in trajectories for location prediction, ACM Transactions on Intelligent Systems and Technology 5 (1) (2014) 1–33.

[5] H. Gao, J. Tang, H. Liu, Addressing the cold-start problem in location recommendation using geo-social correlations, Data Mining and Knowledge Discovery 29 (2) (2015) 299–323.

[6] M. Fire, L. Tenenboim-Chekina, R. Puzis, O. Lesser, L. Rokach, Y. Elovici, Computationally efficient link prediction in a variety of social networks, ACM Transactions on Intelligent Systems and Technology 5 (1) (2014) 1–25.

[7] Z. Wang, J. Liao, Q. Cao, H. Qi, Friendbook: a semantic-based friend recommendation system for social networks, IEEE Transactions on Mobile Computing 14 (3) (2015) 538–551.

[8] H. Lu, Q. Zhao, Z. Gan, A community detection algorithm based on the similarity sequence, in: Proceedings of the 15th International Conference on Web Information Systems Engineering, 2014, pp. 63–78.

[9] J. J.-C. Ying, E. H.-C. Lu, W.-C. Lee, T.-C. Weng, V. S. Tseng, Mining user similarity from semantic trajectories, in: Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks, 2010, pp. 19–26.

[10] X. Chen, J. Pang, R. Xue, Constructing and comparing user mobility profiles for location-based services, in: Proceedings of the 28th Annual ACM Symposium on Applied Computing, 2013, pp. 261–266.

[11] X. Chen, R. Lu, X. Ma, J. Pang, Measuring user similarity with trajectory patterns: Principles and new metrics, in: Proceedings of the 16th Asia-Pacific Web Conference, 2014, pp. 437–448.

[12] X. Cao, G. Cong, C. S. Jensen, Mining significant semantic locations from gps data, in: Proceedings of the VLDB Endowment 3 (1-2) (2010) 1009–1020.

[13] F. Giannotti, M. Nanni, D. Pedreschi, F. Pinelli, Mining sequences with temporal annotations, in: Proceedings of the 2006 ACM symposium on Applied computing, 2006, pp. 593–597.

[14] Y. Zheng, L. Liu, L. Wang, X. Xie, Learning transportation mode from raw gps data for geographic applications on the web, in: Proceedings of the 17th International Conference on World Wide Web, 2008, pp. 247–256.

[15] Z. Li, J. Han, B. Ding, R. Kays, Mining periodic behaviors of object movements for animal and biological sustainability studies, Data Mining and Knowledge Discovery 24 (2) (2012) 355–386.

[16] H. Jeung, M. L. Yiu, X. Zhou, C. S. Jensen, Path prediction and predictive range querying in road network databases, The VLDB Journal 19 (4) (2010) 585–602.

[17] L. Chen, M. Lv, G. Chen, A system for destination and future route prediction based on trajectory mining, Pervasive and Mobile Computing 6 (6) (2010) 657–676.

[18] L. Chen, M. Lv, Q. Ye, G. Chen, J. Woodward, A personal route prediction system based on trajectory data mining, Information Sciences 181 (7) (2011) 1264–1284.

[19] Y. Zheng, L. Zhang, Z. Ma, X. Xie, W.-Y. Ma, Recommending friends and locations based on individual location history, ACM Transactions on the Web 5 (1) (2011) 1–44.

[20] M. Lv, L. Chen, G. Chen, Mining user similarity based on routine activities, Information Sciences 236 (2013) 17–32.

[21] J. Bao, Y. Zheng, D. Wilkie, M. Mokbel, Recommendations in location-based social networks: a survey, GeoInformatica 19 (3) (2015) 525–565.

[22] Q. Li, Y. Zheng, X. Xie, Y. Chen, W. Liu, W.-Y. Ma, Mining user similarity based on location history, in: Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, 2008, pp. 1–10.

[23] X. Xiao, Y. Zheng, Q. Luo, X. Xie, Finding similar users using category-based location history, in: Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, 2010, pp. 442–445.

[24] J. R. Bray, J. T. Curtis, An ordination of the upland forest communities of southern wisconsin, Ecological monographs 27 (4) (1957) 325–349.

[25] X. Xiao, Y. Zheng, Q. Luo, X. Xie, Inferring social ties between users with human location history, Journal of Ambient Intelligence and Humanized Computing 5 (1) (2014) 3–19.

[26] H. Ma, H. Cao, Q. Yang, E. Chen, J. Tian, A habit mining approach for discovering similar mobile users, in: Proceedings of the 21st International Conference on World Wide Web, 2012, pp. 231–240.

[27] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, M.-C. Hsu, Prefixspan: mining sequential patterns efficiently by prefix-projected pattern growth, in: Proceedings of the 17th International Conference on Data Engineering, 2001, pp. 215–224.

[28] R. Agrawal, R. Srikant, Mining sequential patterns: Generalizations and performance improvements, in: Proceedings of the 5th International Conference on Extending Database Technology, 1996, pp. 1–17.

[29] M. J. Zaki, Spade: An efficient algorithm for mining frequent sequences, Machine learning 42 (1) (2001) 31–60.

[30] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, M.-C. Hsu, Freespan: frequent pattern-projected sequential pattern mining, in: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2000, pp. 355–359.

[31] Y. Zheng, Q. Li, Y. Chen, X. Xie, W.-Y. Ma, Understanding mobility based on gps data, in: Proceedings of the 10th International Conference on Ubiquitous Computing, 2008, pp. 312–321.

[32] Y. Zheng, L. Zhang, X. Xie, W.-Y. Ma, Mining interesting locations and travel sequences from gps trajectories, in: Proceedings of the 18th International Conference on World Wide Web, 2009, pp. 791–800.

[33] Y. Zheng, X. Xie, W.-Y. Ma, Geolife: A collaborative social networking service among user, location and trajectory., IEEE Data Eng. Bull. 33 (2) (2010) 32–39.

[34] J. Wang, A. P. De Vries, M. J. Reinders, Unifying user-based and item-based collaborative filtering approaches by similarity fusion, in: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2006, pp. 501–508.

[35] V. W. Zheng, Y. Zheng, X. Xie, Q. Yang, Collaborative location and activity recommendations with gps history data, in: Proceedings of the 19th International Conference on World Wide Web, 2010, pp. 1029–1038.

[36] Z. Huo, X. Meng, R. Zhang, Feel free to check-in: Privacy alert against hidden location inference attacks in geosns, in: Proceedings of the 18th International Conference on Database Systems for Advanced Applications, 2013, pp. 377–391.

[37] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th International Conference on World Wide Web, 2001, pp. 285–295.

[38] R. Jin, J. Y. Chai, L. Si, An automatic weighting scheme for collaborative filtering, in: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2004, pp. 337–344.

[39] G.-R. Xue, C. Lin, Q. Yang, W. Xi, H.-J. Zeng, Y. Yu, Z. Chen, Scalable collaborative filtering using cluster-based smoothing, in: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2005, pp. 114–121.

[40] J. Wang, A. P. De Vries, M. J. Reinders, Unifying user-based and item-based collaborative filtering approaches by similarity fusion, in: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2006, pp. 501–508.

[41] M. Á. García-Cumbreras, A. Montejo-Ráez, M. C. Díaz-Galiano, Pessimists and optimists: Improving collaborative filtering through sentiment analysis, Expert Systems with Applications 40 (17) (2013) 6758–6765.

[42] S. Dooms, T. De Pessemier, L. Martens, Online optimization for user-specific hybrid recommender systems, Multimedia Tools and Applications 74 (24) (2015) 11297–11329.