# Spatio-Temporal Site Recommendation

Blinded for Double-Blind review

***Abstract*—Recommendation systems have become extremely common in recent years, and are utilized in a variety of areas to predict the "rating" or "preference" that a user would give to a point of interest (PoI), such as a restaurant, a hotel, or a bar. Such systems typically produce a list of recommendations by considering previous ratings of the user, as well as ratings of other users. Not every person rates every point of interest they visit. In this work, we want to explore the use of spatio-temporal data to improve recommendation systems: We postulate that spatio-temporal user data may indicate the liking or disliking of a point of interest. Clearly, if a user frequently visits the same PoI, stays at the PoI for long times, and is willing to travel a long distance to visit a PoI, that might indicate that user likes that PoI. Thus, we propose to extract user-PoI relation features from spatio-temporal trajectory data only. Using these features, we use out-of-the-box data mining and machine learning solutions, to estimate the popularity of a PoI. Our experimental evaluation shows, that the features extracted from spatio-temporal data able to accurately predict the popularity of a PoI, using ground-truth data from Yelp as a baseline.**

## I. Introduction

Modern technology to capture geo-spatial information produces a huge flood of individual trajectory data, coupled with a new user mentality of utilizing this technology to voluntarily share information. By mining this data, and thus turning it into actionable information, the McKinsey Global Institute [5] projects a "$600 billion potential annual consumer surplus from using personal location data globally". Towards this goal of making location data actionable, we propose to mine potential user ratings for location sites (e.g., restaurants, shops) from spatial-temporal data. User rating is critical in many applications, especially recommendation system. Techniques such as widely used collaborative filtering use known user rating information to estimate the interests of a user. However, in practice, the user-site rating matrix is usually highly sparse, meaning that there are not enough user rating information to perform such tasks. This is know as the cold-start problem. There are many reasons, for example, many users do not want to spend time to rate locations they visited, or locations such as shops do not have an efficient way to ask users for feedback.

To solve this problem, we propose to obtain implicit user-rating rather than explicit user ratings. Our approach uses trajectory data to find users that have likely visited a site. Since this data does not explicitly tell us whether the user likes that site, we propose to mine features from the users trajectory, which we intuitively expect to implicitly describe whether the user likes that site. The features that we propose to obtain from trajectory data include:

- **The frequency of visits of a user.** If a user visits a site only once in their life, chances are that the user did not like the site enough to return. If the site if frequented often by the user, he seems to like it.
- **The length of stay of a user.** If a user stays at a restaurant or a hotel for a long time, that indicates that he likes the site.
- **The distance to their home base.** If a user is willing to take a long journey to reach a site, thus bypassing other, similar sites, that indicates that the user is subject to a strong attraction from that site, indicating that the user likes that site.

There are several advantages of using location data for location rating: (i) A potential solution to the cold-start problem, (ii) no user effort to capture their recommendation such as filling in rating forms, and (iii) it is based on user's behavior, thus more objective and prone to alteration by fake-user-ratings and spam/bot-user-ratings.

Our approach becomes viable, to due to the abundance of large open-sources collections of voluntarily contributed trajectory data, including the following data sources:

- **Location-Based Social Networks (LBSNs)** allows user to "Check-in" into a physical site such as a hotel, a restaurant or a metro station. Fairly large LBSN datasets, made anonymous, are available publicly. For instance, the FourSquare dataset used in [8] contains more than 30 million checkins and is available publicly. Such data explicitly includes the sites that a user has visited.
- **Geocoded Social Media Data:** is obtainable from public streaming APIs including for Twitter, Instagram, and Flickr. These data sources provide low-frequency trajectory data. Yet, microblogs and images are often published in sites of interest to a user, thus giving implicit information about the users site preferences.
- As part of the effort to create **Open-Street-Map (OSM)** [6], [2] road network data, users have been uploading GPS traces of their routes to the OSM site. While these routes are typically used to digitize the road network, the majority of routes is uploaded by pedestrians, and can be used as an indicator of sites that the corresponding user frequents. These trajectories are publicly available through the OSM API.

To describe our approach of site-recommendation using trajectory data, the rest of this work is organized as follows. We survey the state-of-the-art on site-recommendation and location-based recommendation systems in Section II. Then, we formally define the problem location-based site recommendation in Section III. Our solution, using deep-learning to bridge the gap from user-behaviour to user-site-recommendations is given in Section V. Our solution is evaluated in Section VII, showing that our rating prediction for a restaurant is able to closely predict authoritative ground-truth site-ratings obtained from Yelp. We conclude our work in Section VIII.

## II. Related Work

## III. Problem Definition

In this section we formally define a user-trajectory, and define our notion of a user-site-stay, which we are going to use to extract features to estimate the affinity between a user and a site later in Section V. We first start by defining a trajectory as follows.

*Definition 1 (Spatio-Temporal Database):* Let $\mathcal{U}$ denote a set of unique user identifiers, let $\mathcal{G} = [-90, 90] \times [-180, 180]$ denote the space of longitude/latitude geo-coordinates, and the $\mathcal{T}$ denote the time domain. A *spatio-temporal database* $\mathcal{ST} \subseteq \mathcal{U} \times \mathcal{G} \times \mathcal{T}$ is a collection of triples $(id \in \mathcal{U}, (lat, long) \in \mathcal{G}, t \in \mathcal{T})$. Each triple $(u, s, t) \in \mathcal{ST}$ is called an observation.
We group a spatio-temporal database into observations of the same user, denoted as user-trajectory, formally:

*Definition 2 (User-Trajectory):* Let $\mathcal{ST}$ be a spatio-temporal database and let $u \in \mathcal{U}$ be a user. The set

$$\mathcal{ST}(u) := \{(u', (lat, long), t) \in \mathcal{ST} | u = u'\}$$

is called the user-trajectory of user $u$.
In order to obtain recommendation information from a user-trajectory, we need to link the user-trajectory to sites, such as restaurants and hotels. For this purpose, we join a spatio-temporal database with a database of points of interest (such as provided by Open-Street Map) like restaurants and hotels. Next, we define our notion of a *stay*. A stay is an event of user visiting a site, enriched by the duration of the stay.

*Definition 3 (Stay Trajectory):* Let $\mathcal{ST}$ be a spatio-temporal database and let $\mathcal{S} \subseteq \mathcal{G}$ be a collection of $(lat, long)$ pairs of sites. A stay is a triple $(u \in U, s \in \mathcal{S}, (t_{\text{start}}, t_{\text{end}}) \in T \times T)$, indicating that user $u$ has stayed at site $s$ from time $t_{start}$ to time $t_{end}$. We let $(\mathcal{ST} \bowtie \mathcal{S})$ denote the set of all stays mined from all trajectories $\mathcal{ST}$ using all sites in $\mathcal{S}$. The sequence of all stays a user $u \in \mathcal{U}$ is called the stay-trajectory $(\mathcal{ST} \bowtie \mathcal{S})(u)$ of $u$, defined as:

$$(\mathcal{ST} \bowtie \mathcal{S})(u) := \{(u, p, (t_{\text{start}}, t_{\text{end}})) \in \mathcal{S} | u = u'\}$$

Finding stay points in trajectory and PoI databases is a research topic that has raised attention in the past. For instance, a state-of-the-art approach [4], [12], [11], [7] uses a distance threshold $\theta_d$ and defines a stay as the duration of time where the trajectory does not exceed a distance of $\theta_d$ to a PoI. In this work, we assume that *stay* detection algorithms are already applied to a trajectory, such that user-trajectory is mapped to a sequence of PoI stays. This assumption is discussed in Section **??**.

Given stay-trajectories for each user, the challenge of this work is to predict the rating between users and a site. As site is PoI which can be rated by a user, such as a restaurant or a hotel. We assume that we have a recommendation database, where users can rate sites. We assume normalized rating values in the interval $[0, 1]$, where $0$ corresponds to the lowest rating and $1$ corresponds to the highest rating.

*Definition 4 (User-Site Recommendation Database):* A recommendation database $\mathcal{R}$ is a set of user ratings $\mathcal{R} \subseteq \mathcal{U} \times \mathcal{S} \times [0, 1]$.
The task of this work is to predict the triples in $R$. That is, the challenge is to predict the rating that a user $u \in \mathcal{U}$ will give to a site $s \in \mathcal{S}$, using a spatio-temporal given in $\mathcal{ST}$.

## IV. Discussion: Stay Point Detection

The first step of our user-site-recommendation approach requires to map a raw trajectory $\mathcal{ST}(u)$ of a user $u$ to a stay-trajectory $(\mathcal{ST} \bowtie \mathcal{S})(u)$. Such stay points can be detected by using existing work such as proposed in [4], [12], [11], [7]. All of these works use a distance threshold $\theta_d$ and defines a stay as the duration of time where the trajectory does not exceed a distance of $\theta_d$ to a PoI. In this work, we entirely circumvent the step of implicit stay point detection, by using data that has explicit stay points. Therefore, in our experimental evaluation we use Check-in data from location based social networks (LBSNs), which explicitly contain the stay points of users. Clearly, by circumventing the problem of stay point detection, we limit our experimental evaluation to LBSN Check-in data, which is relatively small (hundreds of megabytes of data), compared to large raw trajectory databases (Terrabytes of data). Yet, we postulate that our relatively small FourSquare Check-in dataset [**?**], allows to effectively predict the user rating of a restaurant. This hypothesis is also supported by our experiments.

While we circumvent the problem of stay point detection by using Check-in data, we still need to estimate the duration of stay, as the duration of a stay is one of the features that we will use for prediction in Section **??**. Thus, we interpret each Check-in $c$ as a stay point. If the corresponding user has no other check-ins for the next six hours, we set the duration of $c$ to *UNKNOWN*. Otherwise, the set the duration to the time in-between these two check-ins.

## V. Spatio-Temporal User-Site Feature Extraction

Our goal is to extra a set of spatial-temporal feature from user trajectories, which can be used to predict the rating (in terms of a **score**, the higher the better) of a PoI from the users who have visited the PoI. There are many features that could potentially be related to a user's feeling of a PoI. We discuss in the section a set of common features we select for the rating prediction problem, and the methods we use to extra such features. It is easy to understand why these features are selected since they are intuitive.

### A. Extracting the frequency of visits of a user

Our assumption is, if a user repeatedly visits a PoI, e.g., always dine in the same restaurant, it strongly suggests that the user favours the PoI. On the other hand, if a user visits a PoI only once and never comes back, it suggests the user dislike the place. Therefore we choose the frequency of visits as a feature. Here, each stay-point is considered a visit.

In order to extra frequency information from a user's trajectories, we count for how many time does the user visit a location in a given time period. Typically, each check-in to the location is counted as one visit. Users are then assigned into several frequency groups (Table I) based on how often they visit the location. Note that if a user has never visited a PoI, he cannot rate it. Thus we do not assign any group for such users.

TABLE I: Groups based on how often user visits a location

| **Frequency groups** |
|---|
| At least one visit per day |
| At least one visit per week |
| At least one visit per month |
| Less than one visit per month |

We note a user's behaviour may vary over time. For example, if a user likes a shopping center, he may visit the shopping center very frequently during Thanksgiving and similar holidays, while visit the same place only once in several month during the rest of the year. As a result, the timing window used to count visits can have significant impact on the user's group assignment. For the same user, if we consider only visits happened in the Thanksgiving week, the user belongs to the "At least one visit per week " group. However, if we look at the year-long visits, he may be grouped into "Less than one visit per month" since his visits are averaged over twelve months. We propose a simple way to mitigate this problem.

First, we use timing windows with different sizes simultaneously to compute the frequency of visits of a user. Specifically, given a user's trajectory data for a period of $n$ days, for each day/week/month within these $n$ days, we count the number of visits that falls in the same day/week/month. We then use this count to compute the user's group of that specific day/week/month. Here we use day/week/month as nature timing windows, but in practice the size of timing window can be arbitrary. Second, when a user's visit frequency demonstrates inconsistency in different time period with the same timing window size, we use his maximal visiting frequency, e.g., if a user visits a location 3 times a week in week 1, but 0 times in week 2, the user is then categorized as "At least one visit per week" in this case. Given that each user has a unique and consistent ID, for each PoI, we can then calculate how many users falls in each frequency group.

Additionally, for each PoI, we calculate three numerical features that also describe the users' visiting pattern to the PoI: 1) **Average duration between two consecutive visits of a user**, 2) **Minimal duration between two consecutive visits of a user**, and 3) **Maximal duration between two consecutive visits of a user**. These information are supplemental to the frequency groups. We are interested in finding out their impact on the rating prediction result.

### B. Extracting the length of stays of a user

Given a trajectory dataset, the length of stays can be inferred by examining the interval between consecutive check-ins of the user. If two visits are reported within a time threshold $\tau$, e.g., 30-minutes, it is safe to consider the two check-ins belong to the same stay. Thus, the minimal length of this stay is the time frame between the two check-ins. Similarly, an upper bound of the length of stay is the time between a check-in to the location and the closest check-in to a different location. This provides us with a coarse way to estimate the average length of stay of users to a PoI. This method is most accurate if the user's location is reported periodically with small intervals, or whenever the user changes location.

### C. Extracting the travel distance of a user from its home base

If a user is willing to travel a long distance from his home/work area to a PoI, it is likely that the PoI is attractive to him. Unfortunately, measuring the travel distance from one's home/work can be challenging. This is because users' home or work address or coordinates are usually not explicitly marked in a spatial-temporal dataset due to privacy concerns.

We use distance-based location clustering (e.g., weighted k-means [3], where the weight of a location is the number of visits by the user) to estimate a user's home base, i.e., an area where he is likely to live/work. Locations a user visits frequently is likely to formulate a dense cluster in the area where he lives and works [1], [10]. If there exists a location that is isolated from any of these clusters, we deem such a location as "far from home". It requires extra effort for the user to travel this location. Willing to make such effort indicates the user may like the place. An example of home base and isolated locations is illustrated in Figure 1.

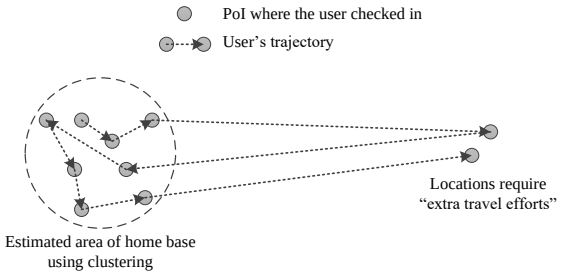


Fig. 1: Example of home base and isolated locations

For each isolated location, we estimate the travelling effort for the user to reach the location. Using the distance between the isolated location and the user's home base may be biased. A user who likes to drive can easily travel to a mall several miles away from home, but for someone who does not have

a car, travelling the same distance means significantly more effort. Instead, we use the relative travelling distance. For a location $l$, we compute $D_l = d_l/\overline{d_{home}}$ where $d_l$ is the minimal travel distance between $l$ and any home base location, and $\overline{d_{home}}$ the average travel distance between locations within the user's home base.

### D. Extracting the type of PoI

A user's spatial-temporal behaviour can be very different in different types of PoIs. Suppose a user gives high rate for both a coffee stand and a theatre. It is common for a user to visit the coffee stand every day or even multiple times a day, while such a visiting pattern is not likely to appear for his visiting to the theatre. Given the diversity of locations, we believe the type of different PoIs serves as an important feature in the rating prediction process.

Fortunately, in many spatial-temporal datasets (e.g., [9]), a category is given for each PoI. Nevertheless, a dataset may provide only coordinate of visited locations with no additional information. One way to infer the type of PoI in this case is to match the coordinates to PoIs using geoinfo systems such as Google Map, which provides detailed category information of PoIs.

### E. Discussion

Due to limited space, we briefly discuss some other useful features that could be explored for user rating prediction.

**Regional difference:** User's visiting pattern to some PoIs can be region-sensitive. Consider an example of two restaurants. Restaurant A locates near a busy train station while restaurant B is in suburb region. Even if the B provides better service than A, A may attract much more visitors comparing with B due to its better location. A possible way to mitigate the problem is to add a feature that describes region of a location (e.g., "Downtown region", "Less popular region", etc.) but it requires extra geo-demographic information which cloud be hard to collect. A simpler and efficient way is to explore the relation between nearby PoIs of the same type. The reason is intuitive. For example, given several restaurants close to each other, if all of them are visited by 100 users per day, except for one restaurants which has only 10 visitors per day, it is very likely that this restaurants has a bad rating.

**Scale of the PoI:** It is worth mentioning that the scale of a PoI has an obvious impact on the total number of visits to it. A small convenient store, regardless of how users like it, is not likely to have even close number of visitors comparing with a large super market. However, to our knowledge, these is no convincing way that can accurately estimate scale of a PoI. We leave this for future exploration when the data is available.

**Social connection between users:** The prevalence of LBSNs makes it easy to learn social connects between users. If several users who are socially connected check in at a PoI at the same time, it is likely that they have a group social event at the PoI. The fact that a group of friends choose to meet at a PoI suggests most of the users in the group like the place. Group-visits may demonstrate different pattern comparing with individual visits, and therefore can be used as a distinguished feature.

## VI. User-Site Rating Prediction

1. Simple linear regression - not enough expression power
2. Cov-DNN - Why?

## VII. Experimental Evaluation

### A. Experiment Settings

We evaluate the proposed technique over the FourSquare check-in dataset [9]. The dataset contains 227,428 check-ins in New York city and 573,703 check-ins in Tokyo collected in a duration of 10 month. We are interested in the user ID, PoI ID, check-in time stamp, and the PoI category of each record. We use the user rating of PoIs on FourSquare as ground truth. Specifically, FourSquare uses a 10-points based rating system. A high score indicates a good rating. The score is computed based on user's selection of three options: "Like" (10 points), "Neither like nor dislike" (5 points), or "Dislike" (0 point). For comparison purpose, we have implemented three schemes:

- Linear regression (LR) A baseline approach that predicts user ratings with a model generated by simple linear regression with the proposed features.
- Matrix factorization (MF) The standard non-negative matrix factorization [**?**] for recommender systems. We use matrix factorization to predict the rating of each users who have visited a PoI, then use their average rating score as the predicted rating of the PoI.
- Deep neural network (DNN) The scheme predicts user ratings with a model trained by deep neural network as proposed in Section VI, using the proposed features as input.

Key parameters of our experiment is given in Table II. For LR and DNN, we use the four types of features discussed in Section V. As for MF, it does not rely on any explicit feature. Our experiment platform is a virtual machine with Intel Xeon 64-bit 8-core CPU running on 2.93GHz and 32GB RAM. The algorithms are implemented with Microsoft Azure machine learning toolkit and modules.

TABLE II: Groups based on how often user visits a location

| Parameter | Default value |
|---|---|
| Number of users | 10000 |
| Number of PoIs | 5000 |
| Training set size (%) | 10% |
| Neural network depth (for DNN) | 3 |
| Number of latent features (for MF) | 8 |

### B. Data Cleaning

A few data cleaning steps are needed to prepare the dataset. First, we remove the PoIs which do not have user rating or were rated by less then 20 users. Second, we also remove the *inactive* users from the dataset. We define inactive users as those who have less than 2 check-ins per week in the 10

month period. The lack of data makes it hard to extra the aforementioned features based on these user's trajectory. After the cleaning process, we extract the check-in information of approximately 10000 users and 5000 PoIs from the dataset.

In order for MF to work, we generate a user-PoI rating matrix based on the users and PoIs appeared in the FourSquare dataset. The user ratings are directly crawled from FourSquare. It is possible because which PoI is liked or saved by a user is publicly available.

### C. Prediction Results

We first compare the Mean Square Error (MSE) of the prediction results of the three schemes. For each test point, we use a different number of number of users and PoIs, respectively, and exam the trend of performance of the compared schemes. The proposed DNN prediction method demonstrate clear advantage over the other two scheme, but even performance of the baseline method is very close to that of pure matrix factorization. The performance of DNN increases as the number of users increases. This implies a larger group of users demonstrates more reliable spatial-temporal features for rating prediction.

We also shows the MSE of the prediction result for PoIs in four different categories, namely "Food", "Shopping", "Fun", and "Nightlife". These PoI categories are provided by FourSquare. There appears to be a difference between PoI categories. For the proposed methods, the prediction result for "Food" and "Nightlife" is more accurate comparing with that of "Shopping" and "Fun". A closer look at check-in data in different PoI category reveals that users reported less check-ins for "Shopping" and "Fun" PoIs comparing with that of 'Food" and "Nightlife". Thus the relatively worse performance can be explained by the lack of data, which may affect the quality of extracted features.

## VIII. Conclusions

## References


[1] P. Golle and K. Partridge. On the anonymity of home/work location pairs. In *International Conference on Pervasive Computing*, pages 390–397. Springer, 2009.

[2] M. Haklay and P. Weber. Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, October-December:12–18, 2008.

[3] K. Kerdprasop, N. Kerdprasop, and P. Sattayatham. Weighted k-means for density-biased clustering. In *International Conference on Data Warehousing and Knowledge Discovery*, pages 488–497. Springer, 2005.

[4] Q. Li, Y. Zheng, X. Xie, Y. Chen, W. Liu, and 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*, page 34. ACM, 2008.

[5] J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh, and A. H. Byers. Big data: The next frontier for innovation, competition, and productivity. 2011.

[6] Open street map. http://www.openstreetmap.org.

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

[8] D. Yang, D. Zhang, L. Chen, and B. Qu. Nationtelescope: Monitoring and visualizing large-scale collective behavior in lbsns. *Journal of Network and Computer Applications*, 55:170–180, 2015.

[9] D. Yang, D. Zhang, V. W. Zheng, and Z. Yu. Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(1):129–142, 2015.

[10] H. Zang and J. Bolot. Anonymization of location data does not work: A large-scale measurement study. In *Proceedings of the 17th annual international conference on Mobile computing and networking*, pages 145–156. ACM, 2011.

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

[12] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma. Mining interesting locations and travel sequences from gps trajectories. In *Proceedings of the 18th international conference on World wide web*, pages 791–800. ACM, 2009.