Geo-Social Keyword Search *

Ritesh Ahuja, Nikos Armenatzoglou, Dimitris Papadias, and George J. Fakas

Department of Computer Science and Engineering,
Hong Kong University of Science and Technology
{rahuja, nikos, dimitris, gfakas}@cs.ust.hk

Abstract. In this paper, we propose Geo-Social Keyword (GSK) search, which enables the retrieval of users, points of interest (POIs), or keywords that satisfy geographic, social, and/or textual criteria. We first introduce a general GSK framework that covers a wide range of real-world tasks, including advertisement, context-based search, and market analysis. Then, we present three concrete GSK queries: (i) NPRU that returns the top-k users based on their spatial proximity to a given query location, their popularity, and their similarity to an input set of terms; (ii) NSTP that outputs the top-k POIs based on their proximity to a user v, the number of check-ins by friends of v, and their similarity to a set of terms; (iii) FSKR that discovers the top-k keywords based on their frequency in pairs of friends located within a spatial area. For each query, we develop a processing algorithm that utilizes a novel hybrid index. Finally, we evaluate our framework with thorough experiments using real datasets.

1 Introduction

The rising popularity of social networks and smart-phones has led to the development of techniques for personalized search and targeted advertisement that combine social, geographic and textual criteria. As an instance of social and textual fusion, social networks, such as Facebook, permit the promotion of products to connected users that share common interests, e.g. the advertisement of a rock festival to a group of friends that like rock music [1]. As an example of geographic and textual integration, Web search engines, such as Google, allow search for Points Of Interest (POIs) that match some description and are near the query location , e.g., ”Chinese restaurants nearby” [2]. Finally, Geo-Social Networks (GeoSNs), such as Foursquare, combine geographic and social aspects by enabling users to check-in at POIs, i.e., publish their current location to friends. Moreover, advertisers can send GroupON-like offers to users in their vicinity to attract them, as well as their friends [3].

Similar combinations of social, geographic and textual criteria have been investigated in the research literature. i) *Keyword search in social networks* focuses of queries that seek groups of users forming a particular social structure (e.g. clique), and their members' profiles cover a set of input terms [16, 13, 14]. ii) *Spatial keyword search* queries return POIs that satisfy various spatial (e.g., range, nearest neighbor) and textual (e.g., text similarity) constraints [24, 20, 11, 10, 7, 18]. iii) *GeoSN queries* output

* Supported by GRF grant 617412 from Hong Kong RGC.

individual users, or groups of friends, that exhibit some spatial and social properties, e.g., the closest clique of m friends to a query point [22, 5, 17, 19].

All the above cases consider only two out of the three criteria, focusing on a single output type (e.g., users or POIs, but not both). On the other hand, we introduce *Geo-Social Keyword* (GSK) search, a class of top-k queries that combine all spatial, social, and textual attributes, and may return users, POIs or keywords. We present three concrete GSK queries: i) *Top-k Nearest, Popular and Relevant Users* (NPRU) that, given a query location q and a set of terms T_q, outputs the top-k users based on their proximity to q, their social connectivity, and the similarity of their profiles to T_q; ii) *Top-k Nearest Socially and Textually Relevant POIs* (NSTP), which, given a user v and a set of terms T_q, returns the top-k POIs based on their proximity to v, the number of check-ins by friends of v, and their similarity to T_q; and iii) *Top-k Frequent Social Keywords in Range* (FSKR) that discovers the top-k keywords based on their frequency in pairs of friends located within a geographic area.

Each query is suitable for a different type of task, including advertisement, context-based search, and market analysis. For instance, NPRU could be used by a restaurant to send promotions to nearby users, who are well-connected and have expressed interest in its cuisine type. Conversely, a user could issue an NSTP query to locate nearby restaurants of a specific type that are 'liked' by his friends. Finally, FSKR could identify trends or word-of-mouth effects in a geographic area, using the frequency of keywords shared by friends.

For each query, we provide a query processing algorithm that utilizes the *GSK Index* (GSKI), a novel hybrid structure that stores users and POIs, based on spatial, social, and textual attributes. GSKI is a lightweight multi-level grid that supports efficient updates. Summarizing, our contributions are:

- We define GSK search as a general framework for retrieval of the top-k users, POIs or keywords using various types of criteria.
- We present the GSKI, a hybrid structure for indexing users and POIs.
- We propose three GSK queries and the respective processing algorithms that utilize the GSKI.
- We conduct a thorough experimental evaluation on real datasets.

The rest of the paper is organized as follows. Section 2 overviews related work. Section 3 formalizes the GSK problem and introduces the general framework. Section 4 presents the GSK Index. Sections 5 to 7 propose the GSK queries and the corresponding query processing methods. Section 8 contains the experimental evaluation. Finally, Section 9 concludes the paper with directions for future work.

2 Related Work

We overview (i) keyword search in social networks, (ii) spatial keyword search, and (iii) GeoSN queries.

Keyword search in social networks. Although, there has been extensive work on keyword search for general graphs, here we focus on social networks. Lappas et al. [16] propose the *Team Formation* (*TF*) query: given a weighted social graph and a set of terms T_q, *TF* returns a subgraph of users, whose textual descriptions cover T_q and their diameter (i.e., maximum shortest-path distance between any two nodes) is minimized. The authors also devise a variant, where the subgraph must be a minimum spanning tree, and show that both problems are NP-Complete. [13] extends *TF* by additionally seeking a team leader, i.e., the member of the resulting group with the minimum total social shortest-path distances from all members. Finally, [14] proposes the r-cliques query: given a weighted social graph and a set of terms T_q, return a sugbraph of users that covers T_q, and has diameter no larger than r. In the above methods, textual information is stored in inverted files and the graph is kept in adjacency lists.

Spatial keyword search. Four types of spatial-keyword queries have received particular attention in the literature [8] namely, the *Boolean Range* (*BR*), the *Boolean k-NN* (*BkNN*), the *Spatial Aware Top-k text retrieval* (*SATopk*), and the *Spatial Group Keyword* (*SGK*) query. Given a spatial region R and a set of terms T_q, *BR* returns all POIs in R, whose textual description contains all terms in T_q [24, 20]. *BkNN* outputs the k nearest POIs to a query point q each of which covers all the query terms [11]. Given q, T_q and a positive integer k, *SATopk* returns a list of k POIs ranked based on their spatial proximity to q and textual similarity to T_q [10]. Finally, *SGK* discovers a set of POIs that collectively cover the query terms and either the sum of their distances to the query location is minimized [7], or the maximum distance between any two POIs in the group is minimized [18]. A recent work [21] introduces the *Social-aware top-k Spatial Keyword* (*SkSK*) query, which enhances personalized spatial-keyword search by additionally taking into consideration the social connectivity of the query issuer to all users, who have liked or recommended the POIs.

Spatial-keyword indices can be broadly classified according to the spatial and textual structures employed. They are usually based on the R-Tree and its variants, where each minimum bounding rectangle (MBR) keeps the textual information of the POIs located within its bounds. Specifically, MBRs in [10, 7] utilize inverted files, while in [11, 23] use bitmaps. Grid-based spatial-keyword structures decompose the space into cells; each cell has a unique id according to a global order (e.g., Hilbert curves [9]). Then, inverted files are primarily used for indexing the cells based on the textual description of the POIs located within their bounds [15, 20]. Indices based on trees are in general more efficient than grid-based structures [24], but the latter are easier to maintain. The *Social Network-aware IR-Tree* [21] is an R-Tree, where each node also contains a set of users relevant to the POIs indexed by the subtree rooted at the node; contrary to its name, it does not index social information (i.e., user connections).

Geo-Social Networks. GeoSN queries return users, or groups of users, that satisfy spatial and social criteria. Given a location q and two positive integers k and m ($k < m$), the *Socio-Spatial Group* query outputs a group of m users, such that the total distance of the users to q is minimized, and each user is connected to at least $m - k$ other group members [22]. Given a location q and two positive integers m, k, the *Nearest Star Group* query [5] returns the k nearest subgraphs of m users, such that each subgraph (i.e., star) has a user, who is socially connected to all users. Given a user v, the *k-Geo-Social Circle of Friends* query [17] finds a group of $k + 1$ users that contains v and k friends with small pairwise social distances, so that the diameter of the group is minimized. Finally,

[19] introduces the *Social and Spatial Ranking* query, which given a user v, reports the top-k users based on their spatial proximity and social connectivity to v.

Most GeoSN approaches maintain separate structures for the spatial and social attributes. For instance, Liu et al. [17] store the social graph in an adjacency matrix, and employ the R*-Tree for spatial indexing. Similarly, [5] uses adjacency lists and a regular spatial grid, respectively. On the other hand, Yang et al. [22] propose a hybrid index that constructs an R-tree while ensuring a specified degree of connectivity among the users within the same node.

3 GSK Query Framework

Our setting consists of a social graph network and a set of POIs. The social network is modeled as an unweighted, undirected graph $G = (V, E)$, where a node $v \in V$ represents a user and an edge $(v, u) \in E$ indicates the friendship between v and $u \in V$. Each user $v \in V$ may be associated with textual and spatial information that represent his preferences and his most recent location, respectively. Each POI $p \in P$ has a spatial location, a textual description and a set of users V_p that have checked-in at p in the past. T denotes a set of terms/keywords; specifically, T_v (resp. T_p) is the set that appears in the preference of user v (resp. the description of POI p).

Figure 1 depicts a running example of a social network with the locations of 10 users as grey points, and the incident edges as their social relations. The black squares represent the location of 4 POIs. Next to each user v and POI p is the corresponding set of terms T_v and T_p, e.g., $\{c, f\}$ for v_4 and $\{c, e\}$ for p_1. Moreover, the list below each POI (e.g., $[v_2, v_4, v_5, v_6]$ for p_1) represents the users that have checked-in there. Depending on the application, the setting may vary; e.g., the textual information of users may correspond to their query history or profile data (instead of preferences), V_p may denote the current (instead of all) check-ins at p, etc.

Geo-Social Keyword (GSK) search constitutes a family of top-k queries that return results of type $RT = (C, l)$, where C denotes the object class (i.e., V, P or T) and l represents the cardinality. For example, $RT = (V, 3)$ denotes that the output contains k groups of 3 users each, whereas $RT = (P, 1)$ signifies that the output consists of k individual POIs. Given a GSK query q, each object o of type RT (e.g., a group of 3 users, or a single POI) is assigned a geographic $f_g(o)$, social $f_s(o)$ and a textual $f_t(o)$ score. In general, $f_g(o)$ depends on the proximity of o to q, $f_s(o)$ on the social connectivity of o, and $f_t(o)$ on the similarity between the terms of o and q.

The total score of an object is obtained by combining the partial ones using a ranking function F. We implement F as a weighted combination of the partial scores, i.e., $F(o) = \alpha_g \cdot f_g(o) + \alpha_s \cdot f_s(o) + \alpha_t \cdot f_t(o)$, where $\alpha_g, \alpha_s, \alpha_t$ are non-negative real numbers such that $\alpha_g + \alpha_s + \alpha_t = 1$, but any monotone[1] function can be used. A criterion (e.g., textual) can be omitted by setting the corresponding weight (e.g., α_t) to zero. Moreover, in some cases we may only be interested in objects that satisfy a set of constraints CN, i.e., POIs in a geographic area, or users who have certain characteristics (e.g., males above 30 years old). Finally, we define a GSK query as follows:

[1] F should satisfy the condition $\forall o, o' : f_g(o) \geq f_g(o') \wedge f_s(o) \geq f_s(o') \wedge f_t(o) \geq f_t(o') \Rightarrow F(o) \geq F(o')$.

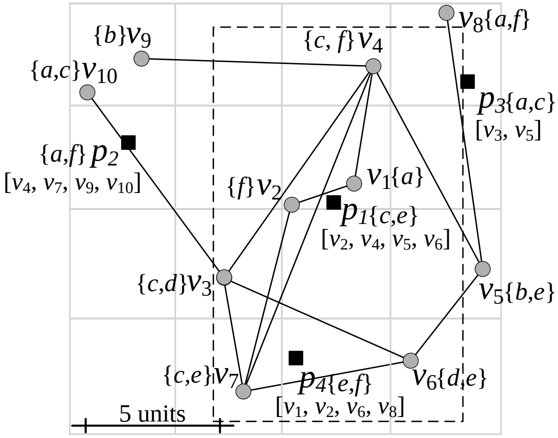


Fig. 1. Running Example

Definition 1. *Given a positive integer k, a result type RT, functions f_g, f_s, f_t, F, and a set of constraints CN, a GSK query returns the k objects of type RT that have the highest scores according to F and satisfy all constraints in CN.*

By employing different combinations of result types, ranking functions and constraints, we can devise a wide range of GSK queries. In this paper, we will present three diverse queries that retrieve individual users, POIs and keywords. All the queries utilize the index of the next section. Table 1 contains the frequent symbols.

Table 1. Basic notations

Notation	**Definition**
v	User in GeoSN, i.e., $v \in V$.
p	Point of interest, i.e., $p \in P$.
N_v	Friends of user v.
T_v (T_p)	Set of terms of user v (POI p).
T_q	Set of query terms.
V_p	Set of users that checked-in at p.
$\|v, q\|$	Euclidean distance of user v to point q. Similarly for p, i.e., $\|p, q\|$.
$\|c, q\|_{min}$	Minimum Euclidean distance of 2D rectangle c to 2D point q.
max_{dist}	Maximum possible Euclidean distance between any two points.
$deg(v)$	Number of v's friends, i.e, $\lvert N_v \rvert$ = $deg(v)$.
max_{deg}	Maximum number of friends in the graph.
$TS(T_1, T_2)$	Normalized textual similarity between term sets T_1 and T_2.

4 Geo-Social Keyword Index

The *Geo-Social Keyword Index* (GSKI) stores users and POIs based on their geographical, social and textual attributes. Given a granularity factor g and a height parameter h, GSKI partitions the geographical space into $g^h \times g^h$ equally sized leaf cells. Each leaf cell lc contains:

– a rectangle R_{lc} that represents the area covered by lc,
– a list of users V_{lc} and a list of POIs P_{lc} that lie in R_{lc},
– the maximal degree D_{lc} of any user in R_{lc},
– inverted files IV_{lc} and IP_{lc}, consisting of lists of keywords appearing in the preferences of users and in the descriptions of POIs in R_{lc}, respectively. Lists are sorted by the *impact* of keywords based on the *cosine-normalized tf-idf* [25], and
– a bloom filter[2] B_{lc} of the union of all users checked-in at POIs in R_{lc}, i.e $B_{lc} =$ bloom filter of $\bigcup_{p \in P_{lc}} V_p$.

Next, a hierarchical grid of height h is constructed in a bottom-up fashion, where each intermediate cell points to g^2 cells at the lower level that lie inside its spatial extent. Every intermediate cell ic keeps only a small amount of information summarizing its children cells. Specifically, ic is associated with a rectangle R_{ic}, maximum degree of users in R_{ic}, namely D_{ic}, and bloom filter B_{ic}. Additionally, for each term that appears in users or POIs located within the bounds of R_{ic}, ic keeps the term's maximum textual impact in sets SV_{ic} and SP_{ic}, respectively.

Figure 2 illustrates the GSKI and Table 2 shows the corresponding cell contents for our running example, assuming $g = 2$ and $h = 2$. Leaf cell $C0_{2,2}$ is a child of $C1_{1,1}$, which in turn is a child of $C2_{0,0}$. $C0_{2,2}$ contains users v_1, v_2 and POI p_1 in its spatial extent. Consequently, as elaborated in the fourth to last row of Table 2, $D_{C0_{2,2}} = 2 = deg(v_1) = deg(v_2)$, $IV_{C0_{2,2}}$ stores terms a, f, since they appear in v_1's and v_2's preferences, and $IP_{C0_{2,2}}$ keeps terms c, e occurring in p_1's description. Each term is associated with an *impact* value [25] in the range [0,1]. $B_{C0_{2,2}}$ contains users v_2, v_4, v_5, v_6 who checked-in at p_1. Intermediate cell $C1_{1,1}$ aggregates the information of its children $C0_{2,2}$, $C0_{2,3}$, $C0_{3,2}$, and $C0_{3,3}$. $D_{C1_{1,1}} = 5 = deg(v_4)$, since v_4 is located in $C0_{2,3}$. $C1_{1,1}$ keeps $SV_{C1_{1,1}}$ and $SP_{C1_{1,1}}$ with the terms that appear in the children, namely $\{a, c, f\}$ and $\{a, c, e\}$, respectively. Finally, $B_{C1_{1,1}}$ contains the union of $B_{C0_{2,2}}$ and $B_{C0_{3,3}}$ ($C0_{2,3}$ and $C0_{3,2}$ do not contain POIs).

To enable effective pruning during query processing, the GSKI preserves monotonicity across the height of the hierarchical grid, i.e., assuming a monotone function F, the overall score of an intermediate cell ic constitutes an upper bound for the score of any user or a POI within R_{ic}. Moreover, since the GSKI only keeps concise aggregated data at the intermediate levels, the size of the inverted file at a non-leaf cells is smaller than that of the original inverted file. Finally, we chose a grid-based structure because grids in general are usually significantly faster that R-trees for highly dynamic settings [12] such as ours, where there are numerous location updates from users.

[2] A bloom filter is a space-efficient probabilistic data structure that is used to test whether an element is a member of a set [6].

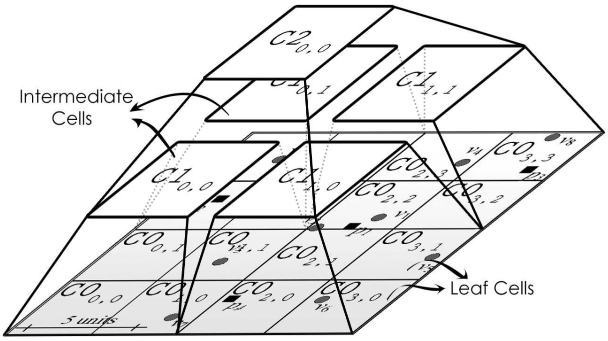


Fig. 2. Hierarchical Grid

Table 2. GSKI Contents

Cell c	IV_c / SV_c	IP_c / SP_c	D_c	B_c
$C2_{0,0}$				
$C1_{0,0}$	$\langle c, 0.71\rangle, \langle d, 0.71\rangle, \langle e, 0.71\rangle$		4	
$C0_{0,0}$			0	
$C0_{1,0}$	$c\colon \langle v_7, 0.71\rangle, e\colon \langle v_7, 0.71\rangle$		4	
$C0_{0,1}$			0	
$C0_{1,1}$	$c\colon \langle v_3, 0.71\rangle, d\colon \langle v_3, 0.71\rangle$		4	
$C1_{1,0}$	$\langle b, 0.71\rangle, \langle d, 0.71\rangle, \langle e, 0.71\rangle$	$\langle e, 0.71\rangle, \langle f, 0.71\rangle$	3	$\{v_1, v_6, v_8, v_9\}$
$C0_{2,0}$		$e\colon \langle p_4, 0.71\rangle, f\colon \langle p_4, 0.71\rangle$	0	$\{v_1, v_2, v_6, v_8\}$
$C0_{3,0}$	$d\colon \langle v_6, 0.71\rangle, e\colon \langle v_6, 0.71\rangle$		3	
$C0_{2,1}$			0	
$C0_{3,1}$	$b\colon \langle v_5, 0.71\rangle, e\colon \langle v_5, 0.71\rangle$		3	
$C1_{0,1}$	$\langle a, 0.71\rangle, \langle b, 1.0\rangle, \langle c, 0.71\rangle$	$\langle a, 0.71\rangle, \langle f, 0.71\rangle$	1	$\{v_4, v_7, v_9, v_{10}\}$
$C0_{0,2}$		$a\colon \langle p_2, 0.71\rangle, f\colon \langle p_2, 0.71\rangle$	0	$\{v_4, v_7, v_9, v_{10}\}$
$C0_{1,2}$			0	
$C0_{0,3}$	$a\colon \langle v_{10}, 0.71\rangle, b\colon \langle v_9, 1.0\rangle, c\colon \langle v_{10}, 0.71\rangle$		1	
$C0_{1,3}$			0	
$C1_{1,1}$	$\langle a, 1.0\rangle, \langle c, 0.71\rangle, \langle f, 1.0\rangle$	$\langle a, 0.71\rangle, \langle c, 0.71\rangle, \langle e, 0.71\rangle$	5	$\{v_2, v_3, v_4, v_5, v_6\}$
$C0_{2,2}$	$a\colon \langle v_1, 1.0\rangle, f\colon \langle v_2, 1.0\rangle$	$c\colon \langle p_1, 0.71\rangle, e\colon \langle p_1, 0.71\rangle$	2	$\{v_2, v_4, v_5, v_6\}$
$C0_{3,2}$			0	
$C0_{2,3}$	$c\colon \langle v_4, 0.71\rangle, f\colon \langle v_4, 0.71\rangle$		5	
$C0_{3,3}$	$a\colon \langle v_8, 0.71\rangle, f\colon \langle v_8, 0.71\rangle$	$a\colon \langle p_3, 0.71\rangle, c\colon \langle p_3, 0.71\rangle$	1	$\{v_3, v_5\}$

5 Top-k Nearest, Popular and Relevant Users

A *Top-k Nearest, Popular and Relevant Users* (NPRU) query returns the top-k users based on their spatial proximity to a location q, their social connectivity, and their textual similarity to an input set of terms T_q. NPRU is useful for advertisement and promotion purposes. For instance, consider a restaurant owner who wishes to send lunch coupons. Promising targets are users that (i) are near the restaurant, (ii) are highly connected, and (iii) express preference to the restaurant's type of food.

In our framework, the output type of NPRU is $RT = (V, 1)$, i.e., the result consists of individual users, and $CN = \emptyset$, i.e., there are no constraints on the users to be retrieved. Regarding the geographic $f_g(v)$, social $f_s(v)$ and textual $f_t(v)$ scores of each user $v \in V$, there are several alternatives. In our implementation, we set $f_g(v) = 1 - \frac{\|v,q\|}{max_{dist}}$, where max_{dist} denotes the maximum Euclidean distance in the data space. Intuitively, the spatial score of a user v decreases as his Euclidean distance $\|v, q\|$ from q increases. The social score of v is defined as $f_s(v) = \frac{deg(v)}{max_{deg}}$, where $deg(v)$ is the number of v's friends, and max_{deg} is the maximum degree of any user in the network. The textual score $f_t(v)$ is the *cosine-normalized tf-idf* similarity $TS(T_v, T_q)$ [25] between the terms T_v of v and those in T_q. All partial scores are in the range [0,1]. The total score of v is $F(v) = \alpha_g \cdot f_g(v) + \alpha_s \cdot f_s(v) + \alpha_t \cdot f_t(v)$, as discussed in Section 3.

Consider, for instance an NPRU query with $k = 2$, $q = p_1$, $T_q = \{c, e\}$ and $\alpha_g = \alpha_s = \alpha_t = \frac{1}{3}$ in the running example of Figure 1, e.g., a Chinese restaurant p_1 wishes to discover the top-2 users in its vicinity, that have many friends and at the same time have matching keywords c, e (Chinese, Restaurant). The best user is u_7 because both keywords c and e are in his preferences. The top-2 user is v_4 with keyword c. Note that v_4 out-ranks v_3, which is slightly closer to p_1 and contains c, because he has higher degree (5 as opposed to 4 for v_3). Although users v_1 and v_2 are the nearest to p_1, they are not in the result because neither contains keyword c or e; accordingly, their f_t score is zero.

Processing NPRU queries is based on the branch-and-bound paradigm using the GSKI. Specifically, a priority heap H maintains visited cells and users along with their score according to F. The score of a cell c takes into consideration (i) the minimum Euclidean distance of the cell to q, (ii) the maximum degree of any user in c, and (iii) the maximum textual similarity of the queried terms amongst the preferences of the users in c. This guarantees that the score of c is an upper bound for the score of child cells and users within its extent. Consequently, if the score of c does not exceed that of the top-kth user, then c can be safely pruned.

Figure 3 illustrates the pseudo-code of NPRU processing. Initially, the algorithm adds GSKI's root cell to H (Line 2). Then, in an iterative manner, it removes the entity with the highest score from H, namely e, and i) if e is an intermediate cell, then it adds all its children cells to H (Lines 5-7), or ii) if e is a leaf cell, then it adds all users within e's spatial extent to H (Lines 8-10), or iii) if e is a user, it adds him to the result set (Lines 11-12). The algorithm terminates when the result set contains k users (Lines 13-14). The cells and users remaining in H have score at most as high as that of the k-th result and, hence, can be ignored.

Table 3 shows the heap state during the execution of the example query: $k = 2$, $q = p_1$, $T_q = \{c, e\}$ and $\alpha_g = \alpha_s = \alpha_t = \frac{1}{3}$, using the GSKI contents of Table 2. Heap entries consist of a cell or a user, and the corresponding score according to F. Cells and users added to H are shown in bold. First, the algorithm inserts the root of GSKI in H. At iteration 1, it removes the root cell and adds its children along with their scores to H. Next, the intermediate cell with the highest score, $C1_{0,0}$, is removed and its child leaf cells $\{C0_{0,0}, C0_{1,0}, C0_{0,1}, C0_{1,1}\}$ are added to H. Similarly, $C0_{1,0}$ is removed at the next iteration and user v_7 is added to H. Next, intermediate cell $C1_{1,1}$ is de-heaped and its child leaf nodes are en-heaped. Then, user v_7 is removed and becomes the top-1 result. The algorithm continues in the same manner and terminates after the 6th iteration, when the top-2 user v_4 is de-heaped.

Input: Social Graph $G = (V, E)$, integer k, location q, set of terms T_q, weights $\alpha_g,\ \alpha_s,\ \alpha_t$
Output: Top-k users according to F

Define H as an empty heap of $GSKI$ cells sorted according to their scores in decr. order
Add the root cell of $GSKI$ to H
While H is not empty
    e = top entity of H // it also removes e from H
    If e is an intermediate cell of $GSKI$
        For each child c of e
            Add to H cell c with score $\alpha_g \cdot (1 - \frac{\|c,q\|_{min}}{max_{dist}}) + \alpha_s \cdot \frac{D_c}{max_{deg}} + \alpha_t \cdot TS(T_c, T_q)$
    Else If e is a leaf cell of $GSKI$
        For each user $v \in V_e$
            Add to H user v with score $\alpha_g \cdot (1 - \frac{\|v,q\|}{max_{dist}}) + \alpha_s \cdot \frac{deg(v)}{max_{deg}} + \alpha_t \cdot TS(T_v, T_q)$
    Else // e is a user
        Add e to R
        If $|R| = k$ then stop the execution
Return R

Fig. 3. NPRU Algorithm

Table 3. Heap of NPRU

Interation #	Heap Contents
0	$\langle C2_{0,0}, \infty\rangle$
1	$\langle \mathbf{C1_{0,0}, 0.90}\rangle$, $\langle \mathbf{C1_{1,1}, 0.81}\rangle$, $\langle \mathbf{C1_{1,0}, 0.71}\rangle$,$\langle \mathbf{C1_{0,1}, 0.51}\rangle$
2	$\langle \mathbf{C0_{1,0}, 0.85}\rangle$, $\langle C1_{1,1}, 0.81\rangle$, $\langle C1_{1,0}, 0.71\rangle$, $\langle \mathbf{C0_{1,1}, 0.71}\rangle$, $\langle C1_{0,1}, 0.51\rangle$, $\langle \mathbf{C0_{0,1}, 0.24}\rangle$, $\langle \mathbf{C0_{0,0}, 0.21}\rangle$
3	$\langle C1_{1,1}, 0.82\rangle$, $\langle \mathbf{v_7, 0.80}\rangle$, $\langle C1_{1,0}, 0.71\rangle$, $\langle C0_{1,1}, 0.71\rangle$, $\langle C1_{0,1}, 0.51\rangle$, $\langle C0_{0,1}, 0.24\rangle$, $\langle C0_{0,0}, 0.21\rangle$
4	$\langle v_7, 0.80\rangle$, $\langle \mathbf{C0_{2,3}, 0.75}\rangle$, $\langle C1_{1,0}, 0.71\rangle$, $\langle C0_{1,1}, 0.71\rangle$, $\langle C1_{0,1}, 0.51\rangle$, $\langle \mathbf{C0_{2,2}, 0.46}\rangle$, $\langle \mathbf{C0_{3,3}, 0.33}\rangle$, $\langle \mathbf{C0_{3,2}, 0.30}\rangle$, $\langle C0_{0,1}, 0.24\rangle$, $\langle C0_{0,0}, 0.21\rangle$
5	$\langle C0_{2,3}, 0.75\rangle$, $\langle C1_{1,0}, 0.71\rangle$, $\langle C0_{1,1}, 0.71\rangle$, $\langle C1_{0,1}, 0.51\rangle$, $\langle C0_{2,2}, 0.46\rangle$, $\langle C0_{3,3}, 0.33\rangle$, $\langle C0_{3,2}, 0.30\rangle$, $\langle C0_{0,1}, 0.24\rangle$, $\langle C0_{0,0}, 0.21\rangle$
6	$\langle \mathbf{v_4, 0.72}\rangle$, $\langle C1_{1,0}, 0.71\rangle$, $\langle C0_{1,1}, 0.71\rangle$, $\langle C1_{0,1}, 0.51\rangle$, $\langle C0_{2,2}, 0.46\rangle$, $\langle C0_{3,3}, 0.33\rangle$, $\langle C0_{3,2}, 0.30\rangle$, $\langle C0_{0,1}, 0.24\rangle$, $\langle C0_{0,0}, 0.21\rangle$

6 Top-k Nearest Socially and Textually Relevant POIs

Given a user v and a set of terms T_q, a *Top-k Nearest Socially and Textually Relevant POIs* (NSTP) query returns the top-k POIs based on their proximity to v, the textual similarity of their descriptions to T_q, and the number of v's friends that checked-in. NSTP enables location-aware, socially-aware, and/or context-aware search. For instance, consider a user who wants to visit a restaurant. NSTP could locate nearby restaurants offering cuisine similar to the user's preferences that are also visited (or 'liked') by his friends.

The output type of NSTP query is $RT = (P, 1)$, i.e., the result consists of individual POIs, and $CN = \emptyset$, i.e., there are no constraints on the POIs to be retrieved[3]. The

[3] Additional constraints in this case could restrict the top-k POIs to be in a certain area, or enforce certain properties (e.g., restaurant must be open after 10pm).

geographic and textual score definitions are similar to NPRU, i.e., $f_g(p) = 1 - \frac{\|v,p\|}{max_{dist}}$ and $f_t(p)$ is based on *cosine-normalized tf-idf* between T_p and T_q. The social score is defined as $f_s(p) = \frac{|N_v \cap V_p|}{|N_v|}$, where set N_v consists of v's friends (i.e., $|N_v| = deg(v)$), and V_p contains the ids of the users who checked-in at p. The partial scores are combined by the linear function F also used in NPRU.

For example, consider an NSTP query with $v = v_7$, $k = 2$, $T_q = \{c, e\}$, and $\alpha_g = \alpha_s = \alpha_t = \frac{1}{3}$ using the running example, e.g., user v_7 searches for two nearby Chinese restaurants (c, e) that have been visited by many of his friends. The best POI is p_1 since it is relatively close to v_7, contains both queried terms, and it has been visited by 3 of his 4 friends (v_2, v_4, v_6). The top-2 POI is p_4 because it is the closest POI to v_7, contains term e, and was visited by two of v_7's friends (v_2, v_6). POIs p_2 and p_3 are not in the result set since they are far from v_7, are not relevant to T (only p_3 contains one of the queried terms), and are not popular among v_7's friends (each is visited by only one friend).

NSTP query processing is similar to NPRU. Specifically, the algorithm uses a max-heap to store cells and POIs sorted in decreasing order of their scores. The score of a cell c is based on: i) the minimum distance of c to v, ii) an upper bound for the number of v's friends that checked-in at any POI within c, and iii) the maximum textual similarity of T to the descriptions of the POIs in c. For the computation of (ii), the algorithm examines if each friend of v is in the bloom filter of c. Bloom filters may falsely indicate the presence of a user. However, although false positives increase the score of c, they do not affect correctness because the score of c is always an upper bound (albeit, in some cases, loose) for that of any child cell or POI in c. The algorithm terminates after it retrieves k POIs from the priority heap.

Consider again the example query with input: $v = v_7$, $k = 2$, $T_q = \{c, e\}$, and $\alpha_g = \alpha_s = \alpha_t = \frac{1}{3}$, using the GSKI contents of Table 2. Table 4 shows the state of the heap at each iteration. Starting from the root cell, the algorithm retrieves the top-1 POI p_1 at iteration 3. Then, it continues until iteration 6, when it discovers p_4 and terminates.

Table 4. Heap of NSTP

Interation #	**Heap H Contents**
0	$\langle C2_{0,0}, \infty\rangle$
1	$\boldsymbol{\langle C1_{1,1}, 0.75\rangle}$, $\boldsymbol{\langle C1_{1,0}, 0.55\rangle}$, $\boldsymbol{\langle C1_{0,0}, 0.33\rangle}$, $\boldsymbol{\langle C1_{0,1}, 0.28\rangle}$
2	$\boldsymbol{\langle C0_{2,2}, 0.75\rangle}$, $\langle C1_{1,0}, 0.55\rangle$, $\boldsymbol{\langle C0_{3,3}, 0.44\rangle}$, $\langle C1_{0,0}, 0.33\rangle$, $\langle C1_{0,1}, 0.28\rangle$, $\boldsymbol{\langle C0_{3,2}, 0.19\rangle}$, $\boldsymbol{\langle C0_{2,3}, 0.15\rangle}$
3	$\boldsymbol{\langle p_1, 0.75\rangle}$, $\langle C1_{1,0}, 0.55\rangle$, $\langle C0_{3,3}, 0.44\rangle$, $\langle C1_{0,0}, 0.33\rangle$, $\langle C1_{0,1}, 0.28\rangle$, $\langle C0_{3,2}, 0.19\rangle$, $\langle C0_{2,3}, 0.15\rangle$
4	$\langle C1_{1,0}, 0.55\rangle$, $\langle C0_{3,3}, 0.44\rangle$, $\langle C1_{0,0}, 0.33\rangle$, $\langle C1_{0,1}, 0.28\rangle$, $\langle C0_{3,2}, 0.19\rangle$, $\langle C0_{2,3}, 0.15\rangle$
5	$\boldsymbol{\langle C0_{2,0}, 0.55\rangle}$, $\langle C0_{3,3}, 0.44\rangle$, $\langle C1_{0,0}, 0.33\rangle$, $\langle C1_{0,1}, 0.28\rangle$, $\boldsymbol{\langle C0_{2,1}, 0.28\rangle}$, $\boldsymbol{\langle C0_{3,0}, 0.25\rangle}$, $\boldsymbol{\langle C0_{3,1}, 0.23\rangle}$, $\langle C0_{3,2}, 0.19\rangle$, $\langle C0_{2,3}, 0.15\rangle$
6	$\boldsymbol{\langle p_4, 0.59\rangle}$, $\langle C0_{3,3}, 0.44\rangle$, $\langle C1_{0,0}, 0.33\rangle$, $\langle C1_{0,1}, 0.28\rangle$, $\langle C0_{2,1}, 0.28\rangle$, $\langle C0_{3,0}, 0.24\rangle$, $\langle C0_{3,1}, 0.23\rangle$, $\langle C0_{3,2}, 0.19\rangle$, $\langle C0_{2,3}, 0.15\rangle$

7 Frequent Social Keywords in Range

A *Frequent Social Keywords in Range* (FSKR) query returns the top-k terms based on their frequency in pairs of friends located within a spatial area SR. FSKR allows the discovery of trends or word-of-mouth effects. For instance, FSKR on textual content derived from Twitter/Facebook posts can reveal topics that are trending among friends in a geographic area. This information can be then utilized by businesses towards social media marketing.

The output of FSKR query is $RT = (T, 1)$, i.e., the result consists of individual terms. In addition, CN contains the constraint that valid terms must appear jointly in the preferences of friends in SR. FSKR does not apply geographic or social scores; instead, the total score of a term t is based solely on its frequency among friends, i.e., $F(t) = f_t(t) = |\{(v, u) \in E \;/\; t \in T_v \wedge t \in T_u \wedge v, u \; inside \; SR\}|$, where T_v (resp. T_u) denotes the terms associated with v (resp. u). Note that an edge (v, u) contributes 2 to the score of t; once per incident user v and u. This does not affect the ranking of the top-k results.

Consider, for instance, the FSKR query with $k = 2$ and an area SR represented by the dashed-line rectangle in Figure 1. The top-1 term is c, with score $F(c) = 6$, since it appears in 3 pairs of friends within the range, i.e., (v_3, v_4), (v_3, v_7), and (v_4, v_7). The top-2 term can be either e (v_6, v_7), or d (v_3, v_6), both with score 2. The remaining terms in SR (a, d, f) are not shared by any pair of friends.

FSKR query processing is performed in two steps: first, for every term t in SR, a list $PL[t]$ is created with the users (in SR) containing t; then, the score $F(t)$ of each term t is computed by examining the connections of users appearing in $PL[t]$. Specifically, the contribution of each $v \in PL[t]$ to $F(t)$ is $|N_v \cap PL[t]|$, where N_v is the set of v's friends. Let $best_{score}$ be the score of the current top-kth term. The upper bound score of any (not-yet-examined) term t is $|PL[t]| \cdot (|PL[t]| - 1)$, when all users containing t form a clique. Consequently, if $|PL[t]| \cdot (|PL[t]| - 1) \leq best_{score}$, then t can be safely pruned. Based on this observation, FSKR examines terms in decreasing order of their list sizes, until the first term that can be eliminated by its upper bound score.

Figure 4 elaborates the procedure. The algorithm first retrieves the non-empty leaf cells of GSKI that intersect with the spatial range SR. For each keyword t in the inverted lists of these cells, Lines 3-13 generate $PL[t]$. Next, the terms are sorted in decreasing order of $|PL[t]|$ size. For each term t, Lines 18-20, compute the score of t, and update $best_{score}$ accordingly. The algorithm terminates at the first term for which $|PL[t]| \cdot (|PL[t]| - 1) \leq best_{score}$ (Lines 16-17), and returns the top-k set (Line 21). Unexamined terms cannot be in the result set, and are pruned.

We describe the algorithm using our running example of Figure 1, where $k = 1$ and the spatial range SR is depicted as a dashed rectangle. Initially, $best_{score} = 0$. The terms associated with users in SR are a, c, d, e, f with lists $PL[a] = \{v_1\}$, $PL[c] = \{v_3, v_4, v_7\}$, $PL[d] = \{v_3, v_6\}$, $PL[e] = \{v_6, v_7\}$ and $PL[f] = \{v_2, v_4\}$. FSKR iterates over the lists in sorted order, starting from c. It computes $|PL[c] \cap N_{v_3}| = 2$, $|PL[c] \cap N_{v_4}| = 2$, $|PL[c] \cap N_{v_7}| = 2$, and $F(c) = 6$. Since $k = 1$, it sets $best_{score} = 6$ and retrieves the second most frequent keyword e. The upper bound score for e is 2, which is below $best_{score}$. Consequently, the algorithm stops and outputs c as the top-1 result.

Input: Social Graph $G = (V, E)$, integer k, spatial range SR
Output: Top-k terms according to F

Initialize list PL as an empty list of sets, $best_{score} = 0$
Set $C =$ all non-empty leaf cells in GSKI that intersect with SR
For each cell $c \in C$
    For each term $t \in IV_c$
        $Occur_t$ = posting list of t in IV_c
        If t appears for first time
            $PL[t] = \{\emptyset\}$
        Else
            If R covers c
                $PL[t] = PL[t] \cup Occur_t$
            Else
                $Occur_{t,valid} =$ Exclude from $Occur_t$ all users not in SR
                $PL[t] = PL[t] \cup Occur_{t,valid}$
Sort PL according to sets' sizes in decreasing order
For each term $t \in PL$
    If $|PL[t]| \cdot (|PL[t]| - 1) \leq best_{score}$
        Exit For Loop
    For each user $v \in PL[t]$
        $Score_t = Score_t + |N_v \cap PL[t]|$
    $best_{score} = k^{th}$ highest score
Return the terms with the k highest scores

Fig. 4. FSKR Algorithm

8 Experimental Evaluation

Section 8.1 presents the real datasets, Section 8.2 contains a qualitative evaluation of the proposed queries, and Section 8.3 evaluates their performance experimentally.

8.1 Datasets

We use two real datasets obtained from *Yelp* [4] that consist of users and POIs located in Las Vegas (LV) and Phoenix (PX). In particular, each dataset includes: i) a social graph, ii) latest and past user check-ins, iii) user preferences, iv) POI locations, and v) POI descriptions. Table 5 summarizes the characteristics of LV and PX. Note that LV contains more users in a smaller geographic area, whose distribution is skewed. Users and POIs in PX are distributed more uniformly.

8.2 Visualization

We qualitatively evaluate the proposed queries using LV. In the following visualizations, users and POIs are depicted as grey points and rectangles, respectively. Query points and top-k results are colored black, and each points to an information table that presents their parameters and partial scores.

Table 5. Datasets

Statistic	LV	PX
$\|V\|$	40,297	30,056
Avg. Degree	9.66	5.41
Max. Degree	2451	1246
Avg. $\|T_v\|$	161	166
$\|P\|$	12,773	16,154
Avg. $\|V_p\|$	14.98	8.89
Avg. $\|T_p\|$	5.35	9.7
Area	$37km \times 46km$	$71km \times 87km$
Max. Dist	$60km$	$112km$

Top-k Nearest, Popular and Relevant Users. Figure 5 illustrates the results of an NPRU query issued by a Mexican bar, where $T_q = \{mexican, alcohol, bar\}$, $k = 3$ and $\alpha_g = \alpha_s = \alpha_t = \frac{1}{3}$. The top-1 user is the closest to the query point, the most popular and the most relevant to T_q. Although the top-2 user is farther than top-3, he receives a better score because he has a higher degree and his preferences are more similar to T_q.

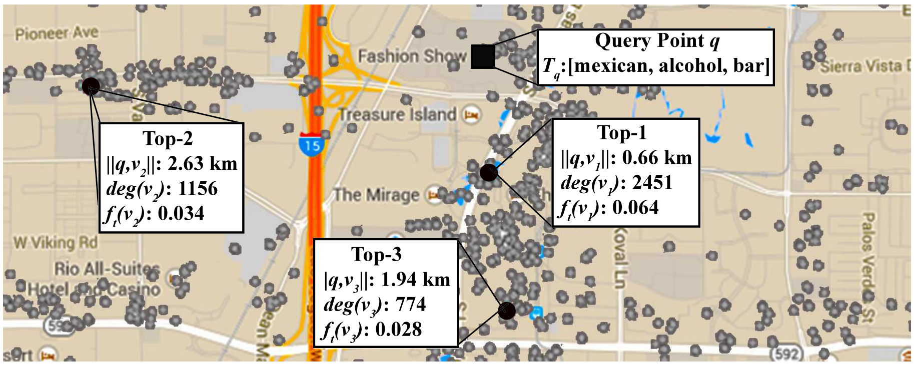


Fig. 5. Top-3 Users in NPRU

Top-k Nearest Socially and Textually Relevant POIs. Figure 6 depicts the results of an NSTP query issued by a user v, who searches for 3 nearby POIs that contain terms "*mexican, alcohol, bar*" and have been visited by his friends ($\alpha_g = \alpha_t = 0.25$ and $\alpha_s = 0.5$). The top-1 bar is 400 meters away from v, and has been visited by one of v's friends. The top-2 bar is $1.53km$ far from v, and has also been visited by one friend. Note that the top-3 bar has the highest textual similarity, but it is relatively far, and has not been visited by any of v's friends.

Frequent Social Keywords in Range. Figure 7 visualizes the results of an FSKR query, where a dashed-lined rectangle represents SR and $k = 1$. The top-1 keyword "*food*" is shared among 9 pairs of friends, connected by the bold edges. The remaining edges denote social connections of users in SR.

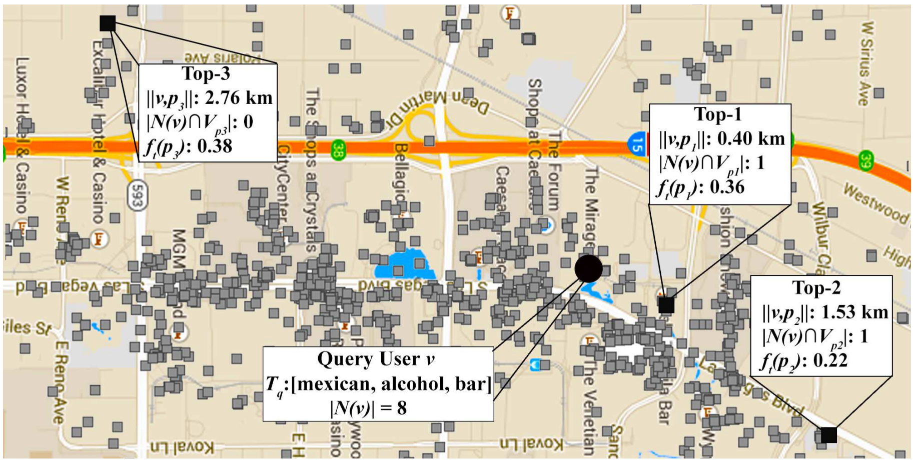


Fig. 6. Top-3 POIs in NSTP

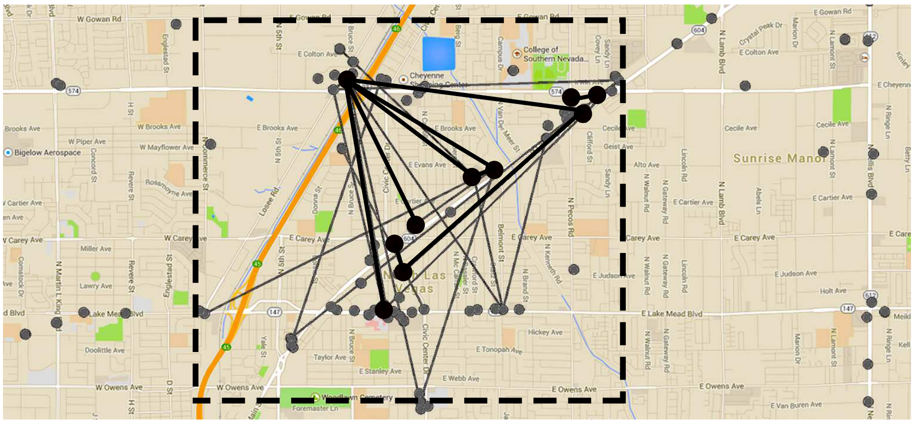


Fig. 7. Top-1 Keyword in FSKR

8.3 Performance

The query processing algorithms were implemented in C++ under Linux Ubuntu, and executed on an Intel Xeon E5-2660 2.20GHz with 8GB RAM. All data and indices are stored in the main memory. The social graph is kept as a collection of adjacency lists, one per user. The reported times are the average of 20 query executions for each of LV and PX. Table 6 includes the tested value ranges for the query and system parameters in our setup; r corresponds to the radius of the circular spatial range SR of FSKR.

Geo-Social Keyword Index. Figure 8 studies the effect of GSKI granularity g on the running time of NPRU, NSTP, and FSKR using LV, for $h = 4$, $k = 16$, $|T_q| = 3$, and $r = 3km$. For granularity up to 5, the running time of NPRU and NSTP decreases with g. Since the cells cover smaller areas, the aggregate information stored in the cells is more accurate, and thus the algorithms visit fewer cells. When the granularity exceeds

Table 6. Query and System Parameters

Parameter	Default	Range
k	16	4, 8, 16, 32, 64
$\|T_q\|$	3	1, 2, 3, 4, 5
g	5	3, 4, 5, 6
r (km)	3	1, 2, 3, 4, 5

5, the GSKI becomes less effective because the heaps in NPRU and NSTP maintain numerous cells, i.e., each intermediate cell has fanout 36. The execution time of FSKR increases slightly with g. Recall that the first step of FSKR creates the occurrence lists of terms in SR by merging the inverted files of the cells that intersect with SR. Consequently, the CPU time grows as the algorithm merges more inverted lists, but the impact is negligible. In the remaining experiments, we set $g = 5$ because it minimizes the execution time of NPRU, NSTP, and it marginally affects FSKR.

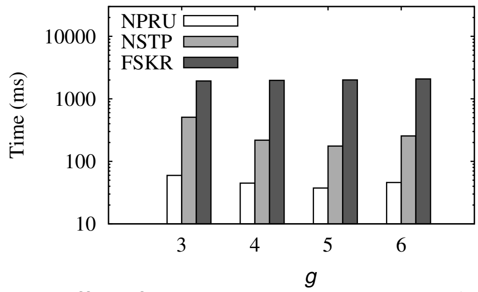


Fig. 8. Effect of GSKI Granularity (LV Dataset, $h = 4$)

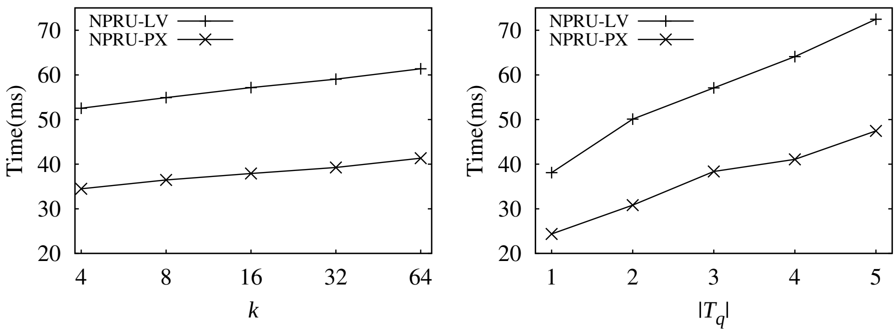
Table 7 assesses the total construction time of GSKI indices under the setup of Figure 8 in both datasets. In the most challenging setting, i.e., $g = 6$ and $h = 4$ (1.6M leaf cells), GSKI needs only 45 seconds for both datasets since it only keeps concise aggregated data at the intermediate levels.

Table 7. GSKI Construction Time

Granularity g	**Height h**	**# Leaf cells**	**LV Time** (sec)	**PX Time** (sec)
3	4	6561	10.2	8.7
4	4	65536	13.3	11.6
5	4	390625	16.6	14.7
6	4	1679616	23.7	21.3

Top-k Nearest, Popular and Relevant Users. Figure 9(a) presents the query time of NPRU as a function of the result size k in LV and PX, for $|T_q| = 3$. In both datasets, the cost increases with k because the algorithm retrieves more users from the priority heap, and thus performs more iterations. NPRU is faster in PX because it contains relatively few users, who are rather uniformly distributed. Therefore, the cells contain more accurate information that leads to better pruning.

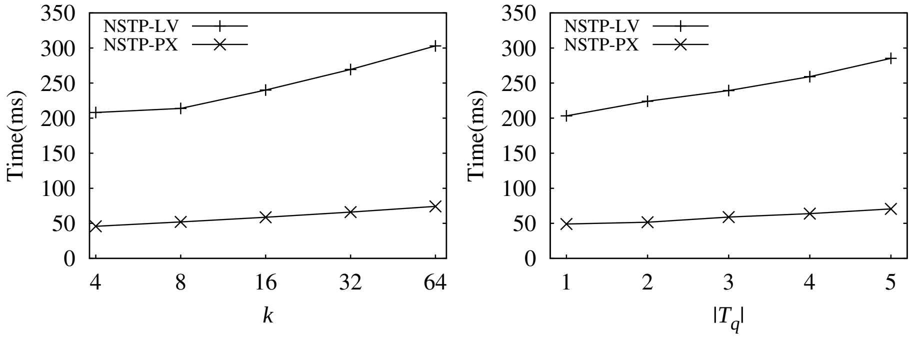
Figure 9(b) plots the running time versus the number of queried terms, i.e., $|T_q|$, for $k = 16$. In both datasets, the cost increases with $|T_q|$ as the algorithm requires more computations to calculate the textual similarity of each visited cell or user. In addition, when $|T_q|$ increases, more cells become textually relevant to the query, reducing the pruning power of the algorithm.



(a) Time vs. k (b) Time vs. $|T_q|$

Fig. 9. Query Time for NPRU

Top-k Nearest Socially and Textually Relevant POIs. Figure 10(a) shows the execution time of NSTP versus the result size k in LV and PX, for $|T_q| = 3$. Similar to NPRU, the running time increases with k since the algorithm executes more iterations. Compared to PX, the cost in LV increases faster because the distribution of POIs is highly skewed. This leads to inaccurate aggregate information at cells covering dense areas, burdening the reported average time. Figure 10(b) measures the running time as a function of $|T_q|$, for $k = 16$. The diagrams and the explanations are similar to those of Figure 9(b).

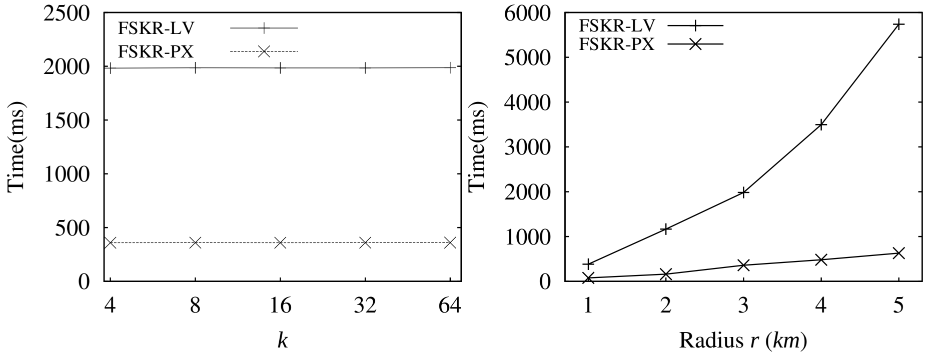


(a) Time vs. k (b) Time vs. $|T_q|$

Fig. 10. Query Time for NSTP

Frequent Social Keywords in Range. Figure 11(a) plots the running time of FSKR versus k, for $r = 3km$. Recall that FSKR initially creates the occurrence lists of the terms in SR by merging the inverted lists of the leaf cells that overlap SR. Then, the terms are sorted in decreasing order of list size. These steps dominate the total cost. Consequently, the value of k does not affect the execution time. FSKR is slower in LV since the average number of users in SR is greater, i.e., 2105 in LV and 464 in PX.

Figure 11(b) shows the execution time as a function of the radius r of SR, for $k = 16$. In both datasets the running time grows with r. In LV, the cost exhibits a steep increase because many new users are covered by the expanded SR. For instance, for $r = 4km$, SR includes on average 3662 users in LV and 627 in PX, whereas for $r = 5km$, it covers 6092 and 776 users, respectively.



(a) Time vs. k

(b) Time vs. Radius r (km) (SR)

Fig. 11. Query Time for FSKR

Summarizing the experimental evaluation, all algorithms are very fast (at most, a few seconds) under all settings. In addition, the construction of GSKI only takes up to 23 seconds for the selected g, h, and the largest dataset. Finally, the GSKI supports efficient location updates as it is based on a grid structure.

9 Conclusion

This paper introduces a class of top-k queries that enable retrieval of users, POIs or keywords based on geographic, social and textual criteria. We propose three concrete queries that can be used in various tasks involving context-based search, profile-based advertisement and market analysis. For each query we provide a processing algorithm that exploits a specialized index. Our experiments with real datasets confirm the effectiveness and efficiency of the proposed methods.

An interesting direction for future work concerns additional GSK queries, applicable to different tasks. Even the same queries can be altered to support alternative partial scores. For instance, instead of the Euclidean, we could apply the road network distance to the definition of geographic score in NPRU and NSTP. Similarly, FSKR could be based on co-occurrences of terms in triangles (instead of pairs) of friends.

References

1. Facebook ads, audience targeting. `https://www.facebook.com/help/229438340403916`.
2. Google mobile maps. `http://www.google.com/mobile/maps/`.
3. GroupOn Now! deals available on Foursquare. `https://blog.groupon.com/cities/groupon-now-deals-available-in-foursquare/`.
4. Yelp academic dataset. `http://www.yelp.com/dataset_challenge/`.
5. N. Armenatzoglou, S. Papadopoulos, and D. Papadias. A general framework for geo-social query processing. *Proc. VLDB Endow.*, 6(10):913–924, 2013.
6. B. H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of ACM*, 13(7):422–426, 1970.
7. X. Cao, G. Cong, C. S. Jensen, and B. C. Ooi. Collective spatial keyword querying. In *SIGMOD*, 2011.
8. L. Chen, G. Cong, C. S. Jensen, and D. Wu. Spatial keyword query processing: An experimental evaluation. *Proc. VLDB Endow.*, 6(3):217–228, 2013.
9. Y.-Y. Chen, T. Suel, and A. Markowetz. Efficient query processing in geographic web search engines. In *SIGMOD*, 2006.
10. G. Cong, C. S. Jensen, and D. Wu. Efficient retrieval of the top-k most relevant spatial web objects. *Proc. VLDB Endow.*, 2(1):337–348, 2009.
11. I. De Felipe, V. Hristidis, and N. Rishe. Keyword search on spatial databases. In *ICDE*, 2008.
12. D. V. Kalashnikov, S. Prabhakar, and S. E. Hambrusch. Main memory evaluation of monitoring queries over moving objects. *Distrib. Parallel Databases*, 15(2):117–135, 2004.
13. M. Kargar and A. An. Discovering top-k teams of experts with/without a leader in social networks. In *CIKM*, 2011.
14. M. Kargar and A. An. Keyword search in graphs: Finding r-cliques. *Proc. VLDB Endow.*, 4(10):681–692, 2011.
15. A. Khodaei, C. Shahabi, and C. Li. Hybrid indexing and seamless ranking of spatial and textual features of web documents. In *DEXA*, 2010.
16. T. Lappas, K. Liu, and E. Terzi. Finding a team of experts in social networks. In *SIGKDD*, 2009.
17. W. Liu, W. Sun, C. Chen, Y. Huang, Y. Jing, and K. Chen. Circle of friend query in geo-social networks. In *DASFAA*, 2012.
18. C. Long, R. C.-W. Wong, K. Wang, and A. W.-C. Fu. Collective spatial keyword queries: A distance owner-driven approach. In *SIGMOD*, 2013.
19. K. Mouratidis, J. Li, Y. Tang, and N. Mamoulis. Joint search by social and spatial proximity. *IEEE Transactions on Knowledge & Data Engineering*, 10(16):1169–1184, 2015.
20. S. Vaid, C. B. Jones, H. Joho, and M. Sanderson. Spatio-textual indexing for geographical search on the web. In *SSTD*, 2005.
21. D. Wu, Y. Li, B. Choi, and J. Xu. Social-aware top-k spatial keyword search. In *Mobile Data Management (MDM), 2014 IEEE 15th International Conference on*, volume 1, pages 235–244. IEEE, 2014.
22. D.-N. Yang, C.-Y. Shen, W.-C. Lee, and M.-S. Chen. On socio-spatial group query for location-based social networks. In *KDD*, 2012.
23. D. Zhang, Y. M. Chee, A. Mondal, A. Tung, and M. Kitsuregawa. Keyword search in spatial databases: Towards searching by document. In *ICDE*, 2009.
24. Y. Zhou, X. Xie, C. Wang, Y. Gong, and W.-Y. Ma. Hybrid index structures for location-based web search. In *CIKM*, 2005.
25. J. Zobel and A. Moffat. Inverted files for text search engines. *ACM Comput. Surv.*, 38(2), 2006.