Inferring Venue Visits from GPS Trajectories

Qihang Gu
Beijing University of Posts and Telecommunications
Beijing, China
guqihang@bupt.edu.cn

Dimitris Sacharidis
Technische Universität Wien
Vienna, Austria
dimitris@ec.tuwien.ac.at

Michael Mathioudakis
CNRS LIRIS & INSA Lyon
Lyon, France
michael.mathioudakis@liris.cnrs.fr

Gang Wang
VirginiaTech
Blacksburg, Virginia, USA
gangwang@vt.edu

ABSTRACT

Digital location traces can help build insights about how citizens experience their cities, but also offer personalized products and experiences to them. Even as data abound, though, building an accurate picture about citizen whereabouts is not always straightforward, due to noisy or incomplete data.

In this paper, we address the following problem: given the GPS trace of a person's trajectory in a city, we aim to infer what venue(s) the person visited along that trajectory, and in doing so, we use honest Foursquare check-ins as groundtruth. To tackle this problem, we address two sub-problems. The first is groundtruthing, where we fuse GPS trajectories with Foursquare check-ins, to derive a collection of detected stops and truthful check-ins. The second sub-problem is designing an inference model that predicts the check-in venue given a stop. We evaluate variants of the model on real data and arrive at a simple and interpretable model with performance comparable to that of Foursquare recommendations.

CCS CONCEPTS

• **Information systems → Location based services**;

KEYWORDS

check-ins, stop detection, venue inference, geographic choice

1 INTRODUCTION

As a larger part of modern life is digitized, individuals generate an increasing volume and variety of digital traces, which reveal information about their everyday activity and location. Users of online social networks often inform their online connections of their whereabouts, e.g., via *check-ins* on Foursquare or Facebook. If analyzed properly, such data can help us better understand how citizens experience the cities they live in.

Even as data abound, though, extracting accurate information from them is not always straightforward. A direction to obtain accurate estimates of people's activities is to combine data from different sources. We attempt to achieve this by combining GPS data, which provide a sample of a user's whereabouts but are noisy and lack semantics, with Foursquare check-ins that provide visits to venues of exact location but can be untruthful.

Specifically, the problem we consider is the following: given the GPS trace of a person's trajectories in a city, infer the venues they visit. Our approach to this problem entails two tasks. The first is *groundtruthing*, where we de-noise the GPS data and detect stops from it, then match these stops to Foursquare check-ins. The output of this task is a set of reliable Foursquare check-ins, each associated with a stop. Subsequently, based upon this groundtruth data, the second task is to predict the venue of a check-in, given the location of a trajectory. We refer to this task as *venue inference* and address it by defining appropriate probabilistic models. The techniques and models developed in these two tasks can then be applied on ad-hoc GPS trajectories of individuals and predict venues at which they did or would check-in — thus allowing us to infer a distribution of venue visits from GPS trajectories, or generate venue recommendations, depending on the application context.

To perform the second task, we employ the *geographic choice model* of Kumar *et al.* [3]. Given a geographic location as input, the model assigns probabilities to nearby venues proportionally to a *score* associated with each venue. Different specifications of the venue score lead to different instances of the model.

The primary contributions of this paper are (1) the fusion of GPS trajectory and check-ins, (2) an elaborate choice model that infers check-in venues from GPS trajectories, and (3) experimental results demonstrating the comparable performance in prediction accuracy (compared to Foursquare).

2 RELATED WORK

Urban computing is an active field of research, aiming to model and improve various aspects of urban life via the analysis of digital urban traces. We next discuss two major themes.

Human Mobility Analysis. One major line of work is that of modeling the movement of people in cities [1, 2, 6]. The central task here is to understand how individuals move from one place to the next in a city — and specifically, discover a universal pattern in terms of the geographic *distance* between the two successively-visited venues and *rank* (i.e., the number of venues within a same distance radius around the first venue). Findings suggest that rank exhibits a universal, power-law distribution across a large number of cities worldwide. One closely related task is studied by Kumar *et al.* [3] in the context of direction queries on Google Maps. The paper analyzed query logs from Google Maps to study how users

SIGSPATIAL'17, Los Angeles Area, CA, USA
© 2017 Copyright held by the owner/author(s). 978-1-4503-5490-5/17/11…$15.00
DOI: 10.1145/3139958.3140034

choose (request directions for) restaurants based on their distance and their rank compared to other restaurants.

Venue Recommendation based on Check-ins. Another body of related work focuses on venue recommendation by mining users' check-in data [4, 9, 10]. Most of existing work focuses on mining check-in histories of either users (and their friends) or venues to infer check-in preferences. Our work differs from existing work by exploring a cleaner and more interpretable model that does not use individual-user data. A model to predict the venue visited by a user given the user's location not only has direct applications to venue recommendations on LBSNs, but can also be used to estimate the discrepancy between the venues where users check-in and where they don't. One attempt to do that is [8], where our data comes from. The paper describes a basic version of the tasks we take on in this paper.

3 DATA

In this section, we first describe the dataset we use (Section 3.1), and subsequently the three steps for *groundtruthing*: denoising (Section 3.2), stop detection (Section 3.3), and data fusion (Section 3.4), leading to the groundtruth data used for the *visit inference* task.

3.1 Dataset

We use a dataset from previous work by Wang *et al.* [8, 11]. The dataset was collected during a user study, the goal of which was to compare the check-in activity of users on Foursquare to their actual whereabouts. For the purposes of the study, 372 users installed a GPS tracker on their smartphone and had their location tracked for a time of about two to three weeks. Moreover, the users had Foursquare installed on their smartphone and produced check-ins to inform their friends of the venues they visited. In total, the dataset contains $4,313,408$ GPS points (about $11,595$ per user), and $31,507$ Foursquare check-ins.

3.2 Trajectory Denoising

Manual inspection of the user trajectories revealed that the GPS traces were noisy. The bigger part (about 80%) of successive GPS traces for the same user have a distance up to 20 meters, but there are occasional gaps in the trajectories of several hundred meters. To identify truly noisy points, we also consider the estimated velocity between successive GPS points. Even as the larger part of successive GPS points reflect a velocity of less than $4m/sec$, there are occasional gaps of unnaturally large distance (e.g., larger than $50m/sec$).

To de-noise the trajectories, we follow an approach similar to the Heuristics-Based Outlier Detection method of [12]. Specifically, we define a distance threshold of 500 meters, and a speed threshold of 50 m/s. A trajectory point is filtered out if both the distance and speed w.r.t. its predecessor is above the thresholds, an event that happens about 6% of the time. The denoising algorithm examines each point in sequence. It removes all successors of the current point that do not satisfy the filter. At the end, the surviving points constitute the denoised trajectory.

3.3 Stop Detection

Given the GPS trajectory of each user, we extract *stops*, parts of the trajectory that correspond to the user's visit to a particular location. The intuition behind our approach is that a set of points represent a stop if they fall within a small geographic region for a long enough interval of time. To operationalize this, we need to determine what constitutes a "small region" and what a "long interval". We do this via two parameters ϵ and τ.

Specifically, for each GPS point p, we identify its *neighborhood* of immediately preceding or succeeding points that fall within a distance ϵ from p. More formally, the neighborhood N_p of point p is a maximal set of successive points q such that: (i) $p \in N_p$, (ii) $\|p - q\|_2 \le \epsilon$, where $\|\cdot\|_2$ denotes the L_2 distance between two points. We define a point p as *core* if the time for which the trajectory stays within its neighborhood exceeds time threshold τ, i.e., $dt(p) = \max_{q \in N_p} |t(p) - t(q)| \ge \tau$. Let C be the set of core points.

Based on the above definitions, we identify as *stops* $S \subseteq C$ a subset of the core points C that are the output of the following iterative procedure.

```
Repeat  while $C \neq \emptyset$:
        (1) Add to $S$ the core point $p \in C$ with maximum $dt(p)$.
        (2) Remove $p$ from $C$, along with all other core points $r$
            whose neighborhood $N_r$ overlaps with $N_p$.
```

Intuitively, the stops S we acquire at the end are a set of points near which the trajectory stayed for a long interval of time.

3.4 Data Fusion: Matching Stops to Check-Ins

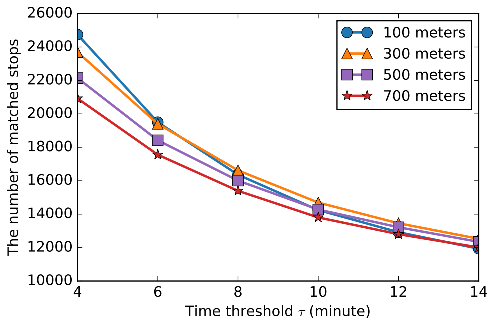
The final task in processing data is to match the stops detected in the previous step (Section 3.3) to actual check-ins and thus form our groundtruth dataset. We implement a matching algorithm similar to that in [11].

For each check-in, we determine all stops that are within distance α meters, and occur within β minutes after the check-in or anytime before. Among the candidate stops for a check-in, we select the one whose time interval is closest to the time of the check-in; then this check-in is called *matched*. The resulting pairs of a stop and check-in are then considered as *truthfull* and are included in the groundtruth data. Here, we settle to a definition of truthfullness that appears reasonable: a check-in can occur at most $\beta = 30$ minutes before the stop and anytime after, while the geographical distance between the check-in and stop should not exceed $\alpha = 500$ meters.

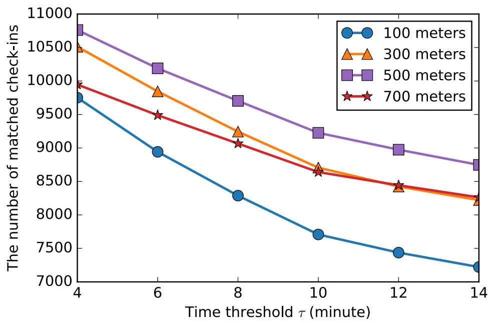
Optimizing Stop Detection. Having fixed the aforementioned definition of truthfullness, we come back to stop detection (Section 3.3) and how to set its parameters, ϵ and τ. On one hand, increasing the number of detected stops offers more opportunities for matched check-ins , as demonstrated in Figures 1a and 1b. On the other hand, an increased number of stops comes with an increased number of stops that remain un-matched – something we want to avoid. The appropriate stop detection parameters are thus determined after performing grid search in their domain space and aiming for values that lead to quantifying the number of stops produced and check-ins matched. Therefore, we set $\epsilon = 500m$ and $\tau = 8$ min, as $\epsilon = 500m$ leads to larger number of check-ins compared to other ϵ values, but also $\tau = 8$ min allows us to combine a large number of check-ins with somewhat limited number of stops.

4 MODELING

The *venue inference task* can be simply stated as follows: *given the location of a trajectory stop, predict the venue where the user checks in.* To approach the task, we build a model to assign probabilities to



(a) Matched stops.



(b) Matched check-ins.

Figure 1: The number of matched stops and check-ins as a function of ϵ and τ.

the different venues that might be visited by the user, conditional on the location at which the user has made a stop.

Motivated by the work of Kumar *et al.* [3], we adapt a *general choice model* describing the relative merit of a venue with respect to alternative options:

$$\mathbf{P}(v|m) = \frac{s(v,m)}{\sum_{v' \in \mathcal{V}_m} s(v',m)}, \tag{1}$$

where $\mathbf{P}(v|m)$ is the probability of a check-in at venue v given stop m, $\mathcal{V}_m$ is the set of all venues within distance of 500 meters to the stop m, and $s(v,m)$ assigns a score to venue v given stop m.

All examined models obey this general formulation and differ in the way they compute the venue scores. In particular, the variants define $s(v,m)$ as the product of *functions* defined over *features* associated with venue v and stop m.

4.1 Features

We consider the following two features.

Distance. For a venue v at location $v.l$, its distance $D(v,m)$ from the stop location $m.l$ is: $D(v,m) = \|v.l - m.l\|_2$.

Rank. For a venue v at location $v.l$, its rank $R(v,m)$ is the number of candidate venues that are closer to the stop location $m.l$, i.e.,

$$R(v,m) = |\{v' | v' \in V_m : \|v.l' - m.l\|_2 < \|v.l - m.l\|_2\}|.$$

In the interest of simplicity, we'll omit the designation v of the venue and m of the stop for these features. Since all our models require discrete values, we discretize *Distance* and *Rank* into 40 evenly spaced values from 0 up to the maximum distance (500 meters) or rank.

4.2 Scoring Functions

The score $s(v,m)$ of a venue v for stop location m is the product of a probability mass function (*pmf*) for each feature. We consider two types of *pmf* for each feature, *empirical* and *learned*.

Empirical Probability Mass Function. For a feature X, the empirical probability mass function $\mathbf{P}(X)$ is estimated as the relative frequency with which observed check-ins have a particular value $X = x$ for feature X in the training data.

Learned Probability Mass Function. For a feature X, a learned function $\mathbf{\Phi}(X)$ assigns to each (discrete) value of feature X a probability value estimated from the training dataset. For both features, the learned probability corresponds to a step function for values of feature X, where sets (or buckets) of successive feature values are assigned the same probability by the training procedure discussed in Section 4.4.

4.3 Models

To describe the models, we use mnemonic rules for the ingredients, listed below. At all times, function $f()$ is either the empirical probability mass $\mathbf{P}()$, or a learned function $\mathbf{\Phi}()$.

- D corresponds to a function of *Distance* $f(D(v,m))$;
- R corresponds to a function of *Rank* $f(R(v,m))$;

The models we consider define the score $s(v,m)$ of a venue v for stop m as a product of *ingredients*, where each ingredient is a function of a feature, that is: $s(v,m) = \prod_{i=1}^{n} f(X_i)$, where $f(X_i)$ is the function of the feature X_i.

4.4 Training

For a stop m, let $v(m)$ denote the matching check-in venue, extracted as described in Section 3.4. The data is a set of stop - check-in-venue pairs: $C = \{\ldots, \langle m, v(m)\rangle, \ldots\}$. Then, given a fully specified model instance, the likelihood of observing the data is defined as follows:

$$L = \prod_{C} \mathbf{P}(v(m)|m) = \prod_{C} \frac{s(v(m),m)}{\sum_{v' \in \mathcal{V}_m} s(v',m)}.$$

The log likelihood, LL(Q) is then:

$$LL = \sum_{C} \log s(v(m),m) - \sum_{C} \log\left(\sum_{v' \in \mathcal{V}_m} s(v',m)\right).$$

Training (i.e., the fitting of parameters) is performed through gradient ascent similar to [3]. Let X be a parameter, then at the i-th step of gradient ascent we update it as $X^{(i)} = X^{(i-1)} + \eta_X \cdot \frac{\partial LL}{\partial X}$, where η_X is the learning rate for parameter X. In the following, we derive the partial derivatives with respect to distance as an example.

The distance ingredient $\mathbf{\Phi}D$ contains a parameter $\mathbf{\Phi}D[d_i]$ for each discrete distance value d_i. The corresponding partial derivate of log-likelihood is then:

$$\frac{\partial LL}{\partial \mathbf{\Phi}D[d_i]} = \sum_{\substack{m \in C: \\ D(v(m).l, m.l) = d_i}} \frac{1}{\mathbf{\Phi}D[d_i]} - \sum_{m \in C} \frac{\sum_{v' \in \mathcal{V}_m : D(v'.l, m.l) = d_i} s(v',m)}{\sum_{v' \in \mathcal{V}_m} s(v',m)} \cdot \frac{1}{\mathbf{\Phi}D[d_i]},$$

Finally, following standard practice, we use a random but fixed 80% of check-ins as the training dataset, and evaluate model performance on the remaining 20% of check-ins.

Table 1: Distance or Rank

P(D)	Φ(D)	P(R)	Φ(R)
-7972.69	**-7925.77**	-8098.92	**-8025.41**

Table 2: Distance and Rank

	P(D)	Φ(D)
P(R)	-8538.68	-8193.85
Φ(R)	-7932.03	**-7937.45**

5 EVALUATION

5.1 General Choice Model

In this part of the section, we evaluate the performance of different instances of the general choice model.

Single-Feature Models. We begin evaluation with model instances that use a single feature, i.e., either *Distance*, or *Rank*. Evaluation is performed in terms of log-likelihood on the test dataset, and the results are shown in Tables 1; higher values are better. For reference, a model that assigns equal scores to all venues has log-likelihood of -8349.21. As shown in the table, the best model instance used learned functions for either distance or rank.

Two-Feature Models. We proceed similarly by considering model instances that combine distance and rank, and show the results in Tables2. For each feature, we evaluate model instances for each ingredient that was evaluated in single feature models. As shown, the model $\Phi(D) + \Phi(R)$ achieves the best performance which proves the benefit of feature combination.

The aforementioned results, seem to indicate that we obtain better performance for the *venue inference* task when: (i) both features are used; (ii) we use stepwise scoring functions with learned coefficients instead of pointwise marginal distributions.

5.2 Comparison with Foursquare

We take it as given that Foursquare uses more complex models than ours (see [7]), that better exploit the large amount of available data they have at their disposal. Nevertheless, for completeness, we wish to compare the performance of our approach with theirs.

As we do not have direct access to their algorithms, we proceed as follows. Given the location m of a trajectory stop, Foursquare provides us with a ranked list of venues[1], and evaluate them based on whether they include the matched check-in in our dataset. We evaluate similarly our models.

To evaluate, we borrow two measures from the field of *Information Retrieval* [5]. *N̲ormalized D̲iscounted C̲umulative G̲ain* (NDCG) considers the top k venues returned by a method and sums up their relevance. In our setting, only the matched check-in can have relevance score 1, while the rest is 0. Specifically, if i is the position[2] at which the check-in venue is returned, then

$$NDCG@k = \frac{1}{\log(i+1)}, \text{if } i \leq k; \quad 0, \text{otherwise.} \qquad (2)$$

The reported NDCG values for each method are averages over all stops in the test dataset and are reported in Table 3.

[1]We use the `venues/explore` API call (https://developer.foursquare.com/docs/venues/explore) to obtain a list of recommended venues.
[2]We use the term *position* here instead of the traditional term 'rank', to avoid confusion with the *Rank* feature.

Table 3: Ranking Effectiveness

Models	NDCG@1	NDCG@2	NDCG@5	NDCG@10	NDCG@20	MAP
Foursquare	0.149	0.192	0.245	0.278	0.301	0.238
Φ(D)	0.087	0.126	0.18	0.219	0.252	0.184
Φ(R)	0.091	0.134	0.187	0.226	0.261	0.192
Φ(D)+Φ(R)	0.087	0.127	0.182	0.222	0.257	0.186

M̲ean A̲verage P̲recision (MAP) considers the *average precision* of each stop and returns their mean value. *Average precision* in the special case of our setting is simply defined as $AP = \frac{1}{i}$, where i is the position at which the check-in venue is returned by a method.

As we see in Table 3, the ranking of Foursquare outperforms that of our models according to NDCG and MAP. This was expected, given the clear advantage that Foursquare has due to available data and the simplicity constraints we imposed on our models. We make the following observations. (i) Due to the discretization of features, the advantage of Foursquare over our model instances is more pronounced when we consider only the first few results. (ii) Our models have consistent relative performance across all metrics. (iii) NDCG@k and MAP do not capture exactly the performance in terms of likelihood, our models were trained for.

6 CONCLUSION AND FUTURE WORK

Our goal in this paper was to infer the venues visited by individuals from their trajectory. Given a new, ad-hoc trajectory of an individual, one can now use our procedure to pre-process the trajectory and use the developed model instance to assign probabilities to venues, and so capture the likelihood that the individual visit(ed) each of them. One immediate application for future work would be to explore how such a model helps improve recommendations for location-based social networks or other services. Another application is to apply the same process on a large set of trajectories that represent movements of citizens in a city, and thus obtain a distribution for visits at different venues and times.

REFERENCES

[1] E. Cho, S. A. Myers, and J. Leskovec. 2011. Friendship and mobility: user movement in location-based social networks. In *KDD*.
[2] Y. Han, J. Yao, X. Lin, and L. Wang. 2017. GALLOP: GlobAL feature fused LOcation Prediction for Different Check-in Scenarios. *TKDE* PP, 99 (2017), 1–1.
[3] Ravi Kumar, Mohammad Mahdian, Bo Pang, Andrew Tomkins, and Sergei Vassilvitskii. 2015. Driven by Food. In *WSDM*.
[4] Huayu Li, Yong Ge, Richang Hong, and Hengshu Zhu. 2016. Point-of-Interest Recommendations: Learning Potential Check-ins from Friends. In *KDD*.
[5] Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, and others. 2008. *Introduction to information retrieval.* Vol. 1. Cambridge university press Cambridge.
[6] Anastasios Noulas, Salvatore Scellato, Renaud Lambiotte, Massimiliano Pontil, and Cecilia Mascolo. 2012. A Tale of Many Cities: Universal Patterns in Human Urban Mobility. *PloS one* 7, 5 (2012), e37027–10.
[7] Blake Shaw, Jon Shea, Siddhartha Sinha, and Andrew Hogue. 2013. Learning to rank for spatiotemporal search. In *WSDM*.
[8] Gang Wang, Sarita Yardi Schoenebeck, Haitao Zheng, and Ben Y Zhao. 2016. "Will Check-in for Badges" - Understanding Bias and Misbehavior on Location-Based Social Networks.. In *ICWSM*.
[9] Hao Wang, Manolis Terrovitis, and Nikos Mamoulis. 2013. Location Recommendation in Location-based Social Networks Using User Check-in Data. In *SIGSPATIAL*.
[10] Haochao Ying, Liang Chen, Yuwen Xiong, and Jian Wu. 2016. PGRank: Personalized Geographical Ranking for Point-of-Interest Recommendation. In *WWW*.
[11] Zengbin Zhang, Lin Zhou, Xiaohan Zhao, Gang Wang, Yu Su, Miriam J. Metzger, Haitao Zheng, and Ben Y. Zhao. On the validity of geosocial mobility traces. In *HotNets*.
[12] Yu Zheng. 2015. Trajectory Data Mining: An Overview. *TIST* 6, 3 (2015), 29:1–29:41.