Efficient Similarity-aware Influence Maximization in Geo-social Network

Xuanhao Chen, Yan Zhao, Guanfeng Liu, Rui Sun, Xiaofang Zhou, *Fellow, IEEE*, and Kai Zheng*, *Member, IEEE*

Abstract—With the explosion of GPS-enabled smartphones and social media platforms, geo-social networks are increasing as tools for businesses to promote their products or services. Influence maximization, which aims to maximize the expected spread of influence in the networks, has drawn increasing attention. However, most recent work tries to study influence maximization by only considering geographic distance, while ignoring the influence of users' spatio-temporal behavior on information propagation or location promotion, which can often lead to poor results. To relieve this problem, we propose a Similarity-aware Influence Maximization (SIM) model to efficiently maximize the influence spread by taking the effect of users' spatio-temporal behavior into account, which is more reasonable to describe the real information propagation. We first calculate the similarity between users according to their historical check-ins, and then we propose a Propagation to Consumption (PTC) model to capture both online and offline behaviors of users. Finally, we propose two greedy algorithms to efficiently maximize the influence spread. The extensive experiments over real datasets demonstrate the efficiency and effectiveness of the proposed algorithms.

Index Terms—Geo-social networks, Influence maximization, Similarity-aware.

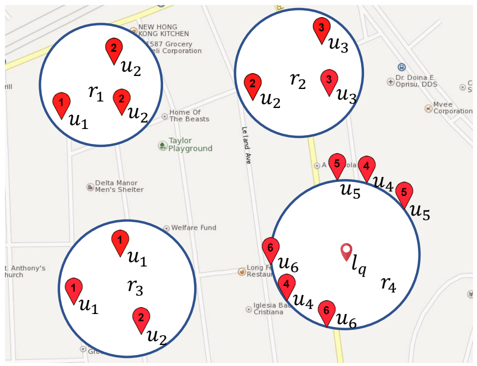
1 Introduction

With the rapid development of Internet technology, social networks such as Twitter and Microblog have served as important platforms for people to obtain and share information. Influence maximization, which leverages the benefit of word-of-mouth effect in these social networks, is a key problem in viral markerting and has received tremendous attention in the last years [1], [2], [3], [4], [5], [6], [7], [8], [9]. Influence maximization problem is defined as search for a k-size subset of users (usually called seeds) with maximum influence spread in social networks, where k is a positive integer and can be specified in advance.

With the ubiquity of GPS-equipped smart devices, users are now willing to share their geo-location information in social networks (called geo-social networks), allowing for the study of the role of geographic information in social ties. Influence maximization problem in geo-social networks needs to take location information into account [10], [11], [12], [13]. Users who have higher probability visiting the query location contribute more to the influence spread of seeds [14], and usually a power-law function is employed to measure the probability of users visiting the query location based on the geometric distance [15]. In most existing work studying the influence maximization problem in geo-social networks [14], [15], only the query location is taken into account, while the importance of users' spatio-temporal behavior is neglected, which will lead to unsatisfying results in some situations. For example, in Figure 1 (the timestamp means the check-in time, i.e., year-month-day T hour Z, and ROI (region of interest) means the check-in region which is the daily activity area of users), there is a new restaurant at location l_q, and the owner plans to offer a free meal coupon to user who is near the restaurant and most influential to propagate the news about her restaurant through social networks. There are six people who are friends in an online social network. Since daily activity areas of u_1, u_2 and u_3 are far away from the restaurant, the probability of them visiting the restaurant is low (it is less likely for users to visit the locations that is far away from their daily activity areas [16]). It is clear that u_4, u_5 and u_6 are candidates. Since the distance between these users' check-in locations and query location l_q is almost the same, if we only consider the influence of distance [14], the probability of these users visiting l_q is the same, and these users have the same influence spread based on the equation of influence spread calculation. As a result, the seeds selecting algorithms fail to select a suitable user as the seed. In fact, u_4 shows more similar spatio-temporal behavior with u_5 and u_6 (i.e., the probability that u_4 activates both u_5 and u_6 is larger). By taking spatio-temporal behavior into account, we can choose u_4 as the seed to get the largest influence spread.

To overcome the limitation, in the this paper, we propose a Similarity-aware Influence Maximization (SIM) problem, which aims to maximize the influence spread by efficiently measuring the influence of users' spatio-temporal behaviors. Specifically, we cluster the check-in locations to compute the spatial similarity and design a method to measure the temporal similarity. Subsequently, a Propagation to Consumption (PTC) model is proposed to support the SIM

- *X. Chen, R. Sun and K. Zheng are with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China. Email: {xhc, sunrui}@std.uestc.edu.cn, zhengkai@uestc.edu.cn. *K. Zheng is the corresponding author of the paper.*
- *Y. Zhao is with the Department of Computer Science, Aalborg University, Aalborg DK-9220, Denmark. Email: yanz@cs.aau.dk.*
- *G. Liu is with the Department of Computing, Macquarie University, Sydney NSW2109, NSW Australia. Email: guanfeng.liu@mq.edu.au.*
- *X. Zhou is with the University of Queensland, Brisbane, Australia and Zhejiang Lab, China. Email: zxf@itee.uq.edu.au.*



User	Timestamp	ROI
u_1	2010-10-17T13Z	r_1
	2010-10-06T08Z	r_3
	2010-10-06T19Z	
u_2	2010-10-18T10Z	r_1
	2010-10-18T15Z	
	2010-10-13T09Z	r_2
	2010-10-06T20Z	r_3
u_3	2010-10-19T19Z	r_2
	2010-10-21T20Z	
u_4	2010-10-13T10Z	r_4
	2010-10-15T11Z	
u_5	2010-10-13T09Z	r_4
	2010-10-25T15Z	
u_6	2010-10-02T16Z	r_4
	2010-10-15T12Z	

Fig. 1. The check-ins of users

problem. Our solution seamlessly combines two factors: the similarity between users and users' online and offline behaviors. Given a geo-social network G and the historical check-ins of each user, each edge between two users is assigned a probability to measure the similarity between users. In order to make the PTC model more realistic, we divide the propagation process into two parts, i.e., online part and offline part. Users will get information from their friends online first and then take offline experience to decide whether to visit the query location or not according to their location preference. Moreover, we design a *Neighbor Contribution (NC)* parameter to prune the insignificant nodes and build Influence Propagation Trees (IPT) to efficiently calculate the influence spread of nodes. Finally, we develop two algorithms, i.e., Influence Propagation Trees Based (IPTB) algorithm and Cutting Tails (CT) algorithm, to maximize the influence spread. In IPTB algorithm, we not only propose an efficient strategy to prune insignificant nodes but also a method to calculate the upper bound influence of users. In CT algorithm, we prune insignificant nodes based on *NC* and employ submodular property [1] to find the most influential nodes.

In summary, our major contributions can be outlined as follows:

- We formulate a similarity-aware influence maximization problem on geo-social networks, where we take an important step toward efficient influence maximization calculation by considering both user similarity and users' online and offline behaviors.
- To efficiently compute the influence spread of users, we design the influence propagation trees, by which we can efficiently compute the upper bound influence spread of each node.
- We propose several algorithms to efficiently maximize the influence spread.
- We conduct comprehensive experiments on the real datasets. The experimental results confirm the efficiency and effectiveness of the proposed algorithms.

The rest part of this paper is organized as follows: Section 2 provides the notations and the proposed problem, along with the methods used to calculate the spatio-temporal similarity, and the related models are shown in Section 3. In Section 4, we show the method used to calculate the influence spread of users and propose two greedy algorithms to efficiently select the most influential nodes. The experimental results are shown in Section 5. Section 6 introduces the related work. Finally, we conclude the paper in Section 7.

TABLE 1
Summary of Notations

Notation	Definition
$G = \langle U, E \rangle$	Geo-social network
U	The set of users/nodes
E	Social connections between users/nodes
$E(u_i, u_j)$	The edge between u_i and u_j
S	A selected seed set $S \subseteq U$
C	The set of check-in records
u_i	The i-th user/node in U
l_q	The query location
$P_{st}(u_i, u_j)$	Online propagation probability based on similarity
$P_{l_q}(u_i)$	The probability that u_i heads for l_q
δ	The proportion of candidates
θ_{off}	Offline threshold for pruning insignificant users

2 Preliminaries

2.1 Problem Definition

In this part, we will formally introduce the problem of similarity-aware influence maximization in a geo-social network, which is also called Location-Based Social Network (LBSN). In the rest of the paper, we will use geo-social network and LBSN interchangeably. Table 1 lists the major notations used throughout the paper.

Definition 1. (LBSN) An LBSN $\langle G, C \rangle$ consists of a social network $G = \langle U, E \rangle$ (where U is the set of users, E is the social connections between users and $E(u_i, u_j)$ is used to denote the connection between u_i and u_j), and the set of check-in records $C = \{(c_{u_i,l,t})\}$. $c_{u_i,l,t}$ represents the record where user u_i checks in at location l at time t. A location l is a coordinate which consists of latitude and longitude.

Definition 2. (Region of Interest, ROI) Given the historical check-ins, we cluster the check-ins locations. Each cluster is the region of interest.

Definition 3. (Query location) The query location, l_q, is a location that users need to visit.

Definition 4. (Spatial Similarity) The spatial similarity between users u_i and u_j, denoted as $sim_s(u_i, u_j)$, is a metric used to measure how similar of users' spatial behavior is.

Definition 5. (Temporal Similarity) The temporal similarity between users u_i and u_j, denoted as $sim_t(u_i, u_j)$, is a metric used to measure how similar of users' temporal behavior is.

Definition 6. (Spatio-temporal Similarity) The spatio-temporal similarity between users u_i and u_j, denoted as $sim_{st}(u_i, u_j)$, is the incorporation of spatial similarity and temporal similarity.

Definition 7. (Similarity-based online information propagation probability) The online information propagation probability based on spatio-temporal similarity between users u_i and u_j, denoted as $P_{st}(u_i, u_j)$, is a metric used to measure how likely u_j can be influenced by u_i online.

Definition 8. (Offline probability) The offline propagation probability of user u_i, denoted as $P_{l_q}(u_i)$, is a metric used to measure the probability that u_i visits the query location l_q.

Definition 9. (Seeds set) The seeds set, denoted as S, is a k-size subset of users with maximum influence spread in the networks.

Definition 10. (Proportion of Candidates) The proportion of candidates (users who are likely to be selected as seeds), denoted as δ, is a metric used to decide how many nodes are considered to participate in the seeds selection process.

Problem Statement. Given an LBSN network $\langle G, C\rangle$, a query location l_q and an integer k, the problem of Similarity-aware Influence Maximization (SIM) is to find a set S^* of k nodes in G, which maximizes the number of expected influenced users who will visit the query location l_q, i.e., $S^* = \arg\max_{S\subseteq U}\{\sigma(S)||S| = k\}$.

2.2 Spatio-temporal Similarity Calculation

In this part, we will show the details of calculating spatio-temporal similarity.

2.2.1 Spatial Similarity Calculation

Since the activity of users is usually in a specific region, we measure the spatial similarity based on ROI. First, we cluster the check-in locations by utilizing DBSCAN [17]. The DBSCAN requires two parameters, ϵ and *MinPts*, which denote the distance threshold and the minimum number of points that are no farther than ϵ from the given point, respectively. By changing ϵ, the size of the cluster can be adjusted. Therefore, we can adjust ϵ and *MinPts* to hierarchically cluster the users' check-ins.

After clustering the check-ins, we can get the set of ROI, i.e., $RS = \{r_1, r_2, \cdots, r_m\}$, where r_i is the i-th ROI. Given the RS and the historical check-in records of user u_i, the vector of spatial behavior of u_i can be represented as $V_{u_i} = \begin{bmatrix} v_{u_i,1} & v_{u_i,2} & \cdots & v_{u_i,m}\end{bmatrix}$, where m is the size of the RS and $v_{u_i,j}$ is the total number that user u_i checks in at one of the locations in r_j. According to the V_{u_i}, we can get the check-ins matrix $M_{i'}$ of all users in the i'-th level:

$$M_{i'} = \begin{bmatrix} v_{u_1,1} & v_{u_1,2} & \cdots & v_{u_1,m} \\ v_{u_2,1} & v_{u_2,2} & \cdots & v_{u_2,m} \\ \vdots & \vdots & & \vdots \\ v_{u_n,1} & v_{u_n,2} & \cdots & v_{u_n,m} \end{bmatrix} \tag{1}$$

where n is the number of users and m is the size of the RS by the given ϵ and *MinPts*. By Equation 1, we can compute the spatial similarity of users at i'-th level, $sim_s^{i'}(u_i, u_j)$, as follows:

$$sim_s^{i'}(u_i, u_j) = \frac{M_{i'}(u_i) \cdot M_{i'}(u_j)}{\|M_{i'}(u_i)\| \|M_{i'}(u_j)\|} \tag{2}$$

Due to the hierarchy of different ϵ, we need to count the spatial similarity after we calculate it in different levels. So the spatial similarity, $sim_s(u_i, u_j)$, between user u_i and user u_j can be computed as:

$$sim_s(u_i, u_j) = \sum_{i'=1}^{I} \alpha sim_s^{i'}(u_i, u_j) \tag{3}$$

where $\alpha = \frac{\beta_{i'}}{\sum_{i'=1}^{T}\beta_{i'}}$, I is the total levels and $\beta_{i'}$ is the weight of the i'-th level. The smaller ϵ leads to a larger $\beta_{i'}$.

2.2.2 Temporal Similarity Calculation

Since the temporal behavior of users is different at different time, we split a day into multiple equal time slots based on hours. In this case, $c_{u_i,l,t}$ is the total number that u_i checks in at location l at time slot t. In the rest of the paper, time and time slot are used interchangeable. Given the historical check-in records C, the temporal similarity between users is based on their temporal behaviors over time [18]. We extend the cosine similarity to measure the temporal similarity between u_i and u_j as follows:

$$sim_t(u_i, u_j) = \frac{\sum_{t=1}^{T}\sum_{l=1}^{L} c_{u_i,l,t} \cdot c_{u_j,l,t}}{\sqrt{\sum_{t=1}^{T}\sum_{l=1}^{L} c_{u_i,l,t}^2}\sqrt{\sum_{t=1}^{T}\sum_{l=1}^{L} c_{u_j,l,t}^2}} \tag{4}$$

where T is the number of time slots, L is the number of Points of Interest (POI) in the dataset.

In Equation 4, it is clear that the daily temporal behavior of users is regarded as the same, while the recent work shows that the temporal pattern of a POI's popularity changes largely between weekdays and weekends [19]. So we further divide the check-in records into weekdays pattern $\mathbb{P}_y$ and weekends pattern $\mathbb{P}_d$. Accordingly, $c_{u_i,l,t}^{\mathbb{P}}$ is used to represent that user u_i check in at location l at time t with respect to weekdays or weekends, where $\mathbb{P} \in \{\mathbb{P}_y, \mathbb{P}_d\}$. Moreover, we use $c_{u_i,t}^{\mathbb{P}}$ to represent the temporal check-in vector of user u_i at time t with respect to $\mathbb{P}$, where $c_{u_i,t}^{\mathbb{P}} = \begin{bmatrix} c_{u_i,1,t}^{\mathbb{P}} & c_{u_i,2,t}^{\mathbb{P}} & \cdots & c_{u_i,L,t}^{\mathbb{P}}\end{bmatrix}$.

However, the dividing approach makes the data sparser, which will lead to wrong results when characterizing the temporal similarity between users in some scenarios. For example, assuming two users u_1 and u_2, u_1 checks in at l_1 and l_2 at time t_1 and t_2, respectively, and u_2 checks in at l_2 and l_1 at time t_1 and t_2, respectively. If we directly use Equation 4 to compute the temporal similarity between u_1 and u_2, we will get $sim_t(u_1, u_2) = 0$. Obviously, the temporal similarity is not desirable in this case, especially when t_1 is very close to t_2. To address the data sparsity problem, we utilize the temporal check-in vectors at other time slots to recompute $c_{u_i,t}^{\mathbb{P}}$. Specifically, for each user u_i, we calculate the cosine similarity between each temporal check-in vectors $c_{u_i,t_i}^{\mathbb{P}}$ and $c_{u_i,t_j}^{\mathbb{P}}$ at time t_i and t_j, respectively, and then the value of the similarity between two time slots t_i and t_j, $s_{t_i,t_j}^{\mathbb{P}}$, can be evaluated by the average similarity of all users between t_i and t_j. Note that the check-in behavior of users is different between weekdays and weekends, so $s_{t_i,t_j}^{\mathbb{P}}$ should be computed separately based on weekdays pattern, $\mathbb{P}_{u_i,t}^{y}$, and weekends pattern $\mathbb{P}_{u_i,t}^{d}$.

The temporal behaviors of u_i can be recomputed as follows:

$$\hat{c}_{u_i,l,t}^{\mathbb{P}} = \sum_{t_i=1}^{T} \frac{s_{t,t_i}^{\mathbb{P}}}{\sum_{t_j=1}^{T} s_{t,t_j}^{\mathbb{P}}} c_{u_i,l,t_i}^{\mathbb{P}} \tag{5}$$

where $s^{\mathbb{P}}_{t,t_i}$ is the similarity between time slots t and t_i, which can be computed as follows:

$$s^{\mathbb{P}}_{t,t_i} = \frac{1}{n}\sum_{j}^{n} \frac{c^{\mathbb{P}}_{u_j,t} \cdot c^{\mathbb{P}}_{u_j,t_i}}{\|c^{\mathbb{P}}_{u_j,t}\|\|c^{\mathbb{P}}_{u_j,t_i}\|} \quad (6)$$

where n is the number of users. According to Equations 4, 5 and 6, we can calculate the temporal similarity between any users with respect to weekdays pattern and weekends pattern as follows:

$$sim^{\mathbb{P}}_t(u_i,u_j) = \frac{\sum_{t=1}^{T}\sum_{l=1}^{L} \hat{c}^{\mathbb{P}}_{u_i,l,t} \cdot \hat{c}^{\mathbb{P}}_{u_j,l,t}}{\sqrt{\sum_{t=1}^{T}\sum_{l=1}^{L}(\hat{c}^{\mathbb{P}}_{u_i,l,t})^2}\sqrt{\sum_{t=1}^{T}\sum_{l=1}^{L}(\hat{c}^{\mathbb{P}}_{u_j,l,t})^2}} \quad (7)$$

After getting the temporal similarity with respect to weekdays pattern and weekends pattern, we can recompute the temporal similarity as follows:

$$sim_t(u_i,u_j) = \gamma sim_t^{\mathbb{P}_y}(u_i,u_j) + (1-\gamma) sim_t^{\mathbb{P}_d}(u_i,u_j) \quad (8)$$

where $0 < \gamma < 1$ produces a weight for temporal similarity to compensate the difference between weekdays and weekends. For example, supposing each day in a week is of the same importance, we can obtain $\gamma = \frac{5}{7}$ for five days of the weekdays.

2.2.3 Spatial and Temporal Similarity Incorporation

Because the spatial similarity and the temporal similarity are calculated in different ways, the obtained results for quantification of similarity might have different ranges. Thus, we first normalize the two similarities by min-max normalization, which are computed as follows:

$$\widetilde{sim}_s(u_i,u_j) = \frac{sim_s(u_i,u_j) - \min(sim_s(u_{i'},u_{j'}))}{\max(sim_s(u_{i'},u_{j'})) - \min(sim_s(u_{i'},u_{j'}))} \quad (9)$$

$$\widetilde{sim}_t(u_i,u_j) = \frac{sim_t(u_i,u_j) - \min(sim_t(u_{i'},u_{j'}))}{\max(sim_t(u_{i'},u_{j'})) - \min(sim_t(u_{i'},u_{j'}))} \quad (10)$$

where $\max(\cdot)$ and $\min(\cdot)$ are the maximum and minimum spatial (temporal) similarity, respectively.

After normalization, we can combine the spatial similarity and temporal similarity reasonably. The spatio-temporal similarity, $sim_{st}(u_i,u_j)$, between user u_i and user u_j, can be calculated as follows:

$$sim_{st}(u_i,u_j) = \lambda\widetilde{sim}_s(u_i,u_j) + (1-\lambda)\widetilde{sim}_t(u_i,u_j) \quad (11)$$

where $0 \le \lambda \le 1$ is a parameter deciding the weight of spatial similarity.

To measure the importance of spatio-temporal similarity on propagation probability, we borrow the edge-weight based compartmental approach used to study the epidemic spreading [20] to calculate the online propagation probability. Assuming the original propagation probability is $P_0(u_i,u_j)$, the similarity-based online information propagation probability, $P_{st}(u_i,u_j)$, between u_i and u_j can be computed as follows:

$$P_{st}(u_i,u_j) = 1 - (1 - P_0(u_i,u_j))^{w^{sim_{st}(u_i,u_j)}} \quad (12)$$

where $w > 1$ is a parameter controlling the weight of spatio-temporal similarity, called enhancement factor. A larger w indicates that the spatio-temporal similarity between users plays a more important role in information propagation.

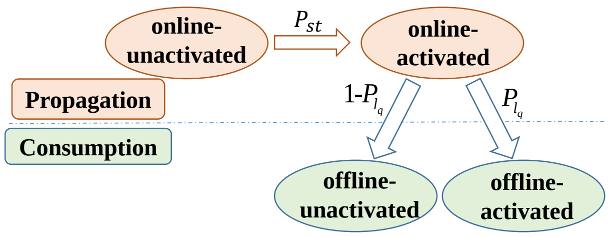


Fig. 2. The propagation model

3 Models

3.1 Propagation to Consumption Model

Since the influence maximization problem in LBSN relies on both information diffusion in online social network and users preference of the target location [14], [21], so in this paper, we propose a Propagation to Consumption (PTC) model to capture both online and offline behaviors of users. As shown in Figure 2, the whole process is divided into two parts (i.e., online part and offline part). Users will get the information from their neighbors online first and then they will take offline experience to decide whether to visit the query location or not. According to the feature of the online and offline parts, users are in one of the following four states:

Definition 11. (Online-unactivated State) In this state, the user has not gotten any information from her online friends.

Definition 12. (Online-activated State) In this state, the user has been activated by her friends online and entered into the offline phase.

Definition 13. (Offline-unactivated State) In the offline phase, if the user does not visit the query location, she will move into the offline-unactivated state. In this state, the user can be activated by her online friends again and transit into online-activated state.

Definition 14. (Offline-activated State) In this state, the user has visited the query location and gets the chance to activate her online friends.

The influence diffusion process of PTC model is as follows:

- At timestamp t_0, only the nodes in seed set, denoted as S_0, are offline-activated and others are online-unactivated.
- Let S_i denote the set of nodes that become offline-activated at timestamp t_i. At timestamp t_{i+1}, each node $u_i \in S_i$ attempts to activate its online-unactivated neighbor u_j with the online probability $P_{st}(u_i,u_j)$ to make u_j transit into online-activated state. If u_i succeeds, u_j will become online-activated and enter into the offline phase. In the offline phase, u_j can transit into offline-activated state with the offline probability $P_{l_q}(u_j)$, otherwise it will switch to offline-unactivated state. No matter whether u_i

succeeds or not, u_i cannot try to influence u_j in the following rounds.

- Once a node becomes offline-activated, it remains offline-activated for subsequence iterations. The procedure terminates when no more nodes can be offline-activated.

In PTC model, the online probably $P_{st}(u_i, u_j)$ is related to the spatio-temporal similarity between users u_i and u_j. Higher similarity leads to larger $P_{st}(u_i, u_j)$ (see Equation 12). The offline probability $P_{l_q}(u_j)$ is related to the query location l_q and the preference of consuming location of user u_j. The farther distance between the query location and user's daily activity area means a lower probability for the user to visit the query location.

Since the influence maximization problem is NP-hard for the Independent Cascade model [1] and that problem is a special case of SIM for the PTC model with all $sim_{st}(u_i, u_j) = 0$ and $P_{l_q}(u_j) = 1$. This leads to the following hardness result.

Lemma 1. The similarity-aware influence maximization problem is NP-hard for the PTC model.

Although to find the optimal solution for SIM under PTC is NP-hard (Lemma 1), we show that the influence function $\sigma(\cdot)$ is monotonic and submodular, which allows a hill-climbing style greedy algorithm (see Algorithm 1) to achieve a $(1 - 1/e)$-approximation to the optimal.

Lemma 2. The influence function $\sigma(\cdot)$ is monotonic and submodular.

Proof 1. Since the PTC model contains both online and offline parts, we reconstruct the original network, G, into a new directed network, G'. If G is an undirected network, for an edge $E(u_i, u_j)$ between user u_i to u_j, we transform $E(u_i, u_j)$ into a directed edge $E'(u_i, u_j)$ from user u_i to u_j to denote that u_i can activate u_j, and add a new directed edge $E'(u_j, u_i)$. It is clear that $E'(u_i, u_j) = E(u_i, u_j)$ if G is already a directed network. Since only the online part is considered in G, to cover the offline behavior, we set the probabilities of $E'(u_i, u_j)$ and $E'(u_j, u_i)$ be $P_{st}(u_i, u_j)P_{l_q}(u_j)$ and $P_{st}(u_j, u_i)P_{l_q}(u_i)$, respectively, denoting that u_j (u_i) first receives the information from online neighbor with probability $P_{st}(u_i, u_j)$ ($P_{st}(u_j, u_i)$), and then chooses to visit target location l_q with probability $P_{l_q}(u_j)$ ($P_{l_q}(u_i)$).
It is clear that the SIM problem under PTC model on the graph G is the same as the influence maximization problem under IC model on the graph G', where $G' = \langle U', E' \rangle$, $|U'| = |U|$ and $|E'| = 2|E|$ ($|E'| = |E|$) for undirected (directed) graphs. Let $S, T \subseteq U'$, $S \subseteq T$ and $u_j \in U' \setminus T$. $\tilde{G}$ is a set of instances of G', i.e., $\{g_i\}$, and $P(g_i)$ denotes the probability of generating g_i. $I_{g_i}(S, u_i)$ is indicator used to indicate whether S can reach u_i in g_i (i.e., $I_{g_i}(S, u_i) = 1$) or not (i.e, $I_{g_i}(S, u_i) = 0$). Then $\sigma(S)$ can be calculated as follows:

$$\sigma(S) = \sum_{g_i \in \tilde{G}} \sum_{u_i \in U'} I_{g_i}(S, u_i)P(g_i) \tag{13}$$

Since $S \subseteq T$, if S can reach u_i, T can reach u_i too, i.e., $I_{g_i}(T, u_i) \geq I_{g_i}(S, u_i)$. By Equation, $\sigma(T) - \sigma(S) \geq 0$. Therefore, the value function $\sigma(S)$ is monotonic.

Moreover, for $u_j \in U' \setminus T$, we can obtain the following equations according to Equation 13:

$$\sigma(T \cup \{u_j\}) - \sigma(T) = \sum_{g_i \in \tilde{G}} \sum_{u_i \in U'} (I_{g_i}(T \cup \{u_j\}, u_i) - I_{g_i}(T, u_i))P(g_i) \tag{14}$$

Similarly, for S:

$$\sigma(S \cup \{u_j\}) - \sigma(S) = \sum_{g_i \in \tilde{G}} \sum_{u_i \in U'} (I_{g_i}(S \cup \{u_j\}, u_i) - I_{g_i}(S, u_i))P(g_i) \tag{15}$$

Since $I_{g_i}(T, u_i) \geq I_{g_i}(S, u_i)$, if u_j can reach u_i in g_i, $I_{g_i}(T \cup \{u_j\}, u_i) = I_{g_i}(S \cup \{u_j\}, u_i) = 1$. Then we can obtain

$$\begin{aligned} I_{g_i}(T \cup \{u_j\}, u_i) - I_{g_i}(T, u_i) - (I_{g_i}(S \cup \{u_j\}, u_i) - I_{g_i}(S, u_i)) \\ = I_{g_i}(S, u_i) - I_{g_i}(T, u_i) \leq 0 \end{aligned} \tag{16}$$

Otherwise, if u_j cannot reach u_i, we have

$$\begin{aligned} & I_{g_i}(T \cup \{u_j\}, u_i) - I_{g_i}(T, u_i) - (I_{g_i}(S \cup \{u_j\}, u_i) - I_{g_i}(S, u_i)) \\ & = I_{g_i}(T, u_i) - I_{g_i}(T, u_i) - (I_{g_i}(S, u_i) - I_{g_i}(S, u_i)) = 0 \end{aligned} \tag{17}$$

By combining Equations 14, 15, 16 and 17, we have $\sigma(T \cup \{u_j\}) - \sigma(S \cup \{u_j\}) \leq 0$. Therefore, the submodular property is proved.

3.2 DMM Model

In this part, we will introduce the Distance-based Mobility Model (DMM) [15], which is used to calculate the probability that users visit the query location, i.e., $P_{l_q}(u_j)$.

In DMM, the probability of user u_i visiting the location l_q can be computed as follows:

$$P_{l_q}(u_i) = \sum_{l} P_l^i f^i(d(l, l_q)) \tag{18}$$

where $P_{l_q}(u_i)$ represents the probability of user u_i visiting l_q, P_l^i denotes the stationary distribution of user u_i staying in location l, $f^i(d(l, l_q))$ denotes the probability density of user u_i moving from location l to location l_q, and $d(l, l_q)$ denotes the distance between location l and location l_q. P_l^i can be calculated using random walk and f^i is the Pareto distribution function $f(x; \eta; \tau) = \frac{\eta\tau^{\eta}}{x^{\eta+1}}$, where η denotes the shape parameter and τ is the minimum value of x. In DMM model, τ is fixed (i.e., $\tau = 1$). The maximum likelihood concept is used to estimate η from the historical check-ins of each user. The corresponding function is as follows:

$$\prod_{i}^{N} \mathcal{L}(x_i; \eta; \tau = 1) = \prod_{i}^{N} \frac{\eta}{x_i^{\eta+1}} \tag{19}$$

where $x_i = y_i + 1$ and y_i is the distance of the i-th next movement of a user. Then to maximize the objective function, the close form of δ is derived in Equation 20.

$$\begin{gathered} \frac{d}{d\eta} \prod_{i}^{N} \frac{\eta}{x_i^{\eta+1}} = 0, \text{ where } x_i > 0 \\ \prod_{i}^{N} \frac{\eta}{x_i^{\eta+1}} \cdot \left[\frac{d}{d\eta} \sum_{i}^{N} \ln \frac{\eta}{x_i^{\eta+1}} \right] = 0 \\ \eta = \frac{N}{\sum_i^N \ln x_i}, \text{ where } \sum_{i}^{N} \ln x_i \neq 0 \end{gathered} \tag{20}$$

Based on Equations 18 and 20, the probability that user u_i visits l_q can be computed as follows:

$$\begin{aligned} P_{l_q}(u_i) &= \sum_l P_l^i \int_{d(l,l_q)}^{\infty} f^i(x)dx \\ &= \sum_l P_l^i (d(l,l_q)+1)^{-\eta} \end{aligned} \tag{21}$$

4 Similarity-aware Influence Maximization Algorithm

In this section, we will show the method used to calculate the influence spread of users and the details of our proposed algorithms.

4.1 Influence Spread Calculation

To solve the SIM problem, a fundamental step is to calculate the influence spread of nodes. In this paper, we propose the *maximum online-offline influence* (MOI_{l_q}) path to approximate the influence propagation.

Given a geo-social network G and two nodes $u_i, u_j \in U$, u_i can activate u_j through the path $p(u_i,u_j) = \langle u_i = p_1, p_2, \cdots, p_{m'} = u_j \rangle$, where $\langle p_{i'}, p_{i'+1} \rangle \in E$. The probability that u_i activates u_j online through $p(u_i,u_j)$ can be calculated as $P(p(u_i,u_j)) = \prod_{i'=1}^{m'-1} P_{nf}(p_{i'}, p_{i'+1})$, where $P_{nf}(p_{i'}, p_{i'+1}) = P_{st}(p_{i'}, p_{i'+1}) P_{l_q}(p_{i'+1})$. The path with the largest probability will be selected, which presents the greatest opportunity for u_i to activate u_j. This path is called the *maximum online-offline influence* path, denoted as MOI_{l_q}, i.e., $MOI_{l_q}(u_i,u_j) = \arg\max_{p \in p(u_i,u_j,G)} P(p)$, where $p(u_i,u_j,G)$ denotes all the paths between u_i and u_j in G. However, the probability of many MOI_{l_q} is quite small, thus it is insignificant to measure the contributions of these MOI_{l_q}. We use a threshold θ_p to prune these paths. Specifically, if $P(MOI_{l_q}(u_i,u_j)) < \theta_p$, we think that u_i can not activate u_j, i.e., $P(MOI_{l_q}(u_i,u_j)) = 0$.

Like the calculation of MOI_{l_q}, we also use an offline threshold θ_{off} to represent the expectation of the promoter on the target users. The definition of θ_{off} is as follows:

Definition 15. (Offline Threshold) The offline threshold, θ_{off}, is a metric used to prune users who have low probability to visit the query location.

In other words, the target node u_i needs to satisfy $P_{l_q}(u_i) \geq \theta_{off}$.

Based on $MOI_{l_q}(u_i,u_j)$, for an arbitrary node u_i and the given target location l_q, we can calculate its influence spread IS_{u_i} as:

$$IS_{u_i} = \sum_{\substack{u_j \in U \\ P_{l_q}(u_j) \geq \theta_{off}}} P(MOI_{l_q}(u_i,u_j)) \tag{22}$$

Figure 3 shows an example of SIM problem (the bidirectional solid line between u_i and u_j means that u_i and u_j are friends in a social network), where we set $\theta_{on} = \theta_{off} = 0.1$. Taking u_1, u_2 and u_3 in Figure 1 into account and assuming that they are friends in a social network. As shown in Figure 1, there are four ROI after clustering. u_1, u_2 and u_3 mainly check in at r_1, r_2 and r_3. Based on the analysis of Section 2.2, it is clear that the check-in behavior of u_1 and u_2 is more similar than that of u_1 and u_3. The information

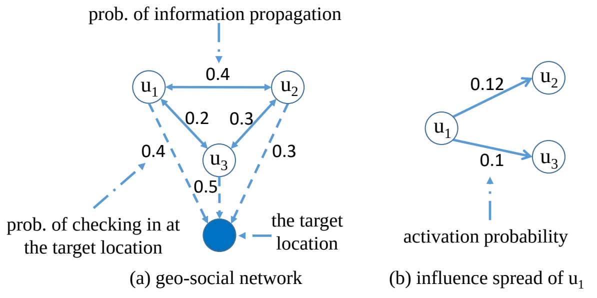

prob. of information propagation
0.4
u1
u2
0.2
0.3
0.4
u3
0.3
0.5
prob. of checking in at the target location
the target location
(a) geo-social network
0.12
u1
u2
0.1
u3
activation probability
(b) influence spread of u1

Fig. 3. An example of SIM problem

propagation probability between u_1 and u_2 should be larger than that between u_1 and u_3 (see the solid lines in Figure 3(a)). Assuming that u_2 and u_3 have the probability of 30% and 50% to check in at the target location respectively (see the dotted lines in Figure 3(a)), so the activation probability of u_1 to u_2 is $0.4 \times 0.3 = 0.12$ and u_1 to u_3 is $0.2 \times 0.5 = 0.1$ (see Figure 3(b)). The users who can be converted into offline-activated by u_i are called influencers of u_i. Based on Equation 22, we can obtain the influence spreads of all the nodes.

4.2 Influence Propagation Trees Based Algorithm

Since the Naive Greedy algorithm (see Algorithm 1) spends much time calculating the marginal gain of every node in each iteration, which takes $O(knRm)$ to complete, where n is the number of nodes, R is the iteration number of Monte-Carlo (MC) simulation and m is the number of edges. The intractable time complexity limits its application in large-scale networks. In order to improve the efficiency, in this part, we will propose a new algorithm called Influence Propagation Trees Based (IPTB) algorithm , which is based on Influence Propagation Trees (IPT) including *influence propagation in tree (IPIT) and influence propagation out tree (IPOT)*. First, we will introduction the IPT, and then we will show how to select seeds based on IPT.

Algorithm 1: Naive Greedy Algorithm

```
   Input: $G$ : a geo-social network; $l_q$ : query location
   Output: $S$ : a set of $k$ nodes
1  Pre-compute $P_{st}(u_i,u_j)$, $P_{l_q}(u_j)$;
2  $S = \emptyset$;
3  for $h = 1$ to $k$ do
4      $u_i = \arg\max_{u_j \in U \setminus S}(\sigma(S \cup \{u_j\}) - \sigma(S))$;
5      $S = S \cup \{u_i\}$;
6  return $S$
```

4.2.1 *Influence Propagation Trees*

From Equation 22, we can clearly see that the influence spread of user u_i is related to the users that can be activated (i.e., transited into the offline-activated state) by u_i. So we need to develop a particular mechanism to estimate IS_{u_i}, especially when the influencers of u_i are chosen as seeds.

For a node u_i, the estimation of IS_{u_i} requires to access all influencers and influential predecessors who can activate

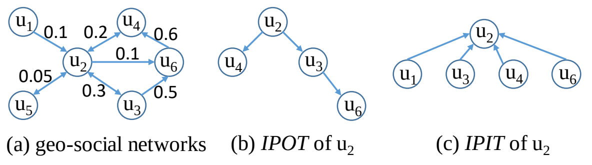


Fig. 4. *IPOT* and *IPIT* buliding

u_i. Since the nodes in a tree can be efficiently visited by its root, we develop two tree structures to achieve this goal. One includes all influencers of u_i to update IS_{u_i} and the other contains all influential predecessors of u_i.

First, we can calculate the original value of IS_{u_i} by using Equation 22. To estimate the value of IS_{u_i} after the first seed having been found, we need to organize all influencers of u_i into *influence propagation out tree*, which is defined as follows.

Definition 16. (Influence Propagation Out Tree) Given a node $u_i \in U$, the *influence propagation out tree* (O_{u_i}, E_{u_i}) of u_i, denoted as $IPOT(u_i)$, contains influencers of u_i, where O_{u_i} is the set of all the influencers of u_i and E_{u_i} is the set of all edges from u_i to u_j if $u_j \in O_{u_i}$.

Definition 17. (Value of IPOT) Given a node u_i, the value of $IPOT(u_i)$, i.e., VO_{u_i}, can be computed as:

$$VO_{u_i} = \sum_{u_j \in O_{u_i}} IPOT(u_i, u_j) \tag{23}$$

where $IPOT(u_i, u_j)$ denotes the probability that u_i activates u_j, which equals to $P(MOI_{l_q}(u_i, u_j))$. Note that VO_{u_i} is also the upper bound influence spread of node u_i, which we will prove in Lemma 3.

To estimate the influence spread of $u_i's$ predecessors, we organize the predecessors of u_i into a tree called *influence propagation in tree (IPIT)*. We do not have to store all the predecessors of u_i, since the probability that some nodes activate u_i is small. We only store the nodes that activate u_i through $MOI_{l_q}(u_j, u_i)$ and $P(MOI_{l_q}(u_j, u_i)) \geq \theta_p$ with u_i as the root and ignoring rings. These nodes are called influential predecessors of u_i. Different from *IPOT*, if u_j is an influential predecessor of u_i, u_j is added into the $IPIT(u_i)$ as one child of u_i, which means that a child node points to its father node to represent the direction of $p(u_j, u_i, G)$ in $IPIT(u_i)$.

The formal definition of *IPIT* is shown as follows.

Definition 18. (Influence Propagation In Tree) Given a node $u_i \in U$, $IPIT(u_i)$ is a tree which contains all the influential predecessors of u_i.

Assume that all users in Figure 1 form a directed network (see Figure 4(a)). In Figure 4(a), the single way arrow from u_i to u_j means that the information can only be passed from u_i to u_j, and the two-way arrow between u_i and u_j means that the information can be propagated between u_i and u_j. We set $\theta_p = 0.1$. Note that the propagation probability between u_i and u_j in Figure 4(a) equals to $P_{nf}(u_i, u_j)$. The $IPOT(u_2)$ and $IPIT(u_2)$ in Figure 4(a) are shown in Figure 4(b) and Figure 4(c), respectively. When building $IPOT(u_2)$, all the influencers of u_2 are added into $IPOT(u_2)$. On the other hand, all influential predecessors of u_2 are added into $IPIT(u_2)$.

4.2.2 Node Selection

Since most nodes are insignificant, we do not need to calculate the influence spread of every node. We assume that only $n \times \delta$ of nodes are likely to be chosen as seeds, which means the rest are regarded as insignificant nodes. The results of our methods on different datasets demonstrate the validity of our assumption. Ignoring the insignificant nodes can effectively reduce the time cost. We use the *Neighbor Contribution (NC)* to measure the local influence of users. The *NC* of u_i is calculated as $NC_{u_i} = \sum_{each\ neighbor\ u_j\ of\ u_i\ and\ P_{l_q}(u_j) \geq \theta_{off}} P_{st}(u_i, u_j) P_{l_q}(u_j)$, which not only considers the probability that users head for the query location but also prunes the insignificant neighbors. First of all, we give an overview of whole process of IPTB algorithm.

Algorithm 2: Influence Propagation Trees Based Algorithm

```
    Input: $G$ : a geo-social network; $k$: seed set size; $l_q$ :
           query location
    Output: $S$ : a set of $k$ nodes
 1  $K = \emptyset$, $S = \emptyset$, $\sigma(S) = 0$, $Q = \emptyset$;
 2  for each node $u_i \in U$ do
 3      compute $NC_{u_i}$ and $P_{l_q}(u_i)$;
 4  for $h = 1$ to $n \times \delta$ do
 5      select $u_i = \arg\max_{u_j \in U \setminus K} NC_{u_j}$;
 6      $K = K \cup \{u_i\}$;
 7  Compute $P(MOI_{l_q}(u_i, u_j))$ for $u_i \in K$;
 8  Set $VO_{u_i} = \sum_{u_j\ if\ P_{l_q}(u_j) \geq \theta_{off}} P(MOI_{l_q}(u_i, u_j))$;
 9  for $h = 1$ to $o$ do
10      select $u_i = \arg\max_{u_j \in K \setminus Q} VO_{u_j}$;
11      $Q$.push($u_i$);
12      building $IPOT(u_i)$ and $IPIT(u_i)$;
13  for each node $u_i \in U$ do
14      for each neighbor $u_j$ of $u_i$ do
15          if $P_{l_q}(u_j) < \theta_{off}$ then
16              delete $E(u_i, u_j)$ from $G$;
17  for $h = 1$ to $k$ do
18      set $T(u_i) = 0$ for $u_i \in Q \setminus S$;
19      while TRUE do
20          $u_i = Q.pop()$;
21          if $T(u_i) = 0$ then
22              $VO_{u_i} = MC(S \cup \{u_i\}) - \sigma(S)$;
23              $T(u_i) = 1$;
24              $Q.push(u_i)$;
25          else if $VO_{u_i} \geq \max_{u_j \in Q \setminus (S \cup \{u_i\})} VO_{u_j}$ then
26              $\sigma(S \cup \{u_i\}) = \sigma(S) + VO_{u_i}$;
27              $S = S \cup \{u_i\}$;
28              for each node $u_j \in IPIT(u_i)$ do
29                  $VO_{u_j} = VO_{u_j} - IPOT(u_j, u_i)$
30              break;
31  return S
```

In IPTB, we first compute $n \times \delta$ nodes with the largest *NC* value (Line 2-6), where $0 < \delta \ll 1$ means how many candidates will be taken into account, and then we store the o nodes with the largest *VO* value in the priority queue Q, and build both *IPOT* and *IPIT* for them (Line 7-12), where o is a positive integer. To select a suitable value of o, we calculate the total influence spread of all candidates and select top-o nodes with largest influence spread. Besides, the total influence spread of those o nodes is no less than 80% of the influence spread of all candidates. We employ Monte-Carlo (MC) simulation algorithm to estimate the influence spread of a given node, where larger graph will lead to higher time complexity. In order to decrease the time cost, in IPTB, we prune the insignificant edges to generate a smaller graph (Line 13-16). In each iteration, we select the node u_i with the largest VO_{u_i} (i.e., the head of Q) (Line 20) and compute its precise margin increase by MC simulation (Line 22). If $u_i's$ precise margin increase is larger than the VO_{u_j} of any other node u_j in Q, it will be selected as a new seed (Line 25-27). Then we will remove u_i from $IPOT(u_j)$ if u_j is in $IPIT(u_i)$ and recompute the VO_{u_j} (Line 28-29). We use $T(u_i) = 0$ to denote that the MC simulation has not been used to estimate $\sigma(S \cup \{u_i\})$ yet in the current iteration, and $T(u_i) = 1$ to mean the MC simulation has already been computed to estimate $\sigma(S \cup \{u_i\})$.

Now, we analyze the time complexity of IPTB algorithm. Line 2 to line 6 select the nodes with largest *NC* value, which takes $O(n' \log n + m)$, where $n' = n \times \delta$ and m is the number of edges. Line 7 computes the *maximum online-offline influence (MOI_{l_q})* path, which takes $O(n'm \log n)$. Then, we compute *VO* and build IPT for the nodes (Line 9-12), which takes $O(nn' + o \log n' + o + m)$. Line 13 to line 16 delete the insignificant edges of G, with time complexity of $O(m)$. Finally, line 17 to line 30 select the most influential nodes, the worst case is that the precise influence spread of the nodes needs to be updated, which takes $O(kR'(\log o + Rm'))$, where R' ($R' < o$) is the maximum iteration number of selecting the new seed, m' ($m' \ll m$) is the total edges after deleting and R is the iteration number of MC simulation. So the time complexity of IPTB algorithm is no more than $O(n' \log n + m + n'm \log n + nn' + o \log n' + o + m + kR'(\log o + Rm'))$.

Lemma 3. For a given node u_i, VO_{u_i} is the upper bound influence spread of u_i.

Proof 2. To prove the Lemma 3, we choose three different nodes u_i, u_j and u_h. Node u_h is in the path $MOI_{l_q}(u_i, u_j)$ and $P(MOI_{l_q}(u_i, u_j)) \geq \theta_{on}$. It is clear that u_i is one of the nodes in $IPIT(u_h)$. Supposing that u_h is selected as the new seed, so we need to recompute the VO_{u_i}. In this case, u_i cannot activate u_j through original Maximum Influence Path from u_i to u_j, denoted as $MOI_{l_q}(u_i, u_j)$. We have to choose another influence path (i.e., the second largest influence path) to calculate the influence contribution of u_j to u_i. This new path is denoted as $MOI'_{l_q}(u_i, u_j)$. It is clear that $P(MOI'_{l_q}(u_i, u_j)) \leq P(MOI_{l_q}(u_i, u_j))$. Based on Equation 23, we have:

$$\begin{aligned} VO'_{u_i} &= \sum_{i' \notin \{h,j\} \text{ and } u_{i'} \in O_{u_i}} IPOT(u_i, u_{i'}) + P(MOI'_{l_q}(u_i, u_j)) \\ &\leq \sum_{i' \notin \{h,j\} \text{ and } u_{i'} \in O_{u_i}} IPOT(u_i, u_{i'}) + P(MOI_{l_q}(u_i, u_j)) \\ &= VO_{u_i} \end{aligned}$$

where VO'_{u_i} is the real influence spread of u_i and VO_{u_i} is the influence spread of u_i calculated by IPTB algorithm.

4.3 Cutting Tails Algorithm

In [22], Leskovec et al. propose a Cost-Effective Lazy Forward (CELF) algorithm which can match the accuracy of the original greedy algorithm, while it spends too much time calculating the influence spread of insignificant nodes. To overcome this limitation, we propose a new greedy algorithm called Cutting Tails (CT) algorithm. Specifically, in CT algorithm, we first find out the $n \times \delta$ nodes with the largest *NC* value as candidates (Line 2-6), and then we remove the insignificant edges (Line 7-10). We compute the influence spread $\sigma(\cdot)$ by MC simulations for those candidates (Line 11-13), these two steps take $O(n' \log n + m + m + Rm)$ time, where $n' = n \times \delta$, m is the number of edges and R is the iteration number of MC simulation. By those steps, we can avoid the unnecessary time for calculating the influence spread of many insignificant nodes. After calculating the precise influence spread of these nodes, in each iteration, the node u_i with the largest margin influence spread $\sigma(u_i)$ is selected (Line 15). If $u_i's$ margin influence spread has been computed in the current step, u_i will be selected as the new seed (Line 16-18); otherwise, $u_i's$ margin influence spread will be recomputed (Line 20). $u_i.flag == |S|$ indicates that the precise margin influence spread of u_i has been computed in the current step; otherwise, we need to recompute the precise margin increase of u_i. The seeds selection steps (Line 14-21) take $O(kR'(\log n' + Rm))$ time, where R' ($R' < n'$) is the maximum iteration number of selecting the new seed, and thus CT takes $O(n' \log n + m + m + Rm + kR'(\log n' + Rm))$ time to complete.

5 Experiments

We conduct experiments on the real-world datasets to evaluate our proposed algorithms. We implement all algorithms using python. All experiments are run on a Linux (Ubuntu 16.04) machine with Intel(R) Xeon(R) E5-2650 v4 2.20GHz processor and 256G memory.

5.1 Datasets

In this paper, we select the Brightkite dataset (BK for short) [16] and FourSquare dataset (FS for short) [23], for evaluation. In the BK dataset, there are 58,228 users, 214,078 social connections and 4,491,143 check-ins from April 2008 to October 2010. In the FS dataset, there are 11,326 users, 47,164 social connections and 1,385,223 check-ins from January 2011 to December 2011.

5.2 Baseline Methods

We compare the IPTB algorithm and the CT algorithm with the following algorithms.

- CELF [22]. The Cost-Effective Lazy Forward algorithm.
- IMM [8]. The Influence Maximization via Martingales algorithm that is an extension of TIM (Two-phase Influence Maximization) algorithm [24], which

```
Algorithm 3: Cutting Tails algorithm
Input: $G$ : a geo-social network; $k$: seed set size; $l_q$ :
       query location
Output: $S$ : a set of $k$ nodes
$K = \emptyset$, $S = \emptyset$;
for each node $u_i \in U$ do
    Compute $NC_{u_i}$ and $P_{l_q}(u_i)$;
for $h = 1$ to $n \times \delta$ do
    select $u_i = \arg\max_{u_j \in U \setminus K} NC_{u_j}$;
    $K = K \cup \{u_i\}$;
for each node $u_i \in U$ do
    for each neighbor $u_j$ of $u_i$ do
        if $P_{l_q}(u_j) < \theta_{off}$ then
            delete $E(u_i, u_j)$ from $G$;
for each $u_i \in K$ do
    $\sigma(u_i) = MC(u_i)$;
    $u_i.flag = 0$;
while $|S| < k$ do
    $u_i = \arg\max_{u_j \in K \setminus S} \sigma(u_j)$;
    if $u_i.flag == |S|$ then
        $S = S \cup \{u_i\}$;
        continue;
    else
        $\sigma(u_i) = MC(S \cup \{u_i\}) - MC(S)$
    $u_i.flag = |S|$
return S
```

takes advantage of martingales to improve the efficiency of TIM algorithm.

- TPH [21]. The Two-Phase Heuristic algorithm that takes both online information propagation and offline consumption behavior into account.
- SD [7]. The Single Discount algorithm that reduces the degree of a node if its neighbors are chosen as the seeds.
- Degree. A heuristic algorithm based on degree centrality, in which nodes with higher degree are more influential.

Since CELF and IMM concentrate on the traditional influence maximization problem, to extend these algorithms to solve the problem we propose, we reconstruct the original network into a new directed network as discussed in Proof 1. These algorithms can be directly used to select seeds in the new directed network.

5.3 Parameter Settings

To obtain the precise influence spread for each seed set, we run Monto-Carlo simulations on the networks $10,000$ times and take average results as the influence spread. To get the hierarchical similarity of each user, we calculate the spatial similarity on three levels in both datasets, and weight of the i-th level is set to 2^i, i.e., $\beta_i = 2^i$. The parameter used to measure the difference between weekdays and weekends is set to $\frac{5}{7}$, i.e., $\gamma = \frac{5}{7}$. We set the threshold of *maximum online-offline influence* (MOI_{l_q}) path to 0.001, i.e., $\theta_p = 0.001$. The size of the priority queue Q in IPTB algorithm is set to $2,000$ (i.e., $o = 2,000$) for BK and 300 for FS. For all algorithms, we limit the largest size of the seed set to be 50, due to the fact that the limited budget in real world limits the number of seeds. We select San Francisco Caltrain Station, San Francisco (37.776430N, 122.394318W) (denoted as l_1) and Central Park, New York City (40.780606N 73.968088W) (denoted as l_2) as the query locations (denoted as l_q), and neglect users who have less than 10 check-in records. To get the suitable parameters for clustering, we calculate the average check-in locations for all users in each day, week and month, respectively. Table 2 shows the details of the corresponding parameters of the clustering.

TABLE 2
The parameters of each level in DBSCAN

Brightkite			FourSquare		
level	ϵ	Minpts	level	ϵ	Minpts
3	0.05	3	3	0.05	3
2	0.1	10	2	0.1	7
1	0.2	20	1	0.2	20

The main experimental parameters are shown in Table 3, where the default value of each parameter is underlined.

TABLE 3
Experiment Parameters

Parameters	Values
Seed set size k	10, 20, 30, 40, <u>50</u>
The weight of spatial similarity λ	0.1, 0.3, <u>0.5</u>, 0.7, 0.9
Information propagation probability P_0	0.02, 0.04, 0.06, 0.08, <u>0.1</u>
The weight of spatio-temporal similarity w	10, 15, <u>20</u>, 25, 30
The query location l_q	<u>l_1</u>, l_2
Offline threshold θ_{off}	0.1, 0.2, <u>0.3</u>, 0.4, 0.5
Proportion of candidates δ	0.01, 0.05, <u>0.1</u>, 0.2, 0.3

5.4 Experimental Results

To evaluate the effectiveness and efficiency of the proposed algorithms, two metrics are compared between our proposed algorithms and other algorithms: 1) Influence Spread (IS for short): the number of nodes that can be activated given a seed set; 2) Running Time (RT for short): the CPU time cost for finding the optimal seed set.

Effect of Seed Set Size. First, we investigate how the seed set size k affects the effectiveness and efficiency of our algorithms. As shown in Figures 5(a) and 5(b), in both datasets, with the increase of k, IMM, IPTB, CT and CELF show higher influence spread than Degree, SD and TPH. IPTB and CT can essentially match the influence spread of CELF on any k. The TPH is better than Degree in BK dataset, while worse than Degree in FS dataset. The reason is that the number of nodes with both large degree and local influence spread is small in BK dataset while large in FS dataset. Therefore, the influence spread of Degree is better than that of TPH in FS dataset. The running time of IPTB, CT and CELF is strongly related to the seed set size in BK dataset (see Figure 5(c)), because the main time cost is the iteration number of determining the new seed. While in FS dataset, the main running time of those algorithms is the influence spread calculation of candidates, indicating that the running time of those algorithm is not affected by the

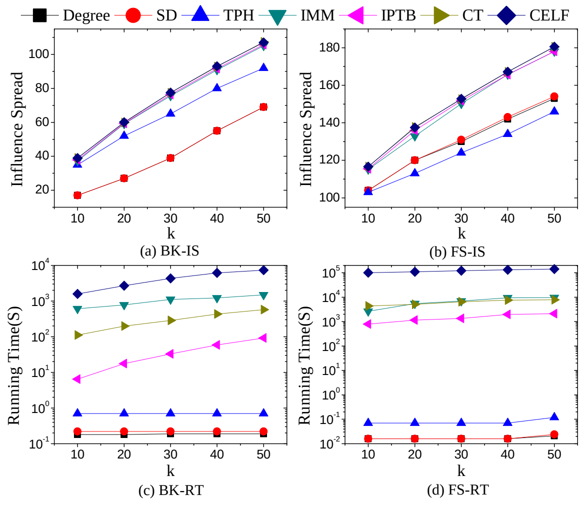


Fig. 5. Performance of our proposed algorithms: effect of k.

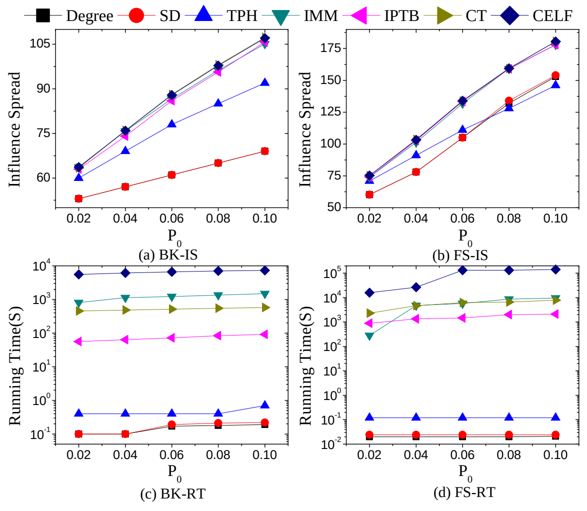


Fig. 6. Performance of our proposed algorithms: effect of P_0.

seed set size (see Figure 5(d)). Moreover, the running time of the algorithms we proposed (i.e., IPTB and CT) is smaller than that of CELF and IMM on any k since we neglect many insignificant nodes. Although our proposed algorithms cost more time than Degree, SD and TPH. The influence spread of IPTB and CT is much larger than that of those three algorithms.

Effect of Propagation Probability. We further study the effect of propagation probability P_0 by changing it from 0.02 to 0.1. From the Figures 6(a) and 6(b), we can see that TPH, IMM, IPTB, CT and CELF perform almost the same on lower P_0. This is due to the fact that when P_0 is small, it is difficult for seeds to activate other nodes. We also observe that in FS dataset, the TPH is better than Degree on smaller P_0 while worse than Degree on larger P_0. That is because with the increase of P_0, the probability that the neighbors are activated becomes larger, i.e., the nodes with large degree (also with large local influence spread in FS dataset) can activate more nodes than nodes only with large local influence spread value. In terms of influence speed, IMM, IPTB, CT and CELF are much better than Degree, SD and TPH because they calculate the precise margin increase of nodes. When P_0 becomes larger, IPTB and CT can also match the influence spread of IMM and CELF while keeping lower time cost (see Figures 6(c) and 6(d)). The results indicate that IPTB and CT perform better than IMM, CELF and other three heuristic algorithms regardless of the change of P_0.

Effect of Weight of Spatial Similarity. As shown in Figures 7(a) and 7(b), the influence spread of all algorithms is growing with the increasing λ, which indicates that larger spatial similarity leads to larger information propagation probability (see Equations 11 and 12), making it easier for the seeds to activate other nodes. When λ becomes extremely large, the influence spread of CELF is slightly above that of IPTB. Because in IPTB, we use the *maximum online-offline influence* (MOI_{l_q}) path to approximate the influence spread of nodes, with the increase of information propagation probability, the correlation between different paths becoming stronger (i.e., the target can also be activated by the seeds through other paths), resulting in a small

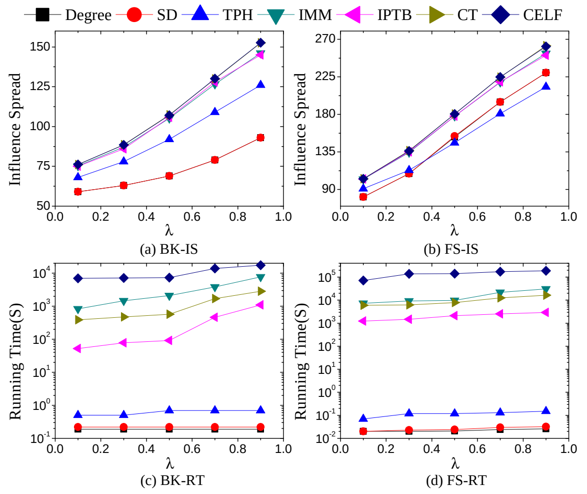


Fig. 7. Performance of our proposed algorithms: effect of λ.

deviation of IPTB in estimating influence spread of each node. The value of λ and the running time of IMM, IPTB and CT are strongly correlated (see Figures 7(c) and 7(d)), since activation probability will change with the varying λ, which affects the influence spread of nodes. For IPTB and CT, larger λ leads to the greater change in the marginal gain of nodes, resulting in the running time of seed selection. For IMM, it takes more time to calculate the reverse reachable set for a given node with the increase of λ. IPTB and CT can essentially match the influence spread of IMM and CELF, respectively, while keeping lower time cost across different λ.

Effect of Enhancement Factor. As shown in Figures 8(a) and 8(b), the influence spread of all algorithms is growing with the increasing enhancement factor w. This is because when w becomes larger, the information propagation probability among users will be larger (see Equation 12), which makes it easier for the seeds to activate other nodes. The influence spread of Degree and SD is almost the same because both the online and offline behaviors are taken into account in PTC model. Discounting the degree of nodes (i.e., SD) has some positive impact in selecting seeds, while the

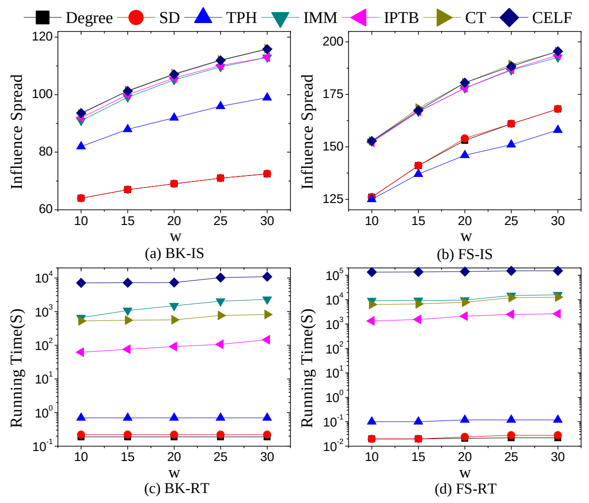


Fig. 8. Performance of our proposed algorithms: effect of w.

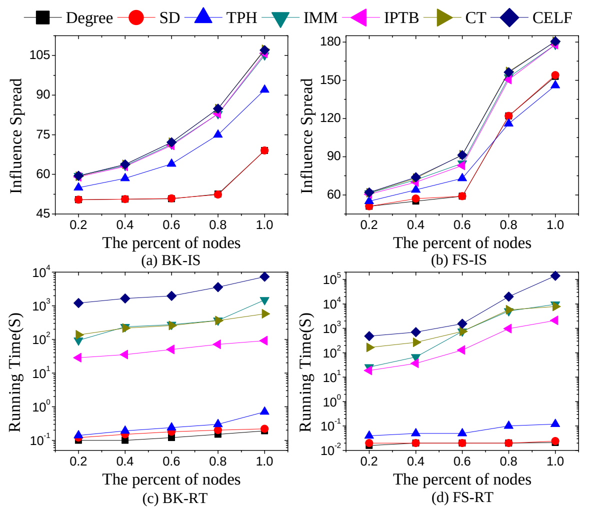


Fig. 9. Performance of our proposed algorithms: effect of graph size.

advantage becomes weaker when considering the offline behavior. CT can essentially match the influence spread of CELF while keeping lower time cost on any w in both datasets (see Figures 8(c) and 8(d)). Although the influence spread of CELF is slightly above that of IPTB, the running time of IPTB is much lower than that of CELF. Moreover, IPTB can guarantee the same influence spread as IMM while costing lower time.

Effect of Graph Size. To study the scalability of the proposed algorithms, we select a part of nodes in the network to form a subgraph. Figures 9(a) and 9(b) show IPTB and CT can essentially match the influence spread of IMM and CELF, and the influence spread of our proposed algorithms is better than that of other three heuristic algorithms with the increase of graph size. Figures 9(c) and 9(d) depict that the running time of all algorithms increases with the graph size growing, since we need to consider more nodes. The running time of IMM is lower than CT in smaller subgraph because IMM takes smaller time to calculate the reverse reachable set and does not need to compute the influence spread of every node. Our proposed algorithms perform better than other three heuristic algorithms. The running time of IPTB and CT is much lower than CELF, which demonstrates the superiority of our proposed algorithms.

Effect of Query Location. To study the robustness of our proposed algorithms at different locations, we choose l_2 as the query location. As shown in Figures 10(a) and 10(b), our proposed algorithms show higher influence spread than that of Degree, SD and TPH. IPTB and CT can essentially match the influence spread of CELF on any k in both datasets, and show better performance than IMM in FS dataset. IPTB performs extremely well at different locations (see Figures 5 and 10). The running time of CT is lower than that of IMM in BK dataset while slightly higher than that of IMM in FS dataset. The reason is that in BK dataset, the number of nodes and edges is much larger, leading to higher time cost of IMM, while CT spends less time calculating the influence spread and marginal gain of candidates because we delete many insignificant edges (see Algorithm 3). The running time of IPTB is much lower than that of IMM and

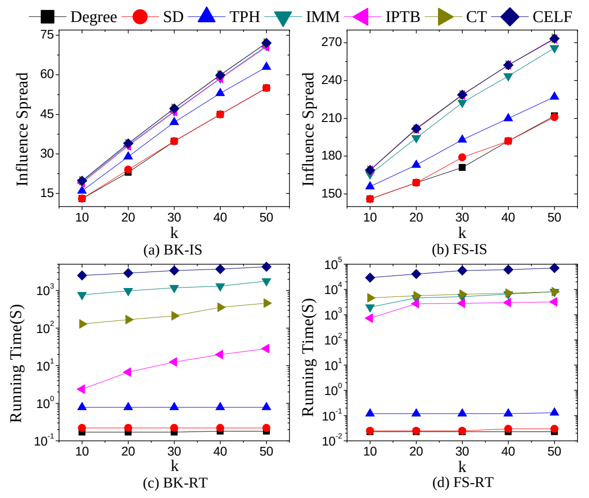


Fig. 10. Performance of our proposed algorithms: effect of l_2.

CELF regardless of the change of locations, and CT spends less time determining the seeds than CELF. The results indicate our IPTB and CT algorithms are more effective in finding top influential seeds compared with the state-of-the-art algorithms.

Effect of Offline Threshold. We further study the effect of offline threshold, θ_{off}, by changing it from 0.1 to 0.5. From Figures 11(a) and 11(b), we can see that all algorithms perform almost the same on larger θ_{off}. This is due to the fact that on those two datasets, the probability of most of the users visiting the target location is small. It is difficult for seeds to activate other nodes offline on larger θ_{off}. When θ_{off} becomes smaller, the influence spread of IPTB is lower than that of CELF. The reason is that we calculate the *maximum online-offline influence* (MOI_{l_q}) path to approximate the influence propagation of users in IPTB. With smaller θ_{off}, other paths will also contribute to the influence spread. As a result, the computation of influence spread in IPTB is lower than the real influence spread of each user. CT can essentially match the influence spread of CELF, and IPTB performs better than other algorithms except CELF on smaller θ_{off}. The value of θ_{off} and the

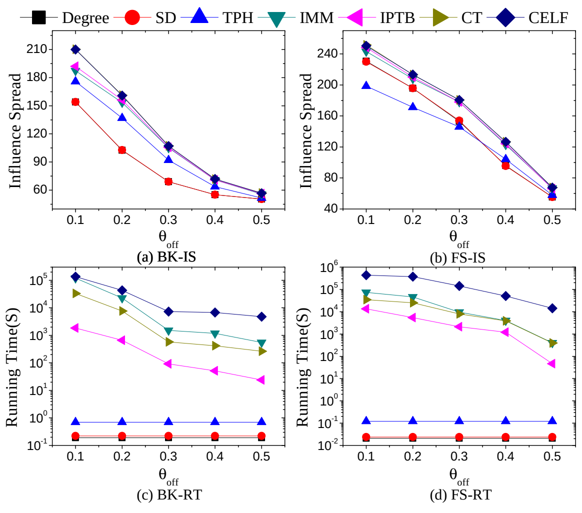


Fig. 11. Performance of our proposed algorithms: effect of θ_{off}.

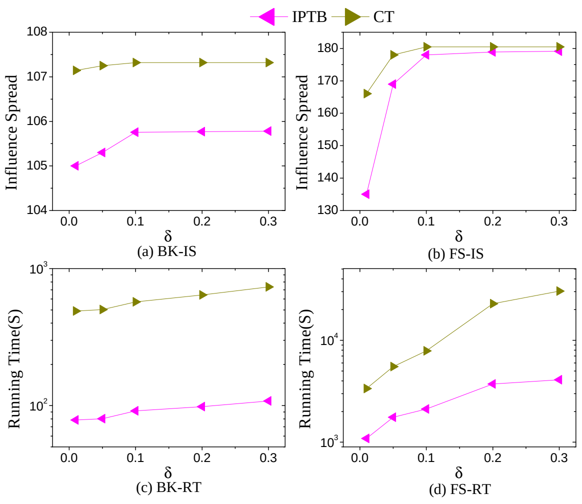


Fig. 12. Performance of our proposed algorithms: effect of δ.

running time of IMM, IPTB, CT and CELF are strongly correlated (see Figures 11(c) and 11(d)), since the influence propagation process will change with the varying θ_{off}. For those algorithms, smaller θ_{off} means that they will spend more time calculating the influence spread of each node. IPTB performs better than IMM, and CT can essentially match the influence spread of CELF, while keeping lower time cost across different θ_{off}.

Effect of Proportion of Candidates. To demonstrate the effect of propagation of candidates, δ, we change it from 0.01 to 0.3. Since δ only affects the influence spread of IPTB and CT, the results of other baselines are not presented. As shown in Figures 12(a) and 12(b), when δ is small, the influence spread of IPTB and CT becomes larger with the increase of δ. This is due to the fact that more nodes will be considered as candidates and more precise marginal gain of each node can be obtained. However, when δ reaches a certain value (i.e., 0.1), the influence spread of these two algorithms is not going to increase with further increase of δ. The reason is that the most influential candidates are already included when δ is 0.1. The value of δ and the running time of IPTB and CT are strongly correlated (see Figures 12(c) and 12(d)), because the number of candidates is larger with the increase of δ, and more time will be spent computing the influence spread of these candidates.

6 Related Work

Influence maximization is first formulated as a discrete optimization problem by Kempe, Kleinberg and Tardos [1]. Kempe et al. propose two discrete influence spread models, Independent Cascade (IC) model and Linear Thresholds (LT) model. In LT model, an activation threshold is defined: given a node, when the total influence contributions of its neighbors exceed this threshold, this node will be activated. On the other hand, in IC model, when a node becomes active, it will try to activate its neighbors. If the node has more than one active neighbors, it will be activated by them independently. The difference between IC model and LT model is that active nodes in IC model have only one chance to activate their neighbors, while in LT model, the active nodes will keep activating their inactive neighbors. The IC model and LT model can only be used to study influence maximization in online social networks. To study the influence maximization problem in location-based social networks (LBSN), we propose a new model, namely Propagation to Consumption (PTC), which considers both the online and offline behaviors of users. The meaning of *activation* (i.e., new seeds) in PTC model is different from that in IC model and LT model. In our proposed PTC model, the *activation* indicates that users can not be only influenced by their online neighbors, but can also choose to visit the query location, while the *activation* of IC model and LT model only means that users are influenced by their online neighbors. PTC model is the extension of IC model, which can be used to study the influence maximization problem in both online social networks and LBSN. For IC model and LT model, the authors prove that it is an NP-hard problem and give a greedy optimization algorithm with provable approximation guarantee. Many researchers try to reduce the time complexity of the greedy-based algorithm. In [22], the authors propose the Cost-Effective Lazy Forward (CELF) algorithm which exploits the submodular property and achieves 700 times speedup compared to the naive greedy algorithm in [1]. In [25], the author propose the CELF++ algorithm which exploits the property of submodularity of the spread function for influence propagation models and achieves higher efficiency than CELF. In [2], the authors prove that it is $\#P$-hard to calculate the influence spread and propose the PMIA algorithm to solve the influence spread maximization problem using the IC model. In [26], the authors propose the Reverse Influence Sampling (RIS) algorithm which utilizes the random sampling technique to generate a sparse hypergraph representation of the network and employs a greedy strategy to determine the k seeds based on the hypergraph. Motivated by RIS algorithm, Two-phase Influence Maximization (TIM) [24] and Influence Maximization via Martingales (IMM) [8] are proposed to decrease the time complexity of RIS while retain the approximation guarantee with the same high confidence. In [27], the authors study a *Community-diversified Influence*

Maximization (CDIM) problem, where the number of activated nodes as well as the number of communities to which the activated nodes belong can be maximized at the end of propagation process. Two algorithms and an innovative *CSPS-Tree* index have been proposed to solve the problem. In [28], the authors make use of the density metric and the influence of active users to address the problem of temporal interaction-biased community detection. A new influence propagation model is proposed to mirror the probabilities of edge activities, and expansion-driven algorithms are developed to find the activity-biased densest community. In [29], the authors propose an efficient reverse reachable set generation algorithm to address the influence maximization in large social networks. The proposed method is orders of magnitude faster than state-of-the-art algorithms. In [30], the authors propose a new propagation model, namely self-activation independent cascade, which considers self activation of nodes, and three influence maximization problems are studied based on the model. In [31], the authors study the problem of Distinct Influence Maximization, which aims to select users who maximize the number of distinct users influenced over a predefined window of time. Two different algorithms based on graph compression techniques are proposed to address the problem. In [32], the authors propose a new bound estimation techniques and node selection strategies to study the budgeted influence maximization problem.

Taking the location into account, Zhang et al. [33] attempt to measure the influence between users by considering both social relation and location information, and aim to identify influential events. In [34], there are two factors considered for users' check-in behaviors, personal preference and social influence. In [13], the authors study the location-aware influence maximization. They propose two greedy algorithms with $1-\frac{1}{e}$ approximation ratio and another two algorithms to meet the instant-speed requirement. In [35], the authors exploit kernel density estimate to model the individual user's check-in records. In [15], the authors design a distance-based mobility model which uses users' visiting history data to infer the propagation probability among users given a promoted location. In [36], the authors study the trajectory influence maximization problem which aims at selecting a trajectory set with advertisement to maximize the number of witness. In [11], the authors investigate the influence maximization based on geographical regions. The authors develop a greedy solution and an upper bound based solution to incrementally select the most influential nodes. In [37] , the authors propose a new holistic influence diffusion model which takes both cyber and physical user interactions into account, and formulate a new problem namely *holistic influence maximization* (*HIM*). Based on the framework, several algorithms are developed to effectively and efficiently solve the *HIM* query problem.

7 Conclusion

The similarity among users plays a significant role in information propagation and recommendations. In this paper, we study the problem of influence maximization based on user similarity, where each friendship-link is assigned a similarity probability according to the check-ins of users. We have taken an important step toward efficient influence maximization in geo-social network with users' spatial-temporal behavior. To settle the intractable complexity of this problem, we propose two algorithms (including Influence Propagation Trees Based algorithm and Cutting Tails algorithm), which incorporate not only the user similarity but also the online and offline behaviors of users. The experimental results demonstrate the effectiveness and efficiency of our proposed algorithms. In the future work, we would further consider the location preference of users by analyzing the semantic information of the query location and users' check-ins.

Acknowledgment

This work is supported by NSFC (No. 61972069, 61836007, 61832017) and Sichuan Science and Technology Program under Grant 2020JDTD0007.

References

[1] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *SIGKDD*, 2003, pp. 137–146.

[2] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *SIGKDD*, 2010, pp. 1029–1038.

[3] S. Chen, J. Fan, G. Li, J. Feng, K.-l. Tan, and J. Tang, "Online topic-aware influence maximization," *PVLDB*, vol. 8, no. 6, pp. 666–677, 2015.

[4] S. Feng, X. Chen, G. Cong, Y. Zeng, Y. M. Chee, and Y. Xiang, "Influence maximization with novelty decay in social networks." in *AAAI*, 2014.

[5] Z. Wang, E. Chen, Q. Liu, Y. Yang, Y. Ge, and B. Chang, "Maximizing the coverage of information propagation in social networks," in *IJCAI*, 2015.

[6] Y. Li, D. Zhang, and K.-L. Tan, "Real-time targeted influence maximization for online advertisements," *PVLDB*, vol. 8, no. 10, pp. 1070–1081, 2015.

[7] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in *SIGKDD*, 2009, pp. 199–208.

[8] Y. Tang, Y. Shi, and X. Xiao, "Influence maximization in near-linear time: A martingale approach," in *SIGMOD*, 2015, pp. 1539–1554.

[9] G. Tong, W. Wu, S. Tang, and D.-Z. Du, "Adaptive influence maximization in dynamic social networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 112–125, 2016.

[10] J. L. Z. Cai, M. Yan, and Y. Li, "Using crowdsourced data in location-based social networks to explore influence maximization," in *INFOCOM*, 2016, pp. 1–9.

[11] J. Li, T. Sellis, J. S. Culpepper, Z. He, C. Liu, and J. Wang, "Geo-social influence spanning maximization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 8, pp. 1653–1666, 2017.

[12] X. Wang, Y. Zhang, W. Zhang, and X. Lin, "Efficient distance-aware influence maximization in geo-social networks," *TKDE*, vol. 29, no. 3, pp. 599–612, 2016.

[13] G. Li, S. Chen, J. Feng, K.-l. Tan, and W.-s. Li, "Efficient location-aware influence maximization," in *SIGMOD*, 2014, pp. 87–98.

[14] X. Wang, Y. Zhang, W. Zhang, and X. Lin, "Distance-aware influence maximization in geo-social network," in *ICDE*, 2016, pp. 1–12.

[15] 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 *SIGKDD*, 2015, pp. 1573–1582.

[16] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: user movement in location-based social networks," in *SIGKDD*, 2011, pp. 1082–1090.

[17] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *SIGKDD*, vol. 96, no. 34, 1996, pp. 226–231.

[18] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann, "Time-aware point-of-interest recommendation," in *SIGIR*, 2013, pp. 363–372.

[19] Z. Yao, Y. Fu, B. Liu, Y. Liu, and H. Xiong, "Poi recommendation: A temporal matching between poi popularity and user regularity," in *ICDM*, 2016, pp. 549–558.

[20] W. Wang, M. Tang, H.-F. Zhang, H. Gao, Y. Do, and Z.-H. Liu, "Epidemic spreading on complex networks with general degree and weight distributions," *Physical Review E*, vol. 90, no. 4, p. 042803, 2014.

[21] T. Zhou, J. Cao, B. Liu, S. Xu, Z. Zhu, and J. Luo, "Location-based influence maximization in social networks," in *CIKM*, 2015, pp. 1211–1220.

[22] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *SIGKDD*, 2007, pp. 420–429.

[23] A. Likhyani, S. Bedathur, and P. Deepak, "Locate: Influence quantification for location promotion in location-based social networks." in *IJCAI*, 2017, pp. 2259–2265.

[24] Y. Tang, X. Xiao, and Y. Shi, "Influence maximization: Near-optimal time complexity meets practical efficiency," in *SIGMOD*, 2014, pp. 75–86.

[25] A. Goyal, W. Lu, and L. V. Lakshmanan, "Celf++ optimizing the greedy algorithm for influence maximization in social networks," in *WWW*, 2011, pp. 47–48.

[26] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*. SIAM, 2014, pp. 946–957.

[27] J. Li, T. Cai, K. Deng, X. Wang, T. Sellis, and F. Xia, "Community-diversified influence maximization in social networks," *Information Systems*, p. 101522, 2020.

[28] N. Alduaiji, A. Datta, and J. Li, "Influence propagation model for clique-based community detection in social networks," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 2, pp. 563–575, 2018.

[29] Q. Guo, S. Wang, Z. Wei, and M. Chen, "Influence maximization revisited: Efficient reverse reachable set generation with bound tightened," in *SIGMOD*, 2020, pp. 2167–2181.

[30] L. Sun, A. Chen, P. S. Yu, and W. Chen, "Influence maximization with spontaneous user adoption," in *WSDM*, 2020, pp. 573–581.

[31] S. Huang, Z. Bao, J. S. Culpepper, and B. Zhang, "Finding temporal influential users over evolving social networks," in *ICDE*, 2019, pp. 398–409.

[32] S. Bian, Q. Guo, S. Wang, and J. X. Yu, "Efficient algorithms for budgeted influence maximization on massive social networks," *PVLDB*, vol. 13, no. 9, pp. 1498–1510, 2020.

[33] C. Zhang, L. Shou, K. Chen, G. Chen, and Y. Bei, "Evaluating geo-social influence in location-based social networks," in *CIKM*, 2012, pp. 1442–1451.

[34] H. Gao, J. Tang, X. Hu, and H. Liu, "Modeling temporal effects of human mobile behavior on location-based social networks," in *CIKM*, 2013, pp. 1673–1678.

[35] M. Lichman and P. Smyth, "Modeling human location data with mixtures of kernel densities," in *SIGKDD*, 2014, pp. 35–44.

[36] L. Guo, D. Zhang, G. Cong, W. Wu, and K.-L. Tan, "Influence maximization in trajectory databases," *TKDE*, vol. 29, no. 3, pp. 627–641, 2016.

[37] T. Cai, J. Li, A. S. Mian, T. Sellis, J. X. Yu *et al.*, "Target-aware holistic influence maximization in spatial social networks," *TKDE*, 2020.

Xuanhao Chen received the Bachelor degree in Computer Science and Technology from Chongqing University of Posts and Telecommunications, in 2016. He is currently a PhD student in University of Electronic Science and Technology of China. His research interests include social networks and recommendation.

Yan Zhao received the PhD degree in Computer Science from Soochow University, in 2020. She in currently an Assistant Professor in Aalborg University. Her research interests include spatial database and trajectory computing.

Guanfeng Liu is current a Lecturer in the Department of Computing at Macquarie University, Australia. He received his Ph.D degree in Computer Science from Macquarie University, Australia in 2013. His research interests include graph data management, trust computing and social networks. He has published over 60 papers in the most prestigious journals and conferences such as IJCAI, AAAI, ICDE, CIKM, TKDE, TSC and ICWS.

Rui Sun received the Bachelor degree in Computer Science and Technology from Chongqing University of Posts and Telecommunications, in 2018. He is currently a Master student in University of Electronic Science and Technology of China. His research interests include natural language processing and recommendation.

Xiaofang Zhou is a professor of computer science with the University of Queensland. He is the head of the Data and Knowledge Engineering Research Division. He is a specially appointed adjunct professor under the Chinese National Qianren Scheme hosted by the Renmin University of China (2010-2013), and by Soochow University since July 2013 where he leads the Research Center on Advanced Data Analytics (ADA). He has been working in the area of spatial and multimedia databases, data quality, high performance query processing, Web information systems, and bioinformatics, and co-authored more than 250 research papers with many published in top journals and conferences. He is a fellow of the IEEE.

Kai Zheng is a Professor of Computer Science with University of Electronic Science and Technology of China. He received his PhD degree in Computer Science from The University of Queensland in 2012. He has been working in the area of spatial-temporal databases, uncertain databases, social-media analysis, in-memory computing and blockchain technologies. He has published over 100 papers in prestigious journals and conferences in data management field such as SIGMOD, ICDE, VLDB Journal, ACM Transactions and IEEE Transactions. He is a member of IEEE.