Location Influence in Location-based Social Networks

Muhammad Aamir Saleem
Department of Computer Science
Aalborg University, Denmark
maas@cs.aau.dk

Rohit Kumar
Department of Computer and Decision Engineering
Universite Libre de Bruxelles, Belgium
rohit.kumar@ulb.ac.be

Toon Calders
Department of Computer and Decision Engineering
Universite Libre de Bruxelles, Belgium
toon.calders@ulb.ac.be

Xike Xie
School of Computer Science and Technology
Suzhou Institute for Advanced Study
University of Science and Technology of China, China
xkxie@ustc.edu.cn

Torben Bach Pedersen
Department of Computer Science
Aalborg University, Denmark
tbp@cs.aau.dk

ABSTRACT

Location-based social networks (LBSN) are social networks complemented with location data such as geo-tagged activity data of its users. In this paper, we study how users of a LBSN are navigating between locations and based on this information we select the most influential locations. In contrast to existing works on influence maximization, we are not per se interested in selecting the users with the largest set of friends or the set of locations visited by the most users; instead, we introduce a notion of *location influence* that captures the ability of a set of locations to reach out *geographically*. We provide an exact on-line algorithm and a more memory-efficient but approximate variant based on the HyperLogLog sketch to maintain a data structure called *Influence Oracle* (Oracle *in short*) that allows to efficiently find a top-k set of influential locations. Experiments show that our algorithms are efficient and scalable and that our new location influence notion favors diverse sets of locations with a large geographical spread.

1. INTRODUCTION

One of the domains in social network analysis [1, 8, 18, 19] that received ample attention over the past years is *influence maximization* [14], which aims at finding influential users based on their social activity. Applications like viral marketing utilize these influential users to maximize the information spread for advertising purposes [4]. Recently, with the pervasiveness of location-aware devices, social network data is often complemented with geographical information. For instance, users of a social network share geo-tagged content such as locations they are currently visiting with their friends. These social networks with location information are called location-based social networks (LBSN). In LBSNs, the location information offers a new perspective to view users' social activities. This information can be utilized to provide more constructive marketing strategies. For example, unlike viral marketing which focuses on finding influential users and spreading the message via word of mouth marketing (WOMM), influential locations can be found and information can be spread using outdoor marketing (OOH) e.g., by putting advertisements on billboards and distributing promotional items on such locations.

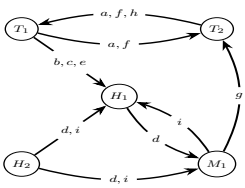
WSDM 2017, February 06 - 10, 2017, Cambridge, United Kingdom

ISBN 978-1-4503-4675-7/17/02…$15.00

DOI: http://dx.doi.org/10.1145/3018661.3018705

Check-ins

loc	Users		
	t=1	t=2	t=3
T_1	b, c, e, f	a, h	f
T_2	a, h	f, g	a
M_1	g	i	d
H_1	–	b, c, d, e	i
H_2	d, i	–	–



Friendships

User	Friends
a	c, h, i
b	d, f, i
c	a, f
d	b, e
e	d, g
f	c, b
g	e
h	a
i	a, b

Figure 1: Running example of a LBSN

In this paper, we study navigation patterns of users based on LBSN data to determine influential locations. Where other works concentrate on finding influential users [22], popular events [23], or popular locations [25], we are interested in identifying sets of locations that have a large *geographical* impact. Although often overlooked, the geographical aspect is of great importance in many applications. For instance, consider the following example.

Example 1. A marketer is interested in creating visibility of her products to the maximum regions in a city by offering free promotional items say T-shirts with a printed promotional message. To do that she would like to choose locations where she should distribute the promotional items to visitors.

In order to choose the most suitable locations for offering these items, not only the popularity of the places is important, but also the geographical reach. By visiting other

locations, people that were exposed to the advertisement, especially the receivers of the promotional items, may indirectly promote the products. For example, by wearing the shirt they expose the T-shirt's message to the people of the places they go to later and talk about it with their friends and relatives etc. Thus, when the goal is to create awareness of the product name, it may be preferable to have a moderate presence in many locations throughout the whole city rather than high impact in only a few locations. An illustration of this example is given in Figure 1. Nodes represent popular locations of different categories, such as tourist attractions (T_1, T_2), a metro station (M_1), and hotels (H_1 and H_2). Lowercase letters represent users. For each user, her friends in the social network and check-ins have been given. The top-2 locations with the maximal number of unique visitors are T_1, and M_1. The geographical impact of these locations, however, is not optimal; visitors of these locations reach only T_2 and H_1. On the other hand, the visitors of T_1 and H_2 visit all locations, i.e., users a, f and b, c, e visits T_2 and H_1 after visiting T_1, respectively, and d, i after H_2 visits H_1 and M_1.

To capture geographical spread and influence, in Section 3 we introduce the notion of a *bridging visitor* between two locations as a user that visits both locations within a limited time span. If there are many bridging visitors from one location to another, we say that there is an influence. We introduce different models that capture when the number of bridging visitors is considered to be sufficient to claim influence between locations. One model is based on the absolute number of visitors, one on the relative number, and we also have variants that take the friendship graph into account. Based on these models, we define influence for sets of locations and the *location influence maximization problem*: *Given a LBSN and a parameter k, find a set of k locations such that their combined location influence on other locations is maximal.*

To solve this problem, in Section 4 a data structure, called *Oracle*, is presented that maintains a summary of the LSBN data that allows to determine the influence of any set of locations at any time. Based on this data structure, we can easily solve the location influence maximization problem using a greedy algorithm. As for large LBSNs with lots of activities the memory requirements of our algorithm can become prohibitively large, we also develop a more memory-friendly version based upon the well-known HyperLogLog sketch [9].

In Section 5 we analyze several LBSNs to select reasonable threshold values for our models. In Section 6 the effectiveness and efficiency of our algorithms are demonstrated on these datasets. In a qualitative experiment, the effect of our new location influence notion is illustrated.

In summary, the main contributions of this paper are (i) the introduction and motivation of a new location influence notion based on LBSN data, (ii) the development of an efficient online Influence Oracle, and (iii) the demonstration of the usefulness of the location influence maximization problem in real-life LBSNs.

2. RELATED WORK

Influence maximization in the context of social networks has already been studied in much detail [12, 11, 5]. We focus here mainly on works that study the identification of influential users, events, or locations from LBSNs data. We divide the studies into two groups. The first group covers studies using check-ins as an additional source of data to identify influential users, whereas the second group utilizes the check-ins for finding influential locations.

Influential users and events. Zhang et al. [23] use social and geographical correlation of users to find influential users and popular events. Users with many social connections are considered influential as well as events visited by them. Similarly, Wu et al. [22] identify influential users in LBSNs on the basis of the number of followers of their activities (check-ins). Li et al. [15] and Bouros et al. [2] on the other hand, identify regionally influential users on the basis of their activities. The focus of the work by Wen et al. [21] and Zhou et al. [24] is to find and utilize the influential users for product marketing strategies such as word-of-mouth. Our focus, however, is to find influential *locations* that could be used, e.g., for outdoor marketing. None of the previous works applies directly to our problem.

Influential locations in LBSNs. Zhu et al. [25], Hai [13], and Wang et al. [20] study location promotion. Given a target location, their aim is to find the users that should be advertised to attract more visitors to this location. Doan et al. [7] computes the popularity ranks of locations based on the number of visitors. On the other hand, in Zhou et al. [24] study the problem of choosing an optimal location for an event such that the event's influence is maximized; that is, they aim at finding a single location which attracts most users.

Novelty. Our work is different from all of the above as we focus on finding a *set of influential locations* where influence is defined using visitors as a mean to spread influence to other locations. Applications include outdoor marketing by selecting locations with maximal geographical spread.

3. LOCATION-BASED INFLUENCE

We first provide preliminary definitions and then present location influence. Moreover, we formally define the *Oracle* problem and *Location Influence Maximization* problem.

3.1 Location-based Social Network

Let a set of users U and a set of locations L be given.

Definition 1. An *activity* is a visit/check-in of a user at a location. It is a triplet (u, l, t), where $u \in U$ is a user, $l \in L$ a location and t is time of the visit of u at l. The set of all activities over U and L is denoted $\mathcal{A}(U, L)$.

Definition 2. A *Location-based Social Network (LBSN)* over U and L consists of a graph $G_S(U, F)$, called *social graph*, where $F \subseteq \{\{u, v\} | u, v \in U\}$ represents friendships between users, and a set of activities $A \subseteq \mathcal{A}(U, L)$. It is denoted $LBSN(G_S, A)$.

3.2 Models of Location-based Influence

We define the influence of a location by its capacity to spread its visitors to other locations. The intuition behind this is to capture a location on the basis of its ability to spread its visitors that are exposed to a message, to other locations. Thus, the location influence indirectly captures the capability of a location to spread a message to other geographical regions. Recall our running example that depicts the influence of locations in Figure 1. We can further filter the locations on the basis of their categories to find the

particular type of influential and influenced locations. For example, in Figure 1, by considering the hotels as influential locations and their influence only on tourist attractions (influenced locations), the most influential hotels can be found which can spread the information to the maximum number of tourist attractions. The effect of an activity in a location, however, usually remains effective only for a limited time. We capture this time with the *influence window* threshold ω. Visitors that travel from one location to another within a time ω are called *Bridging visitors*:

Definition 3. Bridging Visitor: Given $LBSN(G_S, A)$ and ω, a user u is said to be a *bridging visitor from location s to location d* if there exist activities $(u, s, t_s), (u, d, t_d) \in A$ such that $0 < t_d - t_s \leq \omega$. We denote the set of all bridging visitors from s to d by $V_{B(\omega)}(s, d)$.

The influence of a location s is measured by two factors, i.e., the number of locations that are influenced by s and the impact by which s influences the locations. The impact of an influence between two locations s and d is captured by the influence models (M).

3.2.1 Absolute Influence Model (M_A)

In practice, if a significant number of people perform an activity, then it is considered compelling. Thus, in order to avoid insignificant influences among locations, we use a threshold τ_A. The influence of a location s on a location d is considered only if the number of bridging visitors from s to d is greater than τ_A. The influence of a location s on d under M_A is represented by $I_{A(\omega,\tau_A)}(s, d)$:

$$I_{A(\omega,\tau_A)}(s,d) := \begin{cases} 1, & \text{if } |V_{B(\omega)}(s,d)| \geq \tau_A \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

We omit ω and τ_A from the notations when they are clear from the context.

Example 2. Consider the running example of Figure 1. Let $\tau_A = 2$ and $\omega = 2$. Then, $I_A(T_1, H_1) = 1$ because $|V_B(T_1, H_1)| = 3$ ($\geq \tau_A$). Similarly, $I_A(H_2, H_1) = 1$. However, $I_A(M_1, H_1) = 0$ because $|V_B(M_1, H_1)| = 1 (\not\geq \tau_A)$.

The influence between two locations may change with the value of τ_A and ω. For example, if we update the value of τ_A to 3 and ω to 2, $I_A(T_1, H_1) = 1$, however, $I_A(H_2, H_1)$ becomes 0 because $|V_B(H_2, H_1)| = 2 (\not\geq \tau_A)$.

3.2.2 Relative Influence Model (M_R)

In M_A, the influences of two pairs of locations are considered equal as long as the number of their bridging visitors is greater than τ_A. Sometimes, however, the relative number of contributed bridging visitors is important. Consider, for example, a popular location s that attracts many visitors and a non-popular location d with few visitors. In such a setting, to capture the influence of s on d, we may have to set the absolute threshold τ_A very low. This low value of τ_A, however, may result in many other popular locations being influenced by s even if only a very small fraction of their visitors come from s. Therefore, in such situations, it may be beneficial to use different thresholds for different destinations, relative to the number of visitors in these destination locations. This notion is captured by the *relative influence model (M_R)*. The influence of s on d under M_R is represented by $I_{R(\omega,\tau_R)}(s, d)$ and is parameterized by the relative threshold τ_R:

$$I_{R(\omega,\tau_R)}(s,d) := \begin{cases} 1, & \text{if } \dfrac{|V_{B(\omega)}(s,d)|}{|V(d)|} \geq \tau_R \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

where $V(d)$ is the set of users who visited location d.

Example 3. Consider the running example given in Figure 1. Let $\tau_R = 0.4$ and $\omega = 2$. In this example, $I_R(T_1, H_1) = 1$ because $\frac{|V_B(T_1,H_1)|}{|V_{H_1}|} = \frac{|\{b,c,e\}|}{|\{b,c,d,e,i\}|} = \frac{3}{5} \geq \tau_R$, Similarly, $I_R(H_2, H_1) = 1$ and $I_R(M_1, H_1) = 0$.

3.3 Friendship-based Location Influence

Activity data in LBSNs is often sparse in the sense that the number of check-ins per location is low. In Section 6 we see that in the real-world datasets we use there have only up to 6 check-ins per location on average. This sparsity of data affects the computation of location influence. In order to deal with this issue, we use the observation that users tend to perform similar activities as their friends (This claim is verified and confirmed in Section 5). Hence, we define friendship-based influence between locations, by incorporating also friends of bridging visitors, which we consider *potential visitors*. The set of bridging visitors together with the potential visitors from a location s to d is represented by $V_{Bf(\omega)}(s, d)$, and the set of visitors to a location d together with their friends is denoted $V_f(d)$.

In order to incorporate potential visitors in the influence models, we replace $V_{B(\omega)}(s, d)$ in Equation (1) and Equation (2) by $V_{Bf(\omega)}(s, d)$, and $V(d)$ in Equation (2) by $V_f(d)$. The updated influence of s on d under M_A and M_R respectively are represented by $I_{Af(\omega,\tau_{Af})}(s, d)$ and $I_{Rf(\omega,\tau_{Rf})}(s, d)$. Again, we omit ω, τ_{Af} and τ_{Rf} from the notations when it is clear from the context.

Example 4. Let $\tau_{Af} = 2$ and $\omega = 2$. We have $I_{Af}(T_1, H_1) = 1$ because $|V_{Bf}(T_1, H_1)| = |\{a, b, c, d, e, f, g, i\}|$ exceeds τ_{Af}. Similarly, $I_{Af}(H_2, H_1) = 1$ and $I_{Af}(M_1, H_1) = 1$.

Furthermore, let $\tau_{Rf} = 0.4$ and $\omega = 2$. We have $I_{Rf}(T_1, H_1) = 1$ because $\frac{|V_{Bf}(T_1,H_1)|}{|V_{H_1 f}|} = \frac{|\{a,b,c,d,e,f,g,i\}|}{|\{a,b,c,d,e,f,g,i\}|} = 1 (\geq \tau_{Rf})$. Similarly, $I_{Rf}(H_2, H_1) = 1$ and $I_{Rf}(M_1, H_1) = 0$.

3.4 Combined Location Influence

Based on the influence models, a location can influence multiple other locations. In order to capture such influenced locations, we define the *location influence set*:

Definition 4. Given a location s, and an influence model M, the *location Influence Set* $\phi_{I_M}(s)$ is the set of all locations for which the influence of s on that location under M is 1, i.e., $\phi_{I_M}(s) = \{d \in L \mid I_M(s, d) = 1\}$.

Next, we define *combined location influence* for a set of locations S. To do this, we use the following principled approach: any activity at one of the locations of S is considered an activity from S. In that way we can capture the cumulative effect of the locations in S; even though all locations in S in isolation may not influence a location d, together they may influence it. The bridging visitors from a set of locations S to d is represented by $V_{B(\omega)}(S, d)$:

$$V_{B(\omega)}(S,d) = \bigcup_{s \in S} V_{B(\omega)}(s,d) \quad (3)$$

The influence of a set of locations S on location d under M_A and M_R is defined similarly as for single locations.

Example 5. In Figure 1, let $\omega = 2$, $\tau_A = 3$ and $S = \{T_1, M_1\}$. Under M_A, $T_2 \notin \phi(T_1)$ and $T_2 \notin \phi(M_1)$. However, $T_2 \in \phi(S)$ as $|V_B(S, T_2)| = |\{a, f, g\}| \geq \tau_A$.

3.5 Problem Formulation

Based on these influence models, we now define two problems related to finding influential locations in a LBSN. We first present a problem statement of constructing a data structure that can be utilized for providing many interesting applications called Influence Oracle. Next, we present a problem statement for one such application, i.e., finding the top-k most influential locations.

Problem 1. (Oracle Problem) Given a $LBSN$ and an influence model M, construct a data structure that allows to answer: *Given a set of locations $S \subseteq L$ and a threshold τ, what is the combined location influence $\phi_{I_M}(S)$ of S.*

Problem 2. (Location Influence Maximization Problem) Given a parameter k, a $LBSN$, and an influence model M, the location influence maximization problem is to find a subset $S \subseteq L$ of locations, such that $|S| \leq k$ and the number of influenced locations $|\phi_{I_M}(S)|$ is maximum.

4. SOLUTION FRAMEWORK

We first provide a data structure to solve the Oracle problem. We present an exact algorithm in Section 4.1 and an approximate but more memory- and time-efficient algorithm in Section 4.2. Finally, in Section 4.3, we solve Problem 2 with a greedy algorithm.

4.1 Influence Oracle

In this section, we provide a data structure for maintaining location summaries for each location. We assume activities arrive continuously and deal with them one by one. The summary $\varphi(s)$ for a location s consists of the list of all locations to which it has bridging visitors. We present an online algorithm to incrementally update these summaries.

Definition 5. The *Complete location summary* for a location $s \in L$ is the set of locations that have at least one bridging visitor from s, together with these bridging visitors; i.e., $\varphi(s) := \{(d, V_B(s, d)) \mid d \in L \wedge |V_B(s, d)| > 0\}$.

If a user u visits a location s at time t, then u acts as a bridging visitor between all the locations u visited within the last ω time stamps and s. Therefore, for each user $u \in U$, we maintain a set of locations the user has visited and the corresponding latest visiting time. This is called the *visit history* $\mathcal{H}(u)$ and is defined as $\mathcal{H}(u) := \{(s, t_{max}) | u \in V(s), t_{max} = \max\{t \mid (u, l, t) \in A\}\}$. Suppose that we have the complete location summary for the check-ins so far and the visit history of all users, and a new activity (u, d, t) arrives. We update the complete location summary as follows: the location-time pair (d, t) is added in $\mathcal{H}(u)$ if d does not already appear in the visit history, otherwise the latest visit time of d is updated to t in $\mathcal{H}(u)$. Furthermore, for every other location-latest visit time pair (s, t') in the history of u, $\varphi(s)$ is updated by adding user u to the set of bridging visitors from s to d provided that the difference between the time stamps $t - t'$ does not exceed the threshold ω. This procedure is illustrated in Algorithm 1.

Example 6. We illustrate the algorithm using the running example shown in Figure 1. For simplicity, we only consider the activities of two users: d and i. We also add a new activity of d at H_2 at time stamp 5. In this example, we consider $\omega = 2$. The activities are processed one by one in increasing order of time. We show how the visit history $\mathcal{H}(i)$, $\mathcal{H}(d)$ and the complete location summaries $\varphi(H_1)$, $\varphi(H_2)$, $\varphi(M_1)$ evolve with different activities at different time stamp in Figure 2. Note, at time stamp 5 only $\varphi(M_1)$ is updated even though M_1 and H_1 are both in the visit histories of d because $\omega = 2$. The visit history of d is cleaned by removing H_1 from the $\mathcal{H}(d)$ as no future activities by d affect $\varphi(H_1)$. The visit time of H_2 is updated to the latest visit time. Similarly, $\mathcal{H}(i)$ is also cleaned up.

Algorithm 1: Updating complete location summaries

```
Input: New activity (u, d, t), threshold ω, φ(l) for l ∈ L
Output: Updated φ(.) and H(.)
begin
    foreach (s, t') ∈ H(u) do
        if t − t' ≤ ω then
            if (d, V_B(s, d)) ∈ φ(s) then
                V'_B(s, d) ← V_B(s, d) ∪ {u}
                φ(s) ← φ(s) \ {(d, V_B(s, d))}
            else
                V'_B(s, d) ← {u}
            φ(s) ← φ(s) ∪ {(d, V'_B(s, d))}
        else
            H(u) ← H(u) \ {(s, t')}
    if ∃t' : (d, t') ∈ H(u) then
        H(u) ← (H(u) \ {(d, t')})
    H(u) ← H(u) ∪ {(d, t)}
```

	$t = 1$	$t = 2$	$t = 3$	t=5
Activity:	$(i, H_2, 1)$ $(d, H_2, 1)$	$(i, M_1, 2)$ $(d, H_1, 2)$	$(i, H_1, 3)$ $(d, M_1, 3)$	$(d, H_2, 5)$
$\mathcal{H}(i)$:	$\{(H_2, 1)\}$	$\{(H_2, 1),$ $(M_1, 2)\}$	$\{(H_2, 1),$ $(M_1, 2),$ $(H_1, 3)\}$	$\{(H_1, 3)\}$
$\mathcal{H}(d)$:	$\{(H_2, 1)\}$	$\{(H_2, 1),$ $(H_1, 2)\}$	$\{(H_2, 1),$ $(H_1, 2),$ $(M_1, 3)\}$	$\{(M_1, 3),$ $(H_2, 5)\}$
$\varphi(H_1)$:	$\{\}$	$\{\}$	$\{(M_1, \{d\})\}$	$\{(M_1, \{d\})\}$
$\varphi(H_2)$:	$\{\}$	$\{(H_1, \{d\}),$ $(M_1, \{i\})\}$	$\{(H_1, \{d\}),$ $(M_1, \{i, d\})\}$	$\{(H_1, \{d\}),$ $(M_1, \{i, d\})\}$
$\varphi(M_1)$:	$\{\}$	$\{\}$	$\{(H_1, \{i\})\}$	$\{(H_1, \{i\}),$ $(H_2, \{i\})\}$

Figure 2: Updating $\varphi(l)$ and $\mathcal{H}$ for $\omega = 2$ for M_A

It can be observed from the example that a new activity of a user u only updates the complete location summary of the locations in the recent visit history of u. Notice that, since the activities of a user arrive in strictly increasing order of time, the size of $\mathcal{H}(u)$ is upper bounded by ω, as only locations that are visited within a time window ω are processed. The proofs of the following proposition are trivial and thus omitted.

Proposition 1. The time required to process an activity is $\mathcal{O}(\omega \log(|U|))$. The the complete location summary $\varphi(.)$

can be stored in $\mathcal{O}(|L||U|)$ memory and for the visit history $\mathcal{H}(.)$ in $\mathcal{O}(\omega)$ memory.

The time required to produce $\phi(S)$ from $\varphi(.)$ for given threshold τ and set of locations S is $\mathcal{O}(|S||L||U|\log|U|)$.

Relative and Friendship-based Location Influence. For the relative models, we additionally have to maintain the total number of unique visitors per location, which can be done in the worst case time $O(log(|U|))$ and space $O(|U|)$ per activity and hence does not affect the overall complexity. For the friendship-based location influence, for every activity, we process the same activity at the same time for all friends as well. As the number of friends is bounded by $|U|$, we get:

Proposition 2. The time required to process an activity in the friendship-based influence models is $\mathcal{O}(\omega|U|\log(|U|))$. The memory required is the same as for the other models.

4.2 Approximate Influence Oracle

In the worst case the memory requirements of the exact algorithm presented in the last section are quite stringent: for every pair of locations (s,d), in $\varphi(s)$ the complete list of bridging visitors from s to d is kept. Therefore, here we present an approximate algorithm for maintaining the complete location summaries in a more compact form. This compact representation will represent a significant saving especially in those cases where the window size ω is large since in that case the number of bridging visitors increases.

We observe that when computing the number of bridging visitors between s and d we do not need the set of bridging visitors between s and d, but only the cardinality of that set. For the relative number of bridging visitors, we additionally need only the numbers of visitors $|V(s)|$. Furthermore, as per Equation 3, in order to find the accumulated complete location summary, we need to combine two complete location summaries; for instance: the complete location summary $\varphi(\{s_1,s_2\})$ is obtained by taking the following pairwise union of $\varphi(s_1)$ and $\varphi(s_2)$: if $\varphi(s_1)$ and $\varphi(s_2)$ respectively contain the pairs $(d,V_B(s_1,d))$ and $(d,V_B(s_1,d))$, then $\varphi(\{s_1,2_2\})$ contains $(d,V_B(s_1,d)\cup V_B(s_2,d))$. But then again, for further computations, we only need the cardinality of the bridging visitor sets. Hence, if we accept approximate results, we could replace the exact set $V_B(s,d)$ with a succinct sketch of the set that allows to take unions and get an estimate of the cardinality of the set. In our algorithm, we use the HyperLogLog sketch (HLL) [9] to replace the exact sets $V_B(s,d)$ and $V(s)$. The HLL sketch is a memory-efficient data structure of size 2^k that can be used to approximate the cardinality of a set by using an array. The constant k is a parameter which determines the accuracy of the approximation and is in our experiments in the order of 6 to 10. Furthermore, the HLL sketch allows unions in the sense that the HLL sketch of the union of two sets can be computed directly from the HLL sketches of the individual sets. For our algorithm, we consider the HLL algorithm as a black box. By using HLL, we not only reduces memory consumption but also improve computation time, because adding an element in a HLL sketch can be done in constant time and taking the union of two HLL sketches takes time $\mathcal{O}(2^k)$; that is: the time to take the union of two sets is independent of the size of the sets.

Proposition 3. Let $b=2^k$ be the number of buckets in the HLL sketch. The time needed to process an activity using the HLL sketch is $\mathcal{O}(\omega)$. The memory required to maintain the complete location summary is $\mathcal{O}(|L|b)$.

4.3 Influence Maximization

In order to solve the location influence maximization problem, we apply the standard greedy algorithm to compute top-k as obtaining an exact solution is intractable as the next proposition states.

Proposition 4. The following problem is **NP**-hard for all influence models: given a LBSN and bounds k and β, does there exist a set of locations S of size k such that $|\phi(S)|\geq\beta$.

Proof. **NP**-hardness follows from a reduction from set cover. Consider an instance $\mathcal{S}=\{S_1,\ldots,S_m\}$ with all $S_i\subseteq\{1,\ldots,n\}$ and bound k of the set cover problem: does there exist a subset $\mathcal{S}'$ of $\mathcal{S}$ of size at most k such that $\bigcup\mathcal{S}'=\{1,\ldots,n\}$. We reduce this instance to a LBSN as follows: $L=\{l_1,\ldots,l_n\}\cup\{s_1,\ldots,s_m\}$, $U=\{u_1,\ldots,u_m\}$, $F=\emptyset$, $A=\{(u_i,s_i,0)\mid i=1\ldots m\}\cup\{(u_i,l_j,j)\mid i=1\ldots m, j\in S_i\}$. That is, every element j of the domain $\{1,\ldots,n\}$ is associated to a location l_j, and for every set S_i we introduce a location s_i visited by user u_i at time 0. Furthermore, user u_i visits all locations l_j such that $j\in S_i$ at time stamp j. If we use the absolute model with $\tau=1$ and $\omega\geq n+1$, for $i=1\ldots m$, $\phi(\{s_i\})=\{l_j\mid j\in S_i\}$. As such there exists a set cover of size k if and only if there exists a set of locations S of size k such that $|\phi(S)|=n$. □

Recall that the influence of a set of locations S is computed by accumulating the effect of all locations in S. It is hence possible that two locations s and s' separately do not influence a target location d because individually they have too few bridging visitors to d, but together they reach the threshold. This situation occurs for instance in Figure 1, for the locations H_2 and M_1. These locations individually do not reach the threshold to influence H_1 for $\tau_A=2$ and $\omega=1$. However, together they do. One inconvenient consequence of this observation is that the influence function that we want to optimize is not sub-modular [17]. Indeed, in the example above, adding H_2 to the set $\{M_1\}$ gives a higher additional benefit (1 more influenced location) than adding H_2 to $\{\}$. Therefore, we do not have the usual guarantee on the quality of the greedy algorithm for selecting the top-k.

The main reason that we do not have the guarantee is that the benefit is not gradual; before the threshold is reached it is 0, after the threshold is reached it is 1. This means that a location that has $\tau-1$ bridging visitors to 1000 other locations each, gives the same benefit as a location that does not have any bridging visitors. Clearly, nevertheless, the first location is more likely to lead to a good solution if later on additional locations are selected. Therefore, we would like to incorporate potential future benefits into our objective function. Thus, in order to compute the influence of a location, we consider locations that are influenced as well as those locations that are not yet influenced but have potential to be so in future. To characterize the potential of future benefit in combination with the number of influenced locations, we use the following formula:

$$LI(S)=(1-\alpha)\times|\phi(S)|+(\alpha)\times\sum_{d\in L-S}(\min\{|V_B(S,d)|,\tau\}) \tag{4}$$

In this formula, $\alpha=[0,1]$ represents a trade-off between the number of influenced locations and a reward for potential

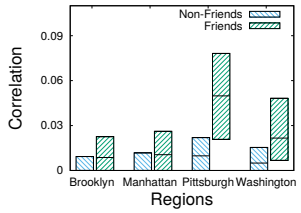


Figure 3: Visit correlations

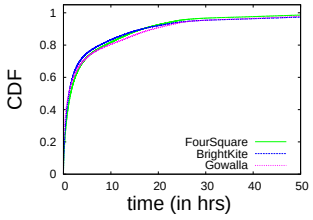


Figure 4: CDF of visit time

influenced locations. For relative models, we replace the $|V_B(S,d)|$ with $|V_B(S,d)|/|V(d)|$.

Next, we apply a greedy method on the basis of location influence to find top-k locations. We start with an empty set S of locations and iteratively add locations to it until we reach the required number of top elements: k. In each step, for each location $s \in L$, we evaluate the effect of adding s to S, and keep the one that gives the highest benefit $LI(S)$. Then, we update $S \leftarrow S \cup \{l\}$.

Example 7. Consider the case in Figure 2 for $\omega = 1$, $\varphi(H_2) = \{(H_1,\{d\}),(M_1,\{i\})\}$, $\varphi(M_1) = \{(H_1,\{i\})\}$ and $\varphi(H_1) = \{(M_1,\{d\})\}$. We aim to find top-2 locations in this example with $\alpha = 0.1$ and $\tau = 2$. During the first iteration, $LI(H_2) = 0.9 \times 0 + 0.1 \times (1+1) = 0.2$, because H_2 does not completely influence any other location, however H_1 and M_1 are potential influenced locations for the bridging visitors d and i, respectively. Similarly, $LI(M_1) = 0.1$ and $LI(H_1) = 0.1$. Thus, we choose H_2 as first seed as it has maximum value. In the next iteration, we first combine the seed H_2 with M_1 and compute the combined influence. Here, $LI(\{H_2,M_1\}) = 0.9 \times 1 + 0.1 \times (2) = 1.1$. Similarly, $LI(\{H_2,H_1\}) = 1.1$. Since, M_1 and H_1 provide equal benefit of 0.9, when combined with H_2, thus we can randomly choose either M_1 or H_1 as a second seed.

5. LBSN DATA ANALYSIS

When constructing the friendship-based influence model the assumption was made that friends tend to follow friends. Furthermore, the influence models of Section 3.3 have several parameters to set: τ and ω. Before going to the experiments, first in this section we verify and confirm the friendship assumption and show how to set the thresholds with reasonable values based on an analysis of the LBSN datasets given in Table 1.

5.1 Mobility analysis of friends

In real life, usually activities of friends are more similar than activities of non-friends. In LBSNs, this implies that a visit of a user to a location increases the chances of visits of his/her friends to the same location. We considered this assumption when constructing our friendship-based influence model in Section 3.3. We illustrate the correctness of this assumption by computing the correlations between activities of users, their friends, and non-friends: Let L_u and L_v be the locations visited by users u and v, respectively. The correlation between activities of u and v is measured by the Jaccard Index [3] between L_u and L_v. The average correlation of activities of users and those of their friends is denoted *friendship correlation* (p^f_{corr}), and the average correlation between activities of users and their non-friends is denoted *Non-friendship Correlation* (p^{nf}_{corr}). In order to avoid an unreasonable bias due to the fact that friends tend to live in the same city, we restrict our computation of the average non-friendship correlation to users in the same city. We randomly picked four regions of the United States, i.e., Brooklyn, Manhattan, Pittsburgh, and Washington and consider the activities of users in these regions to study the correlations. The statistics of p^f_{corr} and p^{nf}_{corr} of all the users are given in Figure 3. The figure presents boxplots without outliers. It can be seen that median of p^f_{corr}, even though still small, is up to 5 times larger than p^{nf}_{corr}. The same pattern is observed for all the datasets, thus only results for `Gowalla` are shown due to space constraints. This validates the claim that the activities of friends are more similar than non-friends.

	Users	Locations	Check-ins	POIs
`FourSquare`	16K	803K	1.928M	582K
`BrightKite`	50K	771K	4.686M	631K
`Gowalla`	99.5K	1.257M	6.271M	1.162M

Table 1: Statistics of datasets

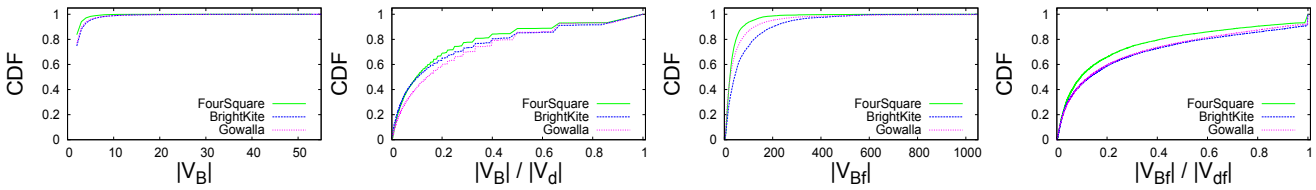
5.2 Setting ω and τ

In order to determine the value of influence window threshold ω, we measure the time difference between consecutive visits of users to distinct locations. The cumulative distribution functions (CDF) for three LBSNs are given in Figure 4. It can be seen that for all LBSNs in our study, 80% of the consecutive activities are performed within 8 hours. After that, there is only a moderate increase in the number of activities with respect to the time interval. Thus, in order to capture only the most common activities, we keep $\omega = 8$. However, it can, of course, be changed if the data distribution is different, or there are different user or application requirements.

We furthermore compute the absolute and relative number of bridging visitors. In order to do that, we consider both the models with-friends and without-friends, for each pair of locations with at least one bridging visitor. The cumulative distribution functions for each of these numbers are depicted in Figure 5. We can utilize the CDF values for controlling the number of influences in the dataset, and thus also for finding the suitable values of thresholds for models. The values of thresholds are an application dependent choice and can be considered accordingly. For example, if an application requires to find many influential relationships, and indirectly many influential and influenced locations, then a lower threshold should be considered and vice versa. In this paper, we consider the top 20% influential relationships among locations for all the models. Thus, the thresholds for all the models are their corresponding CDF values of 0.8 (100%-20%=80%). Therefore, the values of $\tau_A, \tau_R, \tau_{Af}$ and τ_{Rf} are $2, 0.4, 120$ and 0.6, as shown in Figures 5a, 5c, 5b, and 5d, respectively.

6. EVALUATION

We conducted our experiments on a Linux machine with Intel Core i5-4590 CPU @3.33GHz CPU and 16 GB of RAM, running the Ubuntu 14 operating system. We implemented the exact and the approximate algorithms in C++.



(a) Absolute without-friends (b) Relative without-friends (c) Absolute with-friends (d) Relative with-friends

Figure 5: Cumulative distribution function (CDF) of thresholds for all influence propagation models

			No. of Buckets (b)		
			64	128	256
Rel. error	Abs.	mean $\pm\sigma$	0.02 ± 0.15	0.01 ± 0.1	0.01 ± 0.08
	Abs. friends	mean $\pm\sigma$	0.167 ± 0.63	0.08 ± 0.45	0.04 ± 0.49
	Rel.	mean $\pm\sigma$	0.06 ± 0.23	0.06 ± 0.23	0.06 ± 0.23
	Rel. friends	mean $\pm\sigma$	0.05 ± 0.21	0.05 ± 0.21	0.05 ± 0.2
Time	with-out friends	Exact		38.7	
		Approx	40	37.5	42.9
	with friends	Exact		389.6	
		Approx	61.9	67.1	70.9
Memory	with-out friends	Exact		505	
		Approx	531	644	835
	with friends	Exact		3790	
		Approx	541	658	855

Table 2: Exact vs Approx algorithm comparison for accuracy (relative error), time (sec) and memory (MB)

Datasets. We used 3 real-world datasets : `FourSquare` [10], `BrightKite`, and `Gowalla` [6]. These datasets each consisted of two parts: the friendship graph and an ordered list of check-ins. A check-in record contains the user-id, check-in time, GPS coordinates of location, and a location-id. The statistics of the datasets are given in Table 1.

Data Prepossessing. The real-life datasets required preprocessing because many locations are associated with multiple location identifiers with slightly different GPS coordinates. Consider, for instance, Figure 6. In this figure, 13 GPS coordinates that appear in the `FourSquare` dataset are shown which corresponds to different locations Ids in the dataset, but which clearly belong to one unique location. In order to resolve this issue, we clustered GPS points to get POIs. We used the density-based spatial clustering algorithm [16] with parameters *eps*=10 meters and *minpts*=1 to group the GPS points. New location Ids are assigned to each cluster which were used in all our experiments. All 3 datasets have similar problems. The statistics of the new Ids are reported in column POIs of Table 1.

Figure 6: GPS coordinate of 13 location-ids on GoogleMaps

6.1 Approximate vs. Exact Oracle

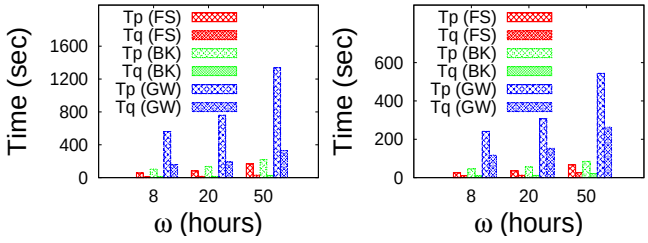
We analyzed the accuracy of the influence approximation based on the HLL sketch. We also analyzed memory consumption and computation time improvement for the approximate approach. The results are similar for all the datasets and hence we only present results for `BrightKite` due to space constraints.

Approximation Accuracy. For every location with a non-empty influence set, we used the HLL-based approximate version of the Oracle to predict the size of the influence set. Then the relative error as compared to the real size was computed for every location. In Table 2 the mean and standard deviation of this relative approximation error over all locations with a non-empty influence are given. The experiments are performed for both with-friends and without-friends for the absolute influence model and relative influence model. We ran the experiments for different numbers of buckets (b) for the *HLL* sketch, being, $64, 128$ and 256. As can be seen in the table, the errors are unbiased (0 on average), and the standard deviation decreases as the number of buckets increases. The error is a bit higher in the relative model as compared to the absolute model because in the relative model the influence is computed by taking the ratio of two approximated sets. Values for b beyond 256 yielded only modest further improvements and hence we used $b = 256$ in all further experiments.

Approximation Efficiency. Next, we compare the computation time and memory requirements for the approximate approach with that of the exact approach. In order to do so, we computed influence sets with friends and without friends. The computation times and memory consumption are shown in Table 2. The approximate approach outperforms the exact approach up to a factor 6 in time using only 15% of memory for the models including friends. Due to the sparsity of data, however, the gain for the without-friend case is negligible. This is because the sizes of the sets of bridging visitors are very modest and hence there is no need to reduce memory consumption. It can be observed that time and memory of the approximate approach increase with increasing number of buckets b.

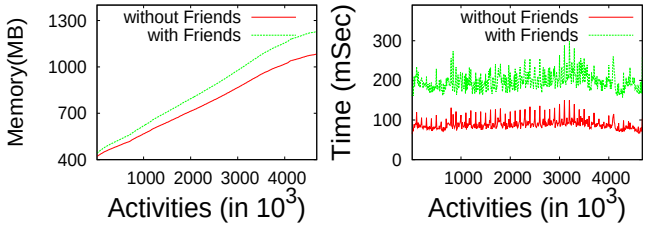
6.2 Influence of ω and τ

Runtime. We study the runtime of the approximate algorithm on all the datasets for different values of $\omega := 8, 20$ and 50. The average runtime for processing all the activities (T_p) under the models varies only depending on whether or not we consider friends; it does not depend on τ. The oracle query time (T_q) is independent of τ and model. Hence we only show results for $\tau = 2$. The run times are shown in



(a) with-Friends (b) without-Friends

Figure 7: Time to process all activities (Tp) and query Oracle (Tq) for $\tau = 2$ at different ω



(a) Memory (b) Time

Figure 9: Performance evaluation for processing 1000 activities for $\omega = 8$

Figure 7 for the three datasets `FourSquare`, `BrightKite` and `Gowalla`. The running time increases with increasing influence window size ω as more locations from the visit history remain active. Running time is higher in the with-friends case which is not surprising either as the number of users to include in the bridging visitors sets increases due to the addition of friends. The time taken to process dataset `Gowalla` is the highest as it has the largest number of locations.

In Figure 9b, we report the time taken in function of the number of activities for $\omega = 8$. Per $1,000$ activities in the `BrightKite` dataset the runtime is reported. As can be seen in the figure, the average time taken per $1,000$ activities remains constant. The time taken for the friendship-based influence model is the highest as more users are merged.

Memory Consumption. We also study the memory required by the approximation algorithm on all the datasets for different values of $\omega := 8, 20$ and 50. Unlike for the processing time, the average memory required to process all the activities under the models does not vary based on whether we consider friends or not. This is because the HLL sketch storing the bridging visitor set size remains constant in size even if a larger number of users is added to it. The memory requirement increases slightly with ω as more locations are getting influenced due to a larger influence window. The results are shown in Figure 8. In Figure 9a, we report the memory used as a function of the number of activities for $\omega = 8$. Per $1,000$ activities in the `BrightKite` dataset the runtime is reported. The total memory requirements increase linearly with time as new locations come in over time for which a complete influence summary needs to be maintained. In Figure 10 on the other hand, we see that over time the size of user visit history remains constant due to the pruning of outdated locations in the visit histories.

6.3 Influence Maximization

Influence of α. Our next goal is to study how the influence maximization algorithm performs for different values of α. In order to avoid data sparsity issues, we filter out those locations which have only one visitor from all the datasets. We tested the spread of top 200 locations obtained by considering values of α from 0.01 to 0.99. We observed that the number of bridging visitors per location is highly skewed as can be learn from Figure 5a. Due to this, the potential influenced locations having few bridging visitors are less likely to affect the influenced set of the locations. The effect of varying alpha on the influence spread is shown in Figure 11. As expected for these sparse datasets, our algorithms perform best with a lower value of α. We use $\alpha = 0.03$ for our experiments.

τ	Time (sec)		
	$k = 10$	$k = 20$	$k = 50$
$\tau_A = 2$	2	3	35
$\tau_R = 0.4$	5	6	46
$\tau_{Af} = 120$	2	5	46
$\tau_{Rf} = 0.6$	4	6	53

Table 3: Time taken to find top k locations (`BrightKite`)

Computation time. We study the computation time for finding top-k influential locations under both the with-friends and the without-friends influence models. The runtime is close in the both absolute and relative models. The

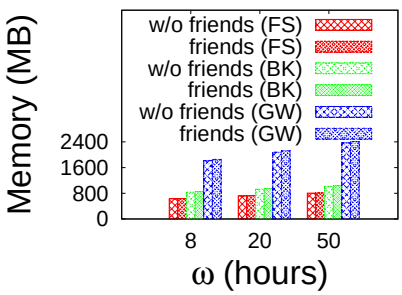


Figure 8: Memory to process all activities at different ω

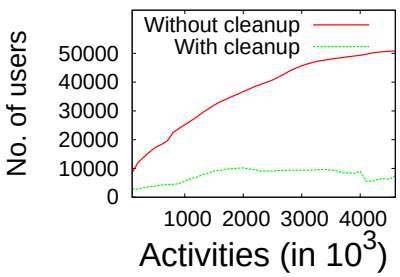


Figure 10: User visit history growth w.r.t. cleanup process

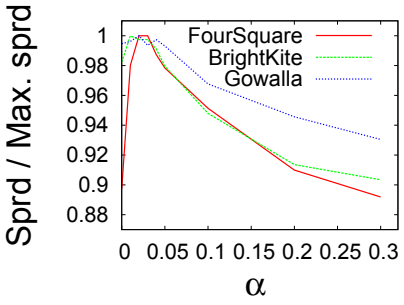


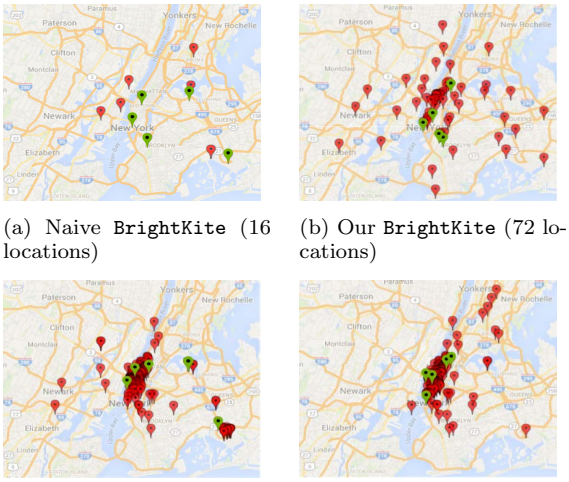
Figure 11: Influence spread w.r.t. alpha (200 seeds)

time increases with k. Nevertheless, the increase is modest; for instance, finding the top-50 locations takes less than a minute. We report the results in Table 3.

6.4 Qualitative Experiment

In order to demonstrate our model of location influence, we compared the results of our method with a naive approach for selecting top-k locations. In the naive approach, we selected the top k locations such that the number of distinct users visiting those locations is maximized. This result is compared to the top-k most influential locations found using the absolute influence model with $\tau = 1$. We compared the influence spread by the top-k locations of both approaches.

We considered the activities performed in the area of New York in all the three data-sets and fetched top-5 locations for $\omega = 8$ *hours* for both approaches. We further computed the influence spread for the selected locations of both approaches using the absolute influence model. Top-5 locations with their influenced locations are plotted using Google Maps as shown in Figure 12 for `FourSquare` and `BrightKite`. In the figure, it can be observed that for `BrightKite` our method leads to a set of locations with a much larger spread as compared to the naive approach, both geographically and in terms of the number of locations influenced. On the other hand, the spread for both approaches for `FourSquare` is similar. The reason is that for this dataset the problem of selecting the top locations is almost trivial as there is only a small set of locations visited multiple times with as a result that once this limited set of locations is selected, it does not matter which other users are selected.



(a) Naive `BrightKite` (16 locations)

(b) Our `BrightKite` (72 locations)

(c) Naive `FourSquare` (239 locations)

(d) Our `FourSquare` (239 locations)

Figure 12: Comparison of top- 5 influential locations (green) and their spread (red) between naive and our approach

7. CONCLUSION AND FUTURE WORK

In this paper, we introduced a notion that can be used to optimize outdoor marketing strategies such as finding optimal locations for advertising products to maximize the geographical spread. In order to do that, we captured the interactions of locations on the basis of their visitors to compute the influence of locations among each other. We provided two models namely the absolute influence model and the relative influence model. We further incorporated friends of users in order to deal with data sparsity. We proposed an Oracle data structure to efficiently compute the influence of locations on the basis of these models. Oracle can be used for different applications such as finding top-k influential locations. In order to maintain this data structure, we first provided a set-based exact algorithm. Then, we optimized the time and memory requirements of the algorithm up to 6 times and 7 times, respectively, by utilizing a probabilistic data structure. Finally, we provided a greedy algorithm to compute the top-k influential locations. In order to evaluate the methods, we utilized three real datasets. We first analyzed the LBSN datasets: `FourSquare`, `BrightKite` and `Gowalla` to verify some claims and to provide optimal values for thresholds of the influence models. Then, we evaluated our approaches for the computation of the Oracle and finding top-k locations in terms of accuracy, computation time, memory requirement and scalability. We further show the effectiveness of our proposed models by comparing the influence spread of top-k locations fetched by our approach with that of a naive approach.

In the future, we plan to enrich location influence models by incorporating the activities users perform with their friends in groups. Moreover, we aim to provide distributed mechanisms for computing the Oracle data structures and influences for the models.

Acknowledgments

This research has been funded in part by the European Commission through the Erasmus Mundus Joint Doctorate "Information Technologies for Business Intelligence - Doctoral College" (IT4BI-DC). Xike Xie is supported by CAS Pioneer Hundred Talents Program.

8. REFERENCES

[1] A. AlDwyish, E. Tanin, and S. Karunasekera. Location-based social networking for obtaining personalised driving advice. In *SIGSPATIAL*, 2015.

[2] P. Bouros, D. Sacharidis, and N. Bikakis. Regionally influential users in location-aware social networks. In *SIGSPATIAL*, 2014.

[3] R. R. Braam, H. F. Moed, and A. F. Van Raan. Mapping of science: Critical elaboration and new approaches, a case study in agricultural biochemistry. In *Informetrics*, 1988.

[4] W. Chen, Y. Wang, and S. Yang. Efficient influence maximization in social networks. In *KDD*, 2009.

[5] W. Chen, Y. Yuan, and L. Zhang. Scalable influence maximization in social networks under the linear threshold model. In *ICDM*, 2010.

[6] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: User movement in location-based social networks. In *KDD*, 2011.

[7] T.-N. Doan, F. C. T. Chua, and E.-P. Lim. Mining business competitiveness from user visitation data. In *SBP*, 2015.

[8] L. Ferrari, A. Rosi, M. Mamei, and F. Zambonelli. Extracting urban patterns from location-based social networks. In *SIGSPATIAL*, 2011.

[9] P. Flajolet, É. Fusy, O. Gandouet, and F. Meunier. Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. In *DMTCS*, 2008.

[10] H. Gao, J. Tang, and H. Liu. Exploring social-historical ties on location-based social networks. In *AAAI*, 2012.

[11] A. Goyal, F. Bonchi, and L. V. Lakshmanan. Learning influence probabilities in social networks. In *WSDM*, 2010.

[12] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan. A data-based approach to social influence maximization. In *PVLDB*, 2011.

[13] N. T. Hai. A novel approach for location promotion on location-based social networks. In *RIVF*, 2015.

[14] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *KDD*, 2003.

[15] G. Li, S. Chen, J. Feng, K.-l. Tan, and W.-s. Li. Efficient location-aware influence maximization. In *SIGMOD*, 2014.

[16] Q. Liu, M. Deng, Y. Shi, and J. Wang. A density-based spatial clustering algorithm considering both spatial proximity and attribute similarity. In *Computers and Geosciences*, 2012.

[17] L. LováSz. Review of the book by alexander schrijver: Combinatorial optimization: Polyhedra and efficiency. In *Oper. Res. Lett.*, 2005.

[18] F. J. Mata and A. Quesada. Web 2.0, social networks and e-commerce as marketing tools. In *J. Theor. Appl. Electron. Commer. Res.*, 2014.

[19] M. A. Saleem, X. Xie, and T. B. Pedersen. Scalable processing of location-based social networking queries. In *MDM*, 2016.

[20] X. Wang, Y. Zhang, W. Zhang, and X. Lin. Distance-aware influence maximization in geo-social network. In *ICDE*, 2016.

[21] Y.-T. Wen, P.-R. Lei, W.-C. Peng, and X.-F. Zhou. Exploring social influence on location-based social networks. In *ICDM*, 2014.

[22] H.-H. Wu and M.-Y. Yeh. Influential nodes in a one-wave diffusion model for location-based social networks. In *PAKDD*, 2013.

[23] C. Zhang, L. Shou, K. Chen, G. Chen, and Y. Bei. Evaluating geo-social influence in location-based social networks. In *CIKM*, 2012.

[24] T. Zhou, J. Cao, B. Liu, S. Xu, Z. Zhu, and J. Luo. Location-based influence maximization in social networks. In *CIKM*, 2015.

[25] W.-Y. Zhu, W.-C. Peng, L.-J. Chen, K. Zheng, and X. Zhou. Modeling user mobility for location promotion in location-based social networks. In *KDD*, 2015.