

On Efficient Spatial Keyword Querying with Semantics

Zhihu Qian¹, Jiajie Xu^{1,2(✉)}, Kai Zheng^{1,2}, Wei Sun¹, Zhixu Li^{1,2},
and Haoming Guo³

¹ School of Computer Science and Technology, Soochow University,
Suzhou, China

{xujj,kevinz,zhixuli}@suda.edu.cn

² School of ITEE, The University of Queensland, Brisbane, QLD, Australia
kevinz@itee.uq.edu.au

³ Institute of Software, Chinese Academy of Sciences,
Beijing, China

haoming@nfs.iscas.ac.cn

Abstract. The fast development of GPS equipped devices has aroused widespread use of spatial keyword querying in location based services nowadays. Existing spatial keyword indexing and querying methodologies mainly focus on the spatial and textual similarities, while leaving the semantic understanding of keywords in spatial web objects and queries to be ignored. To address this issue, this paper studies the problem of semantic based spatial keyword querying. It seeks to return the k objects most similar to the query, subject to not only their spatial and textual properties, but also the coherence of their semantic meanings. To achieve that, we propose a novel indexing structure called NIQ-tree, which integrates spatial, textual and semantic information in a hierarchical manner, so as to prune the search space effectively in query processing. Extensive experiments are carried out to evaluate and compare it with other two baseline algorithms.

Keywords: Spatial keyword query · Query optimization · Probabilistic topic model · Semantic similarity

1 Introduction

Location based services (LBS) is widely used nowadays [3, 20, 23, 25], and spatial keyword query is known as an important technique for LBS systems. Extensive efforts have been made so far to support effective spatial keyword indexing and querying. Some pioneer work [5, 6] mainly focuses on the Spatial Keyword Boolean Query (SKBQ) that requires exact keywords match, and apparently, they may lead too few or no results to be returned because of the diversified textual expressions. To overcome this issue, researchers proposed some novel indexing structures to support Spatial Keyword Approximate Query (SKAQ) more

The original version of this chapter was revised: The authors' affiliations were incorrect. This has been corrected. An erratum to this chapter can be found at [10.1007/978-3-319-32049-6_29](https://doi.org/10.1007/978-3-319-32049-6_29)

recently in [16, 18, 21], which are able to handle the spelling errors and conventional spelling differences (e.g., ‘theater’ vs. ‘theatre’) that frequently appear in real applications. But still, they cannot retrieve the objects that are synonym but literally different to the keywords in query, such as ‘theater’ and ‘cinema’, due to the lack of understanding of the semantics in objects and queries. This gap motivates us to investigate other *semantic-aware* approaches that are able to capture the semantic meanings of spatial keywords.

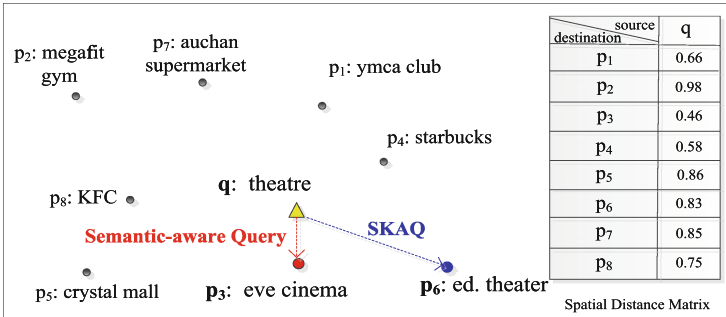


Fig. 1. An example of spatial keyword query

Example 1. Considering the example with eight spatial web objects in Fig. 1, where each object can be seen as a place of interest that has a spatial location and attached keywords. A user issues a query q to find a theatre close to the query location. If the SKBQ approaches [5, 6] are applied in the search engine, no result can be returned to the user because of the none precise match to the keyword ‘theatre’ in query. Alternatively, by using the SKAQ techniques [16, 18, 21], the search engine can return the object p_6 , which seems to be a relatively reasonable object to recommend in terms of spatial and textual similarities. However if checking those objects more carefully, we can easily observe that p_3 is the best match object should be returned, because it is not only closer to q in spatial, but also more relevant to q in semantics, meaning that the user intention can be satisfied as well. In order to make a more reasonable recommendation such as p_3 , the key problem is how to interpret and represent the semantics of keywords, and then take the semantic meanings into consideration of query processing.

To fulfill the gap mentioned above, we apply the probabilistic topic model (e.g., LDA [1]), known as a powerful tool in the field of machine learning, to convert the textual descriptions of objects into semantic representations (i.e. distribution over topics, or called topic distribution). By applying the LDA model on p_1 in Fig. 1, we can obtain five latent topics, and its topic distribution (over the five topics) is (0.72, 0.07, 0.07, 0.07, 0.07) (in Table 1). The topic distribution of p_1 indicates the semantic relevance between its textual description and each topic, e.g., 0.72 for topic ‘exercise’, implying that p_1 is very relevant to the this

topic. Similarly, we can compute the topic distributions of the query and all other objects (in Table 1). Note that, each topic distribution is a high dimensional vector in essential. The semantic similarity between textual descriptions of a query and a spatial object can be measured by their topic distributions (e.g. cosine distance of the two vectors). From Table 1, we can thus infer that ‘theatre’ and ‘eve cinema’ have close semantic similarity.

Once the topic model is incorporated, spatial keyword querying becomes challenging and time-consuming in despite of more meaningful feedbacks can be found. The reason can be summarized in three main aspects. Firstly, the topic distribution based indexing method has much higher dimensions associating to each object, which severely deteriorates the pruning efficiency (known as the ‘curse of dimensionality’ [11]) of most multi-dimensional search algorithms. Secondly, compared to the conventional SKBQ and SKAQ, it incurs more memory and I/O cost because additional space is required to store the topic distribution based object information, and I/O cost increases accordingly. Last but not the least, it is necessary to integrate information of multiple dimensions in the indexing and query processing, which makes the hybrid representation more difficult.

To address all above difficulties, we define a new type of spatial keyword query that incorporates spatial, textual and semantic similarities into account. To prune the search space effectively in query processing, we carefully design a hierarchical indexing structure called NIQ-tree, which can integrate spatial, textual and semantic information seamlessly in a hierarchical manner. Since iDistance [14] is one of the best known high dimensional indexing methods, which coincides to our topic distribution based representation of spatial web objects, we incorporate iDistance into the NIQ-tree to avoid the large dead space when indexing all objects in high dimensional space. To make efficient retrieval, a novel query processing mechanism on top of the NIQ-tree is proposed to prune the search space effectively based on some theoretical bounds. To sum up, our main contributions of this paper can be briefly summarized as follows:

- We introduce and formalize a new type of probabilistic topic model based similarity measure between a query and a object.
- We propose a novel hierarchical indexing structure, namely NIQ-tree, to integrate the spatial, semantic and textual information of the objects seamlessly while avoiding large dead space.
- We further design an efficient search algorithm, which can greatly prune the high dimensional search space in query processing based on some theoretical bounds.
- We conduct an extensive experiment analysis based on spatial databases and make the comparisons with two baseline algorithms, and then demonstrate the efficiency of our method.

2 Preliminaries and Problem Definition

In this section, we introduce some preliminaries and formalize the problem of this paper.

2.1 Probabilistic Topic Model

In order to recommend spatial web objects that can fulfill user’s intention, it is necessary not only to understand the semantic meanings of the textual descriptions embedded in objects and queries [12], but also to measure their semantic relevance accurately. Probabilistic topic model is a mature nature language processing technique that has been proven to be successful on theme interpretation and document classification. Therefore in this paper, we apply the Latent Dirichlet Allocation (LDA) model, i.e. one of the most frequently used probabilistic topic models, to understand the semantic meanings of textual description formed by words with respect to topics. Here, topics can be understood as possible semantic meaning of textual data defined as follows:

Definition 1 (Topic). A topic z represents a type of intended activity that a user may be interested in, such as ‘Chinese restaurant’, ‘coffee shop’, ‘supermarket’ and so on. Z is a preprocessed topic set, which is the union of all topics used to describe the meaningful semantics of textual descriptions.

By carrying out statistical analysis on the large amount of textual descriptions, the LDA model derives the semantic relevance of a topic to all relevant words, known as words distribution defined as follows:

Definition 2 (Words Distribution). Given the topic set Z and the set of all possible words V , the matrix $M = Z \times V_z$ ($V_z \subseteq V$) is used to represent the words distributions of all topics in Z , where V is the collection of all keywords that may appear in textual descriptions, and V_z is the keywords collection belonging to the topic z , apparently, $V_z \subseteq V$. Each M_z represents a distribution of a single topic over all words which belong to this topic and $M_z[w]$ is the topic proportion satisfies $\sum_{w \in V_z} M_z[w] = 1$, where $z \in Z$.

Definition 3 (Topic Distribution). Given a textual description W , the topic distribution of W , denoted as TD_W , is the statistical proportion for each keyword in W , where the topic proportion $TD_W[z]$ from W to topic z is calculated as

$$TD_W[z] = \frac{N_{w \in (W \cap W_z)} + \alpha}{|W| + |Z| \times \alpha} \quad (1)$$

where $N_{w \in (W \cap W_z)}$ is the number of keywords belongs to the given textual description of W in W_z ; α is the symmetric Dirichlet prior and generally set to 0.1. $|W|$ and $|Z|$ are the number of keywords in W and topics in Z respectively.

A topic distribution TD_W is a $|Z|$ -dimensional vector, which can be regarded as a point in a high dimensional topic space. Therefore, the topic distance of two textual descriptions can be calculated as the following definition.

Definition 4 (Topic Distance). Given two textual descriptions W and W' , their topic distance can be quantified by several similarity measures (e.g., Euclidean

Table 1. Topic distributions of textual descriptions

textual descriptions \ topics	topics				
	exercise	movie	drink	shop	food
ymca club (in p_1)	0.72	0.07	0.07	0.07	0.07
megafit gym (in p_2)	0.88	0.03	0.03	0.03	0.03
eve cinema (in p_3)	0.04	0.84	0.04	0.04	0.04
starbucks (in p_4)	0.07	0.07	0.72	0.07	0.07
crystal mall (in p_5)	0.07	0.07	0.07	0.72	0.07
ed. theater (in p_6)	0.07	0.72	0.07	0.07	0.07
auchan supermarket (in p_7)	0.07	0.07	0.07	0.72	0.07
KFC (in p_8)	0.04	0.04	0.04	0.04	0.84
theatre (in q)	0.07	0.72	0.07	0.07	0.07

and Cosine Distance). Here, we adapt the cosine distance to measure their distance in high dimensional topic space. The topic distance $\mathcal{D}_T(TD_W, TD_{W'})$ is defined as

$$\mathcal{D}_T(TD_W, TD_{W'}) = \frac{\sum_{z \in Z} (TD_W[z] \times TD_{W'}[z])}{\|TD_W\| \times \|TD_{W'}\|} \quad (2)$$

where $\|TD_W\|$ is the modulus of TD_W in $|Z|$ dimensions. It is obvious that the less topic distance of two arbitrary textual descriptions is, the more relevant they are in semantics according to the LDA interpretation.

Example 2. Table 1 shows the LDA interpretation on all the objects in Fig. 1. After running LDA, we derive the distribution of topic (e.g., ‘exercise’) over relevant words (e.g., ‘club’, ‘gym’) based on statistical concurrence. Then we derive the topic distribution of each textual description, where each number in the table is a topic proportion (e.g., $M_{gym'}[exercise] = 0.88$) that indicates their semantic relevance. Therefore, the topic distance between the textual description W of query point (e.g., $q.W = \text{‘theatre’}$) and textual description W' of point in the database (e.g., $p_3.W' = \text{‘eve cinema’}$) can be further quantified as $\mathcal{D}_T(W, W') = 0.09$.

2.2 Problem Definition

In this subsection, we give some basic definitions and then formalize the problem of this paper.

Definition 5 (Spatial Web Object). A spatial web object can be a shop, a restaurant or other place of interest whose location and textual descriptive information can be accessed through Internet. It is formalized as $o = (o.\lambda, o.\varphi)$, where $o.\lambda$ is the position of o and $o.\varphi$ is the textual description for describing o . We use the term *spatial object* to represent it in short in the rest of this paper.

Definition 6 (Spatial Keyword Query). Consider a query $q = (q.\lambda, q.\varphi, \tau)$, where $q.\lambda$ is the query location represented by a longitude and a latitude in the two dimensional geographical space; $q.\varphi$ is a group of words that describe user’s intention, such as ‘Chinese restaurant’; τ is a user-specified threshold of textual distance in case that strict textual similarity is required. The textual distance between the query q and spatial object o is denoted as $TD(q, o)$, which is measured by the Edit Distance [15] of their keywords.

Definition 7 (Candidate Object). Given a query q , a spatial object o is said to be a candidate object, if and only if its textual distance to q is no more than the threshold of q , i.e. $TD(q, o) \leq q.\tau$.

Note that, a spatial object can be returned to user only if it is a candidate object. Among all candidate objects, we rank them by the distance function subject to their spatial proximity and semantic coherence.

Definition 8 (Distance Function). Given a query q , a set of spatial object o , their spatial distance are calculated by the Euclidean distance of their geographical locations. We normalize it to range [0,1] by using the sigmoid function as shown in Eq. 3.

$$\mathcal{D}_S(q, o) = \frac{2}{1 + e^{-dist(q.\lambda, o.\lambda)}} - 1 \quad (3)$$

By combining the spatial distance and the topic distance, we further define a distance function of q and o , denoted as $\mathcal{D}(q, o)$ in Eq. 4.

$$\mathcal{D}(q, o) = \lambda \times \mathcal{D}_S(q, o) + (1 - \lambda) \times \mathcal{D}_T(q, o) \quad (4)$$

where λ is a user-specified parameter to balance the weight of the spatial and semantic distance.

Problem Statement. Given a query q , a set of spatial objects D , and user-specified integer k , this paper returns the k candidate objects that have the minimum distance to q .

3 Baseline Algorithms

In this section, we propose two baseline algorithms which explore the possibility of using existing techniques to solve the problem in this paper.

3.1 Quadtree Based Algorithm

The first baseline algorithm uses the Quadtree [7] to prune the search space in spatial dimension. In the method, the Quadtree, which only utilizes the spatial coordinates of the points, is used to index these points in two-dimensional space directly. Given a query q , the first baseline traverses the index structure to

find the spatial nearest object incrementally in terms of the *spatial best match distance* which is computed as follows:

$$\mathcal{D}_{sbm}(q, o) = \lambda \times \mathcal{D}_S(q, o) \quad (5)$$

It is easy to see that the spatial best match distance is always the lower bound of the distance between q and o .

In the processing of the query, we keep finding the next nearest point o' (based on spatial best match distance) and computing its distance $\mathcal{D}(q, o')$ to q . During the process, we keep track of the k -th minimum distance as the upper bound of final results \mathcal{D}_{ub} based on a priority queue. If the spatial best match distance of next obtained object exceeds \mathcal{D}_{ub} already, the search algorithm terminates since all remaining objects have no chance to be better than the current top- k results.

3.2 MHR-tree Based Algorithm

The basic idea of the second baseline algorithm is mainly motivated by some early work for approximate string search in spatial database [5, 6, 21]. We use a hybrid indexing structure called Min-wise signature with linear Hashing R-tree (MHR-tree) [21], which combines R-tree [10] with signatures embedded in the nodes, to prune the search space in both spatial and textual dimension.

The indexing structure of MHR-tree embeds the min-wise signature in a R-tree node. For every leaf node u in the MHR-tree, we compute the n -grams G_p and the corresponding min-wise signature $S(G_p)$ of every point p in this node, then store all $(p, S(G_p))$ pairs in node u . For every non-leaf node u in the MHR-tree, with its child node entries c_1, \dots, c_f and every child node w_i pointed by c_i , we store the min-wise signature of the node pointed to by c_i , i.e., $s(G_{w_i})$. Then the signature of a non-leaf node u can be computed using $s(G_{w_i})$ as

$$s(G_u)[i] = \min(s(G_{w_1})[i], \dots, s(G_{w_f})[i]) \quad (6)$$

where $s(G_u)[i]$ is proportion of $s(G_u)$. Finally, we store $s(G_u)$ of the non-leaf node in the index entry that points to u in u 's parent.

In processing a query q , we use a min-priority queue that orders objects in the queue according to their distance to the query point. We search from the root of the MHR-tree and prune the search space by spatial best match distance similar to the search in the first baseline. Additionally, here we use some strategies to avoid checking all points in the node of MHR-tree according to their textual information. When we reach a leaf node, we traverse every point p in the node and insert it into the queue according to Lemma 1 [8, 19] if it satisfies:

$$|G_p \cap G_q| \geq \max(|p|, |q|) - 1 - (\tau - 1) \times n \quad (7)$$

where G_p and $|p|$ are the set of n -grams and string length in p respectively, τ is the user-specified threshold of textual distance (e.g., Jaccard distance [13] and edit distance [15]) and n is 3 if we choose 3-gram.

Lemma 1 (From [8]). For strings σ_1 and σ_2 , if their edit distance is τ , then $|G_{\sigma_1} \cap G_{\sigma_2}| \geq \max(|\sigma_1|, |\sigma_2|) - 1 - (\tau - 1) \times n$.

However, when we reach a non-leaf node, its child w_i will be added to the queue according to Lemma 2 if it satisfies:

$$|\widehat{G_{w_i} \cap G_q}| \geq |q| - 1 - (\tau - 1) \times n \quad (8)$$

where $|\widehat{G_{w_i} \cap G_q}|$ is the estimation of $|G_{w_i} \cap G_q|$. Whenever a point is removed from the head of the queue, it is added to the result set. The search terminates when there are k points in the result or the priority queue becomes empty.

Lemma 2. Let G_u be the set for the union of n -grams of strings in the subtree of node u in a MHR-tree. Given a query q , if $|G_u \cap G_q| \geq |q| - 1 - (\tau - 1) \times n$, then the subtree of node u does not contain any point in the result.

Proof: G_u is a set, which contains distinct n -grams. The proof follows by the definition of G_u and Lemma 1. \square

4 NIQ-tree Based Algorithm

In this section, we propose an improved hybrid indexing structure NIQ-tree based on iDistance [14]. The iDistance is a well-known index scheme for high-dimensional similarity search, with a basic idea to group all objects by clustering (e.g., by k-means, k-medoids, etc.), which enables us to achieve superior pruning effect in query processing. By utilizing iDistance to sketch the topic distributions, the NIQ-tree is expected to support effective pruning on spatial, textual and semantic dimensions simultaneously.

Indexing Structure. The NIQ-tree is a three layered hybrid indexing structure shown as Fig. 2, where the spatial, semantic and textual layers are integrated in a vertical way. In designing NIQ-tree, we adopt a spatial first method because of the better pruning effect in spatial domains, which can be explained by its two dimensional nature (v.s. high dimensions of topic and textual domains). The basic form of a NIQ-tree node is $N = (p, rect, o, r)$, where p is the pointer(s) to its child node(s); $rect$ is the minimum bounding rectangle (MBR) in spatial of all objects contained by N ; o and r are the center point and radius of a topic space hyper-sphere that covers the topic distributions of all objects contained by N respectively. On top of all spatial objects, we use Quadtree to index them in the spatial domain according to their spatial closeness first. For each leaf node of Quadtree, all objects are further organized by iDistance index in the topic layer, such that objects are grouped and managed by their topic coherence. For each leaf nodes N of the topic layer, it is referenced to a set of n -gram based inverted lists in the textual layer, and similar to MHR-tree, the n -gram inverted lists functionally sketch out the textual descriptions of objects contained in N . In this way, it is possible to filter irrelevant objects according to the $q.\varphi$ and $q.\tau$ specified in query.

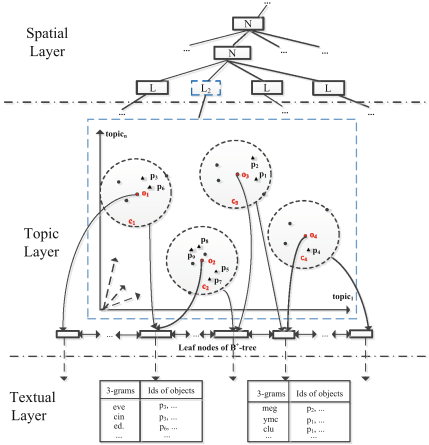


Fig. 2. NIQ-tree

Example 3. A three-layered NIQ-tree is shown in Fig. 2. Assuming that all POIs in Fig. 1 are divided into the same leaf node L_2 in spatial layer, these points are partitioned into four clusters in topic layer, where C_1 contains $\{p_3, p_6\}$, C_2 contains $\{p_5, p_7, p_8, p_9\}$, C_3 contains $\{p_1, p_2\}$ and C_4 contains $\{p_4\}$. It is clear that all points in the same cluster have high semantic similarity. At last, in textual layer, we construct a n -gram based inverted list.

In constructing the NIQ-tree, we build up a Quadtree for all points in spatial database first like the first baseline algorithm. Then for the points in every leaf node of the Quadtree, we use i Distance to cluster these points based on their topic distributions of all contained objects, and construct a B^+ -tree to organize the nodes (each node represents a cluster) according to the *key* value computed as follows:

$$key = i \times c + \mathcal{D}_T(p, o_i) \quad (9)$$

where i is the identifier of the partition P_i , c is a constant to partition the single dimension space into regions so that all points in P_i will be mapped to the range $[i \times c, (i + 1) \times c)$, o_i is a reference point to P_i and p is the point in this partition. In this way, the high dimensional topic space is expressed by a transformed point *key* in single dimension space, and B^+ -tree can thus be applied directly. Next, we set the $N.o$ and $N.r$ for all spatial layer nodes of the NIQ-tree in an bottom up fashion, such that they can cover the center point and radius of N 's child nodes in minimum topic space cost. Finally, we build the inverted lists for every leaf node of the B^+ -tree by the n -gram method.

Query Processing. Algorithm 1 illustrates the query processing mechanism over the NIQ-tree. Given a query q , the objects retrieval is carried out on the spatial, topic and textual domains of the index alternately. Starting from the root

of index, we traverse the spatial layer nodes in the ascending order of the *best match distance* $D_{bm}(q, N)$ with respect to q defined as the following formula:

$$D_{bm}(q, N) = \lambda \times \min_{p \in N.mbr} \mathcal{D}_S(q, p) + (1 - \lambda) \times \min \mathcal{D}_T(q, N) \quad (10)$$

where $\min_{p \in N.mbr} \mathcal{D}_S(q, p)$ and $\min \mathcal{D}_T(q, N)$ denote the minimum possible spatial and topic distance from q to any object contained in the node N . Let $\|TD_{q,\varphi}, N.o\|$ be the cosine distance between textual description $q.\varphi$ and reference point o in topic layer, the minimum possible topic distance $\mathcal{D}_T(q, N)$ can be computed as follows:

$$\min \mathcal{D}_T(q, N) = \begin{cases} 0 & \|TD_{q,\varphi}, N.o\| \leq N.r \\ \|TD_{q,\varphi}, N.o\| - N.r & \|TD_{q,\varphi}, N.o\| > N.r \end{cases} \quad (11)$$

It is noted that $D_{bm}(q, N)$ is the lower bound distance D_{lb} to q for all unvisited points according to its definition.

In the query processing, the node we fetch from the priority queue is a non-leaf node, we add all its child nodes to the queue; otherwise, we access the topic and textual layer indices subject to this node to access the candidate objects they covers. During the search in topic layer, the leaf node in the B^+ -tree can be identified according to key value. Similar to iDistanceKNN search [14], we browse the space by expanding the radius R of the hyper-sphere centered at the query point q . At each time, R is increased by ΔR (i.e., $R = R + \Delta R$). If the leaf node of this layer intersects with the searching sphere, we traverse the points in this node according to its key value in the range of $[i \times c + dis(q, o_i) - R, i \times c + dist(q, o_i) + R]$, where $dist(q, o_i)$ is the distance between q and reference point o_i . Then, by checking its inverted list in textual layer, we find the spatial objects whose textual distance to q is no more than $q.\tau$. Especially, we dynamically maintain the top- k minimum distance for all scanned points and keep the k -th minimum distance as an upper bound \mathcal{D}_{ub} . The radius of search sphere R stops increasing when the following condition holds for all unvisited topic layer leaf nodes:

$$\lambda \times \min_{p \in N.mbr} \mathcal{D}_S(q, p) + (1 - \lambda) \times R \geq \mathcal{D}_{ub} \quad (12)$$

where $N.mbr$ is the MBR of a spatial layer leaf node N . Obviously, $(\lambda \times \min_{p \in N.mbr} \mathcal{D}_S(q, p) + (1 - \lambda) \times R)$ is a lower bound distance from q to a topic layer leaf node rooted in N according to Lemma 3. The whole search algorithm terminates when \mathcal{D}_{lb} is no less than \mathcal{D}_{ub} because the remaining unvisited points have no opportunity to be better than the current top- k results.

Lemma 3. *Given a query q and a NIQ-tree, if N is a spatial layer leaf node of the NIQ-tree, then the search in topic layer with respect to N terminates when $(\lambda \times \min_{p \in N.mbr} \mathcal{D}_S(q, p) + (1 - \lambda) \times R) \geq \mathcal{D}_{ub}$.*

Proof: $\min_{p \in N.mbr} \mathcal{D}_S(q, p)$ is the minimum spatial distance from q to a spatial layer node N , which is the lower bound of spatial distance from q to any unvisited point in this node. R is the minimum topic distance from q to any unvisited point in the cluster, which is the lower bound of topic distance. So $(\lambda \times \min_{p \in N.mbr} \mathcal{D}_S(q, p) + (1 - \lambda) \times R)$ is a lower bound distance from q to a topic layer leaf node. Lemma 3 can be proven. \square

Algorithm 1. NIQ-tree based Search Algorithm

Input: dataset D , query q and user-specified k and λ
Output: top- k result set V

- 1 Upper Bound $\mathcal{D}_{ub} = +\infty$;
- 2 Lower Bound $\mathcal{D}_{lb} = 0$;
- 3 Search radius $R = 0.1$;
- 4 Put NIQ-tree root into a priority queue U ;
- 5 **while** $U \neq \emptyset$ **do**
- 6 Pop an element N from U ;
- 7 **if** N is a non-leaf node **then**
- 8 Add its children to U ;
- 9 **else if** N is a leaf node **then**
- 10 **for every** iDistance node N' in N intersecting with searching sphere **do**
- 11 **for every** object o in N' **do**
- 12 Check the n -gram inverted lists of N' ;
- 13 **if** o is a candidate object **then**
- 14 Compute $\mathcal{D}(q, o)$ using Eq. 4;
- 15 **if** $\mathcal{D}(q, o) < \mathcal{D}_{ub}$ **then**
- 16 Add o to V ;
- 17 Update \mathcal{D}_{lb} and \mathcal{D}_{ub} ;
- 18 **if** $(\lambda \times \min_{p \in N.mbr} \mathcal{D}_S(q, p) + (1 - \lambda) \times R) \geq \mathcal{D}_{ub}$ **then**
- 19 break;
- 20 $R = R + \Delta R$;
- 21 **if** $\mathcal{D}_{lb} \geq \mathcal{D}_{ub}$ **then**
- 22 break;
- 23 return V ;

5 Experiment Study

In this section, we conduct extensive experiments on real datasets to evaluate the performance of our proposed index and search algorithms.

5.1 Experiment Settings

We create the real object datasets by using the online check-in records of Foursquare within the area of New York City. Each check-in record of Foursquare contains the user ID, venue with geo-location (place of interest), time of check-in and the tips written in plain English. We put the records belonging to the same object to form textual descriptions of the objects. The topic distributions over words are obtained by the textual descriptions in the tips associated with the location, and then the textual descriptions for each place are interpreted into a probabilistic topic distribution by LDA model. The number of objects in the whole dataset is 422030 in sum.

Table 2. default values of parameters

Parameter	Default value	Description
k	10	Top- k results
λ	0.5	Weight factor
τ	3	Threshold of edit distance
D	200 K	No. of objects
c	4 K	Capacity of quadtree leaf node
m	20	Number of clusters

We compare the query time cost and number of visited objects during the search processing of the proposed method (NIQ-tree) with the two baseline algorithms proposed in Sect. 3. The default values for parameters are given in Table 2. In the experiments, we vary one parameter and keep the others constant to investigate the effect of this parameter. All algorithms are implemented in Java and run on a PC with 2-core CPUs at 3.2 GHz and 8 GB memory.

5.2 Performance Evaluation

In this part, we vary the values of parameters in Table 2 to compare the NIQ-tree with two baseline algorithms and investigate the effect of each parameter.

Effect of k . In the first set of experiments, we study the effect of the intended number of results k by plotting the query time and visited objects (denoting I/O) on the dataset. As shown in Fig. 3, our proposed indexing structure, NIQ-tree, significantly outperforms all other two baseline indexing methods on the same dataset. Particularly, the NIQ-tree based method is almost 2-3 times faster than the MHR-tree with respect to query time. All algorithms incur high cost in both number of visited objects and query time as k increases, which is not beyond our expectation because the k -th match distance becomes greater, which leads more candidates to be retrieved.

Effect of λ . Next we study the effect of different weight factors λ . As shown in Fig. 4, all algorithms including Quadtree, MHR-tree and NIQ-tree based methods have ascending tendency of I/O and query time when the value of λ goes up. In contrast, the NIQ-tree is superior than two other approaches because of its superior pruning effect in spatial, semantic and textual domains.

Effect of τ . Then we investigate the query performance of these algorithms when the threshold τ of the edit distance between object and query is varying. Figure 5 shows the results of our experiment. With the increase of τ , all algorithms incur more time cost and more visited objects because more candidate objects are retrieved since their edit distance to query are less than the threshold. Especially, the NIQ-tree still outperforms the other two baseline algorithms with respect to query time and visited objects.

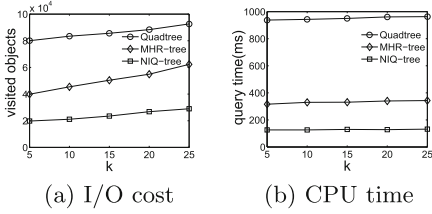


Fig. 3. Effect of k

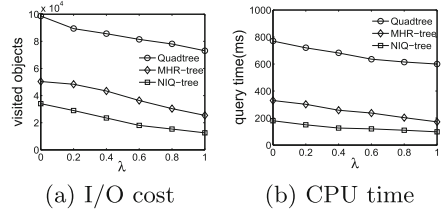


Fig. 4. Effect of λ

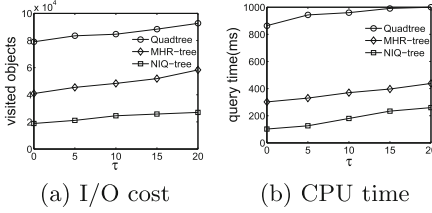


Fig. 5. Effect of τ

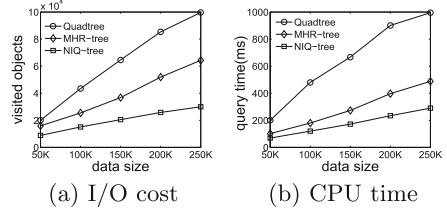


Fig. 6. Effect of D

Effect of D . In order to evaluate the scalability of all algorithms, we sample the dataset of New York City to generate datasets with different number of objects varying from 50 K to 250 K, and report the query time and visited points in Fig. 6. Within our expectation, the query time and the number of visited objects of all three methods increase linearly with respect to the size of dataset. But it is worth to note that the NIQ-tree based method performs much more efficient than the others.

In NIQ-tree, there are several parameters of iDistance index which may have effects on the performance, including the size c of leaf node in spatial layer and the number m of clusters in iDistance.

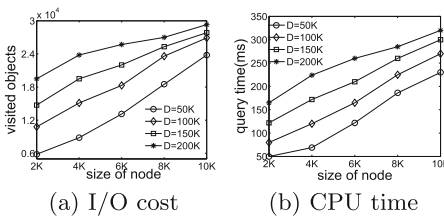


Fig. 7. Effect of c

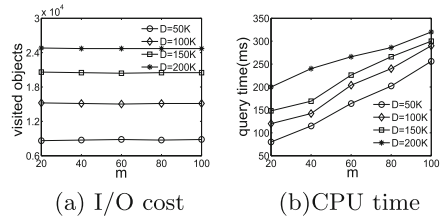


Fig. 8. Effect of m

Effect of c . As shown in Fig. 7, the capacity of leaf node in Quadtree affects the performance of our proposed indexing structure. With the increase of node size c , both visited objects and query time increase. That is to say that the query

time and I/O increase with the size of c . From Fig. 7, it is noted that the increase of data size D also makes more I/O and query time when c remains the same.

Effect of m . It is shown in Fig. 8 that the performance of our proposed index are affected when the number of clusters m changes. On one hand, we can observe that the visited objects remain almost constant with respect to m but increase with the data size D from the left figure of Fig. 8. On the other hand, the query time has a nearly linear increase with m and it also increases when the data size D becomes larger.

6 Related Work

The related work to our study mainly include probabilistic topic model and spatial keyword query.

Probabilistic Topic Model. Probabilistic topic model is a statistical method to analyze the words in documents and to discover the themes that run through the words, how those themes are connected to each other, with no prior annotations or labeling of documents been required. Based on the topic models, it is possible to measure the semantic relevance between a text to a theme, as well as that between different texts (in semantic level). There are several classical topic models including LDA [1], Dynamic Topic Model, Dynamic HDP, Sequential Topic Models, etc. Above techniques have been widely used in applications like document classification, user behavior understanding, functional region discovery, etc. In this paper, we tend to integrate topic model and spatial objects for efficient spatial keyword querying with semantics.

Spatial Keyword Query. With the prevalence of spatial objects associated with textual information on the Internet, spatial keyword queries that exploit both location and textual description are gaining in prominence. Some efforts are made to support the SKBQ [5, 6, 17, 22] that requires exact keywords match, which may lead few or no results to be found. To overcome this problem, some efforts are further made to support the SKAQ [16, 18, 21], so that the query results are no longer sensitive to spelling errors and conventional spelling differences. Many novel indexing structures are proposed to support efficient processing on SKBQ and SKAQ, such as IR-tree [5], IR²-tree [6], MHR-tree [21], S2I [18], etc. Numerous work studies the problem of spatial keyword query on collective querying [2], fuzzy querying [26], confidentiality support [4], continuous querying [9], interactive querying [24], etc. But as far as we know, none of those existing approaches can retrieve spatial objects that are semantically relevant but morphologically different. Therefore, in this paper, we investigate the topic model based spatial keyword querying to recommend users spatial objects that have both high spatial and semantic similarities to query.

To the best of our knowledge, this is the only work to consider the fusion of topic model and spatial keyword query, so that spatial objects can be recommended more rationally by the interpretation of textual descriptions for objects and user intentions.

7 Conclusion

This paper studies the problem of searching spatial objects more effectively by converting keywords matching to semantic interpretation. The probabilistic topic model is utilized to interpret the textual descriptions attached to spatial objects and user queries into topic distributions. To support the efficient top- k spatial keyword query in spatial, topic and textual dimension, we propose a novel hybrid index structure called NIQ-tree, and effective searching algorithm to prune the high dimensional search space regarding to spatial, semantic and textual similarities. Extensive experimental results on real datasets demonstrate the efficiency of our proposed method.

Acknowledgement. This work was partially supported by Chinese NSFC project under grant numbers 61402312, 61402313, 61572335, 61232006, the Key Research Program of the Chinese Academy of Sciences under grant number KGZD-EW-102-3-3, and Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

1. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. *J. Mach. Learn. Res.* **3**, 993–1022 (2003)
2. Cao, X., Cong, G., Jensen, C.S.: Collective spatial keyword querying. In: *SIGMOD* (2011)
3. Chen, L., Cong, G., Jensen, C.S.: Spatial keyword query processing: An experimental evaluation. *PVLDB* **6**(3), 217–228 (2013)
4. Chen, Q., Hu, H., Xu, J.: Authenticating top- k queries in location-based services with confidentiality. *PVLDB* **7**(1), 49–60 (2013)
5. Cong, G., Jensen, C.S., Wu, D.: Efficient retrieval of the top- k most relevant spatial web objects. *PVLDB* **2**(1), 337–348 (2009)
6. De Felipe, I., Hristidis, V., Rishé, N.: Keyword search on spatial databases. In: *ICDE* (2008)
7. Finkel, R.A., Bentley, J.L.: Quad trees a data structure for retrieval on composite keys. *Acta informatica* **4**(1), 1–9 (1974)
8. Gravano, L., Ipeirotis, P.G.: Approximate string joins in a database (almost) for free. In: *ICDE* (2001)
9. Guo, L., Shao, J., Aung, H.H., Tan, K.-L.: Efficient continuous top- k spatial keyword queries on road networks. *Geoinformatica* **19**(1), 29–60 (2015)
10. Guttman, A.: R-trees: A dynamic index structure for spatial searching. In: *SIGMOD* (1984)
11. Har-Peled, S., Indyk, P., Motwani, R.: Approximate nearest neighbors: Towards removing the curse of dimensionality. In: *ACM symposium on Theory of computing* (1998)
12. Hua, W., Wang, Z., Wang, H., Zheng, K., Zhou, X.: Short text understanding through lexical-semantic analysis. In: *ICDE* (2015)
13. Jaccard, P.: *Etude comparative de la distribution florale dans une portion des alpes et du jura*. Impr. Corbaz (1901)
14. Jagadish, H.V., Ooi, B.C., Tan, K.-L.: idistance: An adaptive b+-tree based indexing method for nearest neighbor search. *ACM TODS* **30**(2), 364–397 (2005)

15. Levenshtein, V.I.: Binary codes with correction for deletions and insertions of the symbol 1. *Problemy Peredachi Informatsii* **1**(1), 12–25 (1965)
16. Li, F., Yao, B., Tang, M.: Spatial approximate string search. *TKDE* **25**(6), 1394–1409 (2013)
17. Li, G., Feng, J., Xu, J.: Desks: Direction-aware spatial keyword query. In: *ICDE* (2012)
18. Rocha-Junior, J.B., Gkorgkas, O., Jonassen, S., Nørnvåg, K.: Efficient processing of top-k spatial keyword queries. In: Pfoser, D., Tao, Y., Mouratidis, K., Nascimento, M.A., Mokbel, M., Shekhar, S., Huang, Y. (eds.) *SSTD 2011*. LNCS, vol. 6849, pp. 205–222. Springer, Heidelberg (2011)
19. Ukkonen, E.: Approximate string-matching with q-grams and maximal matches. *Theor. Comput. Sci.* **92**, 191–211 (1992)
20. Wang, H., Zheng, K.: Sharkdb: An in-memory column-oriented trajectory storage. In: *CIKM* (2014)
21. Yao, B., Li, F., Hadjieleftheriou, M., Hou, K.: Approximate string search in spatial databases. In: *ICDE* (2010)
22. Zhang, C., Zhang, Y., Zhang, W., Lin, X.: Inverted linear quadtree: Efficient top k spatial keyword search. In: *ICDE* (2013)
23. Zheng, K., Huang, Z., Zhou, A.: Discovering the most influential sites over uncertain data: A rank based approach. *TKDE* **24**(12), 2156–2169 (2012)
24. Zheng, K., Su, H.: Interactive top-k spatial keyword queries. In: *ICDE* (2015)
25. Zheng, K., Zheng, Y.: Online discovery of gathering patterns over trajectories. *TKDE* **26**(8), 1974–1988 (2014)
26. Zheng, K., Zhou, X.: Spatial query processing for fuzzy objects. *VLDB* **21**(5), 729–751 (2012)