IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 14, 2019, accepted September 4, 2019, date of publication September 16, 2019, date of current version October 3, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2941760

An Efficient Top-*K* Spatial Keyword Typicality and Semantic Query

XIAOYAN ZHANG, XIANGFU MENG, JINGUANG SUN, QUANGUI ZHANG, AND PAN LI
School of Electronic and Information Engineering, Liaoning Technical University, Huludao 125105, China

Corresponding author: Xiangfu Meng (mengxiangfu@lntu.edu.cn)

This work was supported in part by the National Science Foundation of China under Grant 61772249, and in part by the Natural Science Foundation of Liaoning Province, China, under Grant 20170540418.

ABSTRACT Existing spatial keyword query processing models mainly consider the spatial proximity and text relevancy between spatial objects and spatial keyword query, which usually makes the top-k answer objects are similar to each other. However, the user hopes to obtain the top-k results that are typical and semantically related to his/her query intention. This paper proposes a top-k spatial keyword typicality and sematic querying approach which can expeditiously provide top-k typical and semantically related objects to the given query. The approach consists of two processing steps. During the offline step, we first analyze the location-semantic relationships between spatial objects by considering both the location similarity and document semantic relevancy between them. For measuring the semantic similarity between documents associated to the spatial objects, we propose two methods, the keyword coupling relationship-based document similarity measure and the Word2Vec-CNN-based document similarity measure. Then, the Gaussian probabilistic density-based estimation method is leveraged to find a few representative objects from the dataset and then the order/permutation of remaining objects in the dataset can be generated corresponding to each representative object. The objects in the permutation are ranked in descending order according to their location-semantic relationships to the representative object. When a spatial keyword query coming, the online processing step first computes the spatial proximity and semantic relevancy between the query and each representative object, and then a small number of orders generated in the offline step can be selected and used at querying time to facilitate top-k typical and semantically related object selection by using the threshold algorithm (TA). Results of a preliminary user study demonstrate our location-semantic relationship measuring method can capture the location similarity and semantic relevancy between spatial objects accurately. The efficiency of typicality analysis and TA-based top-k selection algorithm is also demonstrated.

INDEX TERMS Spatial keyword query, location-semantic relationship, typicality, top-k selection.

I. INTRODUCTION

With the rapid development of GPS and universal use of mobile internet, more and more spatial objects (usually containing the geo-textual information) are becoming available on the Web that represent Point of Interests (POIs) such as restaurant, hotels, cafes, and tourist attractions. These POIs mainly consist of two types of information, the geo-location (in the form of longitude and latitude) and the textual document (such as names, amenities, and special features, etc.) [5], [8], [27]. Table 1 shows an instance of spatial database, where each row represents to a spatial object/POI.

Nowadays, most of the Web queries are involved of the location information and thus the LBS (Location Based Service) becomes more and more popular (such as Google Map, Foursquare, Ctrip, Dianping, etc). However, the spatial database usually contains a large size of data and thus too many answer problem which is referred to ‘‘information overload’’ often occurs when a user issues a non-selective spatial keyword query [6], [20], [26], [35].

For example, nearly 500 restaurants would be returned for a user who wants to find a restaurant providing *Thai Food located at the* 706 *Mission St*, *San Francisco* by using the Yelp Web site. In such a context, the user may hope the system can recommend a small number of top-k restaurants that are typical among them. Typicality analysis is a concept from

The associate editor coordinating the review of this manuscript and approving it for publication was Bora Onat.

TABLE 1. An Instance of Spatial Database.

OID	Location	Description
o_1	116.36, 39.91	Swimming pool, WiFi, Breakfast
o_2	116.20, 39.99	WiFi, Breakfast
o_3	110.58, 35.74	Breakfast, Swimming pool, Subway
o_4	119.65, 33.32	Conference, Internet, Swimming pool
o_5	121.16, 42.58	Internet, Airport service, Conference

psychology and cognition science and is firstly applied to database query answering by [14]. The concept "typical" means that an object o in a set of spatial objects D is more representative than the others if o is more likely to appear in D. Meanwhile, the user may also like to consider the restaurants providing the same (or related) kind of tastes such as *Vietnamese Food*, *Hunan Food*, etc. This recommendation can also broaden the user knowledge and perspectives. Therefore, providing a few top-k results that are typical and semantically related to the given query is very helpful for users in obtaining the complete and effective information.

To deal with the problem of "information overload", top-k (ranking) query approach and its variants have been proposed to evaluate a few numbers of spatial objects that have both the high spatial proximity and text relevancy to the given spatial query as the answer [6], [7], [17], [18], [30], [35]. More specifically, let o be a spatial object in the form of $(o.loc, o.doc)$, where $o.loc$ is a location consisted of latitude and longitude and $o.doc$ is a text document associated to the spatial object. Let D be the universe of all spatial objects in a spatial database. A top-k spatial keyword query q is in the form of $q : (loc, keywords, k, \alpha)$, where loc means the query location, $keywords$ is the set of $\{w_1, w_2, \ldots, w_m\}$ (the element w_i corresponding to a keyword is usually computed by the traditional $tf \cdot idf$ weighting method), k is the number of requested result objects, and α plays a role for weighting the spatial proximity and text relevancy in the scoring function which is showed as follows [3], [6],

$$Score(o, q) = \alpha * S_{Loc}(o.loc, q.loc) + (1-\alpha) * S_{Doc}(o.doc, q.keywords) \quad (1)$$

The existing top-k results of q is a list of top-k spatial objects having the highest scores by combining the spatial proximity and text relevancy according to the scoring function (Equation (1)). However, this kind of approaches is confronted with three shortcomings, that are, (i) it does not consider the semantic relevancy between the query keywords and textual documents associated to spatial objects, (ii) it needs to compute the similarities of all matched spatial objects to the query and then find the top-k objects with the highest ranking scores as the exact answer, which usually makes the online computation exhaustively and longer response delay, (iii) the top-k answer objects obtained by existing spatial query models are usually too similar with each other, which are not benefit for users to recognize the features of whole dataset and broad the user's perspective.

On top of our previous work [24], this paper proposes a novel top-k spatial keyword query approach which can expeditiously find the top-k typical and semantically related and answer objects to the given query. We first compute the location and semantic similarities between every pair of spatial objects in the entire dataset and then such two kinds of similarities are combined to form the location-semantic relationships between spatial objects. Next, we take advantage of probabilistic density-based estimation method to capture a certain number of representative objects from the entire dataset according to the location-semantic relationships of spatial objects, and then create the order of remaining objects in the dataset corresponding to each representative object. When a user enters a spatial keyword query, a small number of representative objects with the highest location-semantic proximities to the given query and their corresponding orders would be selected and then be leveraged to find the top-k typical and relevant answers.

Our contributions are summarized as follows:

(1). A novel location-semantic relationship measuring method is proposed, which considers both the semantic relevancy and location similarity between spatial objects. For measuring the semantic similarity between documents associated to the spatial objects, we propose two methods, the keyword coupling relationship-based document similarity measure and the Word2Vec-CNN-based document similarity measure.

(2). A new typicality estimation method is proposed for finding the representative spatial objects from dataset according to the location-semantic relationships between all spatial objects.

(3). A TA (threshold algorithm)-based top-k selection algorithm, which is used to expeditiously pick the top-k typical and relevant spatial objects from dataset, is presented.

The rest of paper is organized as follows. Section 2 reviews related work. Section 3 presents the definitions and solution framework. Section 4 describes the location-semantic relationship measuring method while Section 5 proposes the algorithm for retrieving the top-k typical and semantically related answer objects. Section 6 shows the experimental results and the paper is concluded in Section 7.

II. RELATED WORK

Several approaches have been proposed to deal with the issues of spatial keyword queries over spatial databases [2]–[7], [9], [17], [18], [20]. According to the different query objectives and result units, these approaches can be divided into four categories. (i) Boolean *kNN* Query [9], [17], [18] retrieves the k spatial objects nearest to the query location and the text description of each object contains all the query keywords. (ii) top-k Range Query [29], [30], [33] finds the k spatial objects having the highest textual relevance to the query keywords and their locations are within the query region. (iii) top-k *kNN* Query [3]–[6], [35] ranks the top-k spatial objects according to their location proximity and text relevancy. More specifically, it retrieves the k objects having the highest ranking scores which are measured by a weighted combination of their distances to the query location and the textual similarity between their textual descriptions and query keywords. (iv) Collective Spatial Keyword Query (CSKQ) [2], [11] returns a set of the objects that collectively cover user's query keywords, those objects are close to the query location and have small inter-object distances. Following the CSKQ, the Reverse Collective Spatial Keyword Query (RCSKQ) [26], [32] returns a region, in which the query objects are qualified objects with the highest spatial and textual similarity. Recently, the top-k kNN, CSKQ, and RCSKQ query models are the most popular techniques in the current spatial keyword query processing, and the CSKQ and RCSKQ are the variants of the top-k kNN. However, it should be pointed out that the top-k kNN query and its variants rarely consider the relevancy of query keywords and text documents of spatial objects in semantics [24]. Furthermore, the top-k objects in the answer set are usually very similar to each other which can neither effectively reflect the features of the whole dataset nor broaden the user's perspectives.

To quickly retrieve the matching query results, some hybrid index structures (such as IR-tree [6], [19], quad-tree [12], [34], S2I [28], etc.) are developed to assist the online query processing. These can be mainly classified into three categories. The first is the two-stage index structure, in which a R-tree is firstly used to find the nearest spatial objects to the query location and then for each neighbor spatial object, an inverted file is used to rank the objects according to their text relevancy to the query keywords. This two-stage combination cannot be suitable for top-k spatial keyword query processing since it is difficult to determine in advance the number of nearest neighbors needed to retrieve for satisfying the final top-k result selection by combining the spatial proximity and text relevancy. The second is IR-tree and its variants (such as IR2-tree [9], bIR*-tree [33]), these kinds of indexes integrate the inverted file (or signature file, bit map) for text retrieval and the R-tree for spatial proximity querying to formulate a hybrid index structure which can prune the searching space by simultaneously making use of both spatial proximity and text matching. The third is the combination of quad-tree and inverted file, which first builds the spatial region by using quad-tree, and then, builds the inverted file for the objects in each node partitioned by the quad-tree. It should be pointed out that, although the last two indexes are more efficient than the first one, they are sub-optimal to deal with the top-k selection when k is small since they have to compare all the objects in the candidate/matching set by using the scoring function showed in formula (1) and then to pick the top-k objects having the highest ranking scores.

Our approach is also related to the text semantic similarity measuring methods, which can be mainly classified into four categories. The first one is based on the knowledge bases (KB), such as WordNet, Probase, and Wikipedia, to split text and then capture the keyword relationships [16]. However, the keywords and their relationship measures in WordNet and Wikipedia are subjective and cannot reflect the relationships between keywords against the datasets. In addition, the keywords or concepts uncovered by KB would not be processed. The second category is based on statistic methods to capture the keyword relationships. For example, Vector Space Model and Context Vector Model (CVM-VSM) [15] both capture the similarity relationships between terms in document set by using their co-occurrence information while they did not consider the inter-relation between keywords. The third category is based on the topic models such as LDA [31] and LSI [21]. Although the topic models have achieved a certain improvement over the traditional similarity measuring methods such as Bag of Words (BOW) and CVM-VSM models, the significance of improvement and generalized ability is not enough in processing some special scenarios (such as short texts). Unfortunately, the text description of spatial objects are often short texts, as [13] pointed out, the short texts usually do not contain sufficient statistical information to support traditional topic models for text processing. The fourth category is word embedding-based similarity measures. Word embedding is the collective name for a set of language modeling and feature learning techniques in natural language processing where words or phrases from the vocabulary are mapped to dense, distributed, fixed-length vector representations in a low-dimensional space relative to the vocabulary size. The popular techniques of word embedding mainly contain the Word2Vec [22], [23] (such as Skip-gram and CBOW), genism, FastText, and GloVe [25]. Word embedding technique is very successful in the natural language processing (NLP). In this paper, we will adopt word embedding techniques to measure the semantic similarities between documents. However, word embedding technique such as Word2Vec supposes the nearby/adjacent words/phrases (in a fixed window size) usually having the strong contextual relations while it is inefficient for capturing the intra-correlations between keywords that are far from each other in the same document and the inter-correlations between keywords across the different documents. Therefore, in this paper, we develop two document semantic similarity measuring methods. The one is keyword coupling relationship-based document similarity measure, which considers both the intra- and inter-coupling relationships of keywords in the documents. The other is Word2Vec-CNN-based document similarity measure, which combines the Word2Vec technique and Convolutional Neural Network to capture the latent features of the documents.

Furthermore, this paper is an extension of the conference version appeared in DASFAA 2019 [24]. Compared to it, this paper contains the following new materials:

(1) A new method which combines the Word2Vec and CNN techniques is proposed for measuring the document semantic similarity.

(2) The processing procedure of density-based probability estimation method for finding representative spatial objects is described in details and the complexity of the algorithm is also deeply discussed.

(3) The algorithm description of TA-based top-k selection is presented.

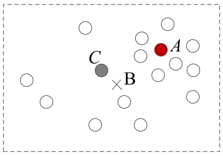


FIGURE 1. The difference from the typical object to the median and mean of a set of objects.

(4) The experiment part is enhanced by adding an evaluation of document semantic similarity measuring methods.

III. PROBLEM DEFINITION AND FRAMEWORK

This section first gives definitions (and/or descriptions) related to the spatial typicality keyword query, and then presents our solution framework.

A. DEFINITIONS

Definition 1 (Typicality): Given a set of spatial objects D with two attributes, *location* and *description*, the objects in D can be treated as a subset of samples of a two-dimensional random vector $\mathcal{Z}$ that takes values in the Cartesian product space of the domains with respect to attributes *location* and *description*. The typicality of an object $o \in D$ corresponds to $\mathcal{Z}$ is defined as $T(o, \mathcal{Z}) = L(o|\mathcal{Z})$ where $L(o|\mathcal{Z})$ is the likelihood of $o \in D$.

Example 1: As illustrated in [14], given a set of objects in Figure 1, suppose objects B and C are the mean and median of the set, respectively. It is clearly to see that the object A is more typical than the objects B and C since there are more objects around A closer than that around B and C. In this scenario, the object A is a good representative of the set rather than B and C.

Computing the probability density of an object is a good way to measure the confidence of the object that may appear compare to the others in the same object set. In this paper, we will use Gaussian kernel-based density estimation method to compute the $L(o|\mathcal{Z})$.

Definition 2: (Problem of top-k typicality and semantic query). Let q be a spatial keyword query over spatial dataset D. Based on the location-semantic relationships between query and spatial objects as well as the object's typicality, the goal is to address the top-k typicality and semantic query problem defined as,

$$\Gamma_k = argmax_{\Gamma'_k} \sum_{i=1}^{k(k<<n)} (Sim(q, o_i) + Typicality(o_i)) \quad (2)$$

where Γ_k is a set of k answer objects and n the number of all spatial objects in D, $Sim(q, o_i)$ represents the semantic similarity between f and o_i, Typicality(o_i) is the typicality of o_i. The objective of the problem is to find a set of k objects in D that are both typical and semantically related closely as possible to the given spatial keyword query.

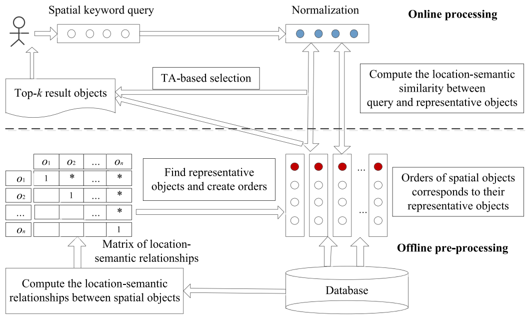

Spatial keyword query
Normalization
Online processing
TA-based selection
Top-*k* result objects
Compute the location-semantic similarity between query and representative objects
o_1
o_2
…
o_n
1
*
Find representative objects and create orders
Orders of spatial objects corresponds to their representative objects
Matrix of location-semantic relationships
Offline pre-processing
Compute the location-semantic relationships between spatial objects
Database

FIGURE 2. Framework of our top-*k* spatial keyword query approach.

B. FRAMEWORK

The framework of our solution is shown in Figure 2. It consists of the offline pre-processing and online processing steps. During the offline pre-processing step, the location-semantic relationship computing component calculates the location similarities and semantic similarities between all pairs of spatial objects in the dataset. These similarities are combined to form the matrix of location-semantic relationships of spatial objects; each element e_{ij} in the matrix represents the location-semantic relationship between object o_i and o_j. Based on the Matrix, the representative object finding and order creating component first picks a certain number of representative objects by using the Gaussian kernel-based probabilistic density estimation method from the entire dataset and then create orders of remaining objects corresponding to all representatives. These orders are maintained as the candidates for facilitating the top-k typical and relevant object selection.

During the online processing step, for a given spatial keyword query, the location-semantic proximity computing component first measures the location-semantic similarities between representative objects and query and then picks the representatives having the highest location-semantic similarities to the current query. After this, the orders corresponding to the selected representatives would be leveraged by threshold algorithm (TA) to select the top-k typical and relevant result objects.

IV. LOCATION-SEMANTIC RELATIONSHIP MEASURING

This section first describes the location similarity and semantic similarity measuring methods for spatial objects, respectively. For measuring the semantic similarity between documents associated to the spatial objects, we propose two measures, the keyword coupling relationship-based measure and the Word2Vec-CNN-based measure, respectively. Then, the location similarity and document similarity are combined to form a location-semantic relationship of spatial objects.

A. LOCATION SIMILARIY MEASURING

The location information of a spatial object is usually denoted by a pair of latitude and longitude. Therefore, we can straightforwardly use the *Euclidean* distance to measure the location distance between a pair of spatial objects according

TABLE 2. The Normalized Location Similarities between Spatial Objects.

	o_1	o_2	o_3	o_4	o_5
o_1	1.0000	0.9858	0.4343	0.4154	0.5640
o_2	0.9858	1.0000	0.4407	0.4039	0.5559
o_3	0.4343	0.4407	1.0000	0.2549	0.0000
o_4	0.4154	0.4039	0.2549	1.0000	0.2553
o_5	0.5640	0.5559	0.0000	0.2553	1.0000

to their geo-locations. Given two spatial objects, o_i and o_j, the Euclidean distance between them is generally defined as,

$$D(o_i, o_j) = \sum_{k=1}^{K} d(o_i^{(k)}, o_j^{(k)}) \tag{3}$$

where K is the spatial dimensions of the spatial objects.

Based on the Euclidean distance, the location similarity between spatial objects o_i and o_j can be computed as,

$$S_{Loc}(o_i, o_j) = 1 - D(o_i, o_j)/MaxD \tag{4}$$

where $MaxD$ is the maximum distance of all pairs of spatial objects which is used to normalize the spatial distance of all pairs of spatial objects into the interval [0, 1]. Table 2 shows the location similarities of spatial objects listed in Table 1.

B. KEYWORD COUPLING RELATIONSHIP-BASED DOCUMENT SEMANTIC SIMILARITY MEASURING

The semantic relevancy between a pair of spatial objects can be reflected by their textual document similarity in terms of semantics. The textual document associated to a spatial object is usually the short text containing the object name, amenities, special features, etc. The documents associated to all the spatial objects are formed a document set. We first use the short text segmentation tools such as AlchemyAPI or Jieba to extract all the distinct keywords from the document set and then each document can be represented by a keyword set. In this section, we first propose the keyword coupling relationship-based document semantic measuring method. The Word2Vec-CNN-based measure will be proposed in the following subsection.

1) KEYWORD COUPLING RELATIONSHIP MEASURING

The basic idea is to analyze the intra- and inter-correlation between a pair of keywords in the document set and then to combine these two kinds of correlations as the keyword coupling relationships.

We use a graph structure to represent the correlations between different keywords. Figure 3 shows a toy example of keyword relationship graph for the keyword $\{A, B, C\}$. Each node represents a distinct keyword while the edge means the directly correlation between two keywords. There would be an edge between two keywords if they co-occur in the same textual document.

As showed in Figure 3, the correlation between a pair of keywords can be divided into two categories, the intra-correlation and inter-correlation. Two nodes (such as nodes A and B) are intra-related if they are directly connected while two nodes (such as A and C) are inter-related if they are inter-connected through at least one common node. The weight on the edge denotes the normalized intra-correlation degree between two keywords. For example, the intra-correlation degree from A to B is 1 while that of which from B to A is 0.5. The reason is that the keywords A and B may have different correlations to the other keywords. Given a pair of keywords, the coupling relationship between them is combined by their intra- and inter-correlations.

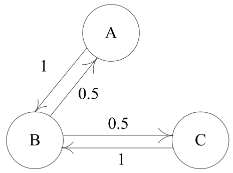


FIGURE 3. Keyword relationship graph.

a: KEYWORD INTRA-CORRELATION

The frequency of co-occurrence of a pair of keywords (t_i, t_j) appearing in the same documents can be measured by the Jaccard coefficient,

$$J(t_i, t_j) = \frac{|T(t_i) \cap T(t_j)|}{T(t_i) \cup T(t_j)|} \tag{5}$$

where T is the set of documents associated to all spatial objects, $T(t_i)$ and $T(t_j)$ represent the documents in which t_i and t_j appears, respectively.

Based on Equation (5), the intra-correlation between keywords t_i and t_j in T is defined,

$$\delta_{Intra}(t_i, t_j|T) = J(t_i, t_j) \tag{6}$$

Since t_i or t_j may also co-occur with other keywords in the same documents, we need to normalize the intra-correlations between t_i and t_j by dividing the total number of intra-correlations between t_i and other keywords, that is,

$$\delta_{Intra}(t_i, t_j) = \begin{cases} 1, & i = j \\ \dfrac{\delta_{Intra}(t_i, t_j|T)}{\sum_{k=1,k\neq i}^{n} \delta_{Intra}(t_i, t_k|T)}, & i \neq j \end{cases} \tag{7}$$

where n is the number of all distinct keywords in T.

For each pair of keywords t_i and t_j, we have $\delta_{Intra}(t_i, t_j) \geq 0$ and $\sum_{j=1,j\neq i}^{n} \delta_{Intra}(t_i, t_j) = 1$. Note that, $\delta_{Intra}(t_i, t_j)$ and $\delta_{Intra}(t_j, t_i)$ may not be equal to each other due to the different dominators.

b: KEYWORD INTER-CORRELATION

The keyword t_i and t_j are inter-related if there is at least one common keyword between them. The inter-correlation between t_i and t_j via their common keyword t_c is defined as,

$$\delta_{Inter}(t_i, t_j|t_c) = min\{\delta_{Intra}(t_i, t_c), \delta_{Intra}(t_j, t_c)\} \tag{8}$$

where $\delta_{Intra}(t_i, t_c)$ and $\delta_{Intra}(t_j, t_c)$ are the intra-correlation between t_i and t_c, t_j and t_c, respectively.

TABLE 3. **The Coupling Relationships between Keywords (Here we set $\beta = 0.2$).**

	A	B	C	D	E	F	G
A	1.0000	0.0790	0.1579	0.1053	0.0790	0.0790	0.0395
B	0.1364	1.0000	0.3636	0.0401	0.0148	0.0148	0.0000
C	0.1666	0.2222	1.0000	0.1111	0.0148	0.0148	0.0000
D	0.2500	0.0401	0.2500	1.0000	0.0148	0.0148	0.0000
E	0.0714	0.0148	0.0148	0.0148	1.0000	0.2857	0.1429
F	0.0714	0.0148	0.0148	0.0148	0.2857	1.0000	0.1429
G	0.0395	0.0000	0.0000	0.0000	0.2500	0.2500	1.0000

Since there is two or more common keywords between t_i and t_j and each one may have different importance in documents. We use the traditional *idf*() weighting function to measure the weight of the keyword in the document set and then use the maximum *idf* value to normalize the weight of each keyword.

After this, let S be the set of common keywords for t_i and t_j, the inter-correlation between t_i and t_j, inter-related by all the common keywords in S can be formalized as,

$$\delta_{Inter}(t_i, t_j) = \begin{cases} 1, & i = j \\ \dfrac{\sum_{\forall t_c \in S} w(t_c) * \delta_{Intra}(t_i, t_j|t_c)}{|S|}, & i \neq j \end{cases} \tag{9}$$

where $|S|$ denotes the number of common keywords between t_i and t_j in S. If $S = \Phi$, then $\delta_{Inter}(t_i, t_j) = 0$.

c: KEYWORD COUPLING RELATIONSHIP

The combination of intra- and inter-correlation between a pair of keywords (such as t_i and t_j) is called the keyword coupling relationship which is defined as follows,

$$\delta_{Coupling}(t_i, t_j) = \begin{cases} 1, & i = j \\ (1-\beta) * \delta_{Intra}(t_i, t_j) & \\ +\beta * \delta_{Inter}(t_i, t_j), & i \neq j \end{cases} \tag{10}$$

where $\beta \in [0, 1]$. Table 3 shows the coupling relationships between keywords extracted from Table 1. We here use A, B, C, D, E, F, and G to represents the keywords *swimming pool*, *WiFi*, *Breakfast*, *Subway*, *Conference*, *Internet*, and *Airport service*, respectively.

2) KEYWORD COUPLING RELATIONSHIP-BASED DOCUMENT SEMANTIC SIMILARITY MEASURING

Based on the keyword coupling relationships, we then use a kernel-based cosine similarity method to compute the semantic similarity between a pair of documents. The solution consists of the three steps.

Step 1: Convert the document into vector representation. Given a pair of documents d_1 and d_2, we assume K the set of all distinct keywords extracted from d_1 and d_2 and m the number of keywords in K. We also let $m = |K|$ and Δ be an fixed order on the keywords appearing in K. $K[i]$ refers to the i-th keyword of K based on the order Δ. After this, a vector representation of $d_1 = \wedge_j K[j] (j = 1, \ldots, m)$ is a vector of $\overrightarrow{d_1}$ size m. If $K[i]$ appears among the keywords of d_1 then $\overrightarrow{d_1}[i] = idf(K[i])$, otherwise it is 0.

TABLE 4. **The Document Semantic Similarities between Spatial Objects.**

	o_1	o_2	o_3	o_4	o_5
o_1	1.0000	0.9148	0.8425	0.4126	0.1121
o_2	0.9148	1.0000	0.6754	0.5525	0.0350
o_3	0.8425	0.6754	1.0000	0.4527	0.1148
o_4	0.4126	0.5525	0.4527	1.0000	0.8447
o_5	0.1121	0.0350	0.1148	0.8447	1.0000

Step 2: Construct the keyword coupling relationship matrix. Given a pair of documents d_1 and d_2, the coupling relationships of m distinct keywords in d_1 and d_2 can then be transformed into a Matrix M, which is a $m*m$ matrix and each element $M(i, j)$ in it corresponds to the coupling relationship between keywords t_i and t_j.

Step 3: Compute the kernel-based cosine similarity. The traditional cosine similarity measuring method ignores the coupling relationships between keywords in the compared documents. To address this shortcoming, based on the matrix M generated in Step 2, each document is transformed into a new vector $\overrightarrow{d}\,' = \overrightarrow{d}\,'M$, which enriches the document vector representation with the coupling relationships between keywords. Then, using this transformation the corresponding kernel [1] of two vector $\overrightarrow{d_1}$ and $\overrightarrow{d_2}$ can be written as,

$$k'(d_1, d_2) = \overrightarrow{d_1}(M^T M)\overrightarrow{d_2}^T \tag{11}$$

After this, we can define the document semantic similarity, which can be computed by using the kernel-based cosine similarity method as follows,

$$\begin{aligned} S_{Doc}(d_1, d_2) &= cos_{ker}(\overrightarrow{d_1}, \overrightarrow{d_2}) \\ &= \frac{k'(d_1, d_2)}{\sqrt{k'(d_1, d_1)}\sqrt{k'(d_2, d_2)}} \end{aligned} \tag{12}$$

By using the method proposed above, we can get the document semantic similarities (showed in Table 4) between the spatial objects in Table 1.

The keyword coupling relationship-based document semantic similarity measuring method considers both the keyword intra- and inter-couplings in the documents, which makes the document semantic similarity more reasonability than the traditional similarity measures (e.g., TFIDF). In the next section, we will propose an alternative for measuring the document semantic similarity by learning the latent features of documents.

C. WORD2VEC-CNN-BASED DOCUMENT SEMANTIC SIMILARITY MEASURING

Word embedding technique (such as Word2Vec) is very successful in the natural language processing (NLP) while convolutional neural network (CNN) is effective for processing the images. We borrow and combine these two techniques to measure the semantic similarities between documents. The solution consists of the following three steps.

Step 1 (Transform a Keyword Into Its Vector Representation): We leverage the Skip-gram model to train the word

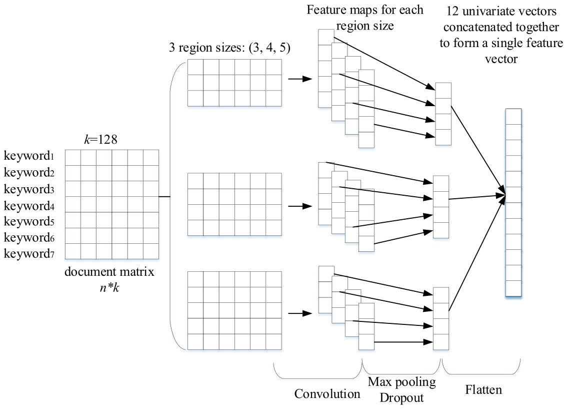

Feature maps for each region size
12 univariate vectors concatenated together to form a single feature vector
3 region sizes: (3, 4, 5)
k=128
keyword1
keyword2
keyword3
keyword4
keyword5
keyword6
keyword7
document matrix
n*k
Convolution
Max pooling
Dropout
Flatten

FIGURE 4. Word2Vec-CNN-based document feature vector learning model.

vector [22], [36]. The vocabulary of words is the keywords in the document set associated to the spatial objects. After the training process, each distinct keyword/phrase in the document set can be transformed to be a 128-dimension vector.

Step 2 (Transform a Document Into Its Matrix Representation): Based on the keyword vector representation generated in step 1, we use a $n*k$ matrix to represent a document. Here, n is the largest number of keywords in the document of the document set, k is the dimension of the keyword vector (in our experiment, $k = 128$). Note that, the length of each document is fixed to n. For the document whose length is less than n, we adopt the 0-vector to fill in the matrix. In other words, each document in the document set should be a represented by a $n * k$ matrix.

Step 3 (Transform a Document Into Its Vector Representation): In this step, we first design a Convolutional Neural Network (CNN) which contains convolutional layer, pooling layer, dropout layer, and flatten layer, the structure of which is showed in Figure 4. In this network, we set the filter size of (3, 3), (4, 4), (5, 5) respectively to do the convolution operation (the filter moving step is set to 1), and use the *ReLU* as the activation function. Then, we take the matrix corresponding to each document as the input of CNN. After processing of convolution layer, pooling layer, Dropout layer, and Flatten layer, the feature/latent vector of a document can be lastly obtained. In our experiments, we empirically set the parameter of dropout layer is to 0.2.

Step 4 (Compute the Semantic Similarity for a Pair of Documents): The output of step 3 is the vector representation of a document. Given a pair of documents, we can take Cosine similarity measure to compute the similarity based on their corresponding vector representations.

D. LOCATION-SEMENTIC RELATIONSHIP MEASURING FOR SPATIAL OBJECTS

Based on the location similarity and the document semantic similarity, we can then obtain the location-semantic relationship for each pair of spatial objects by linearly combining these two similarities, i.e.,

$$Sim_{LD}(o_i, o_j) = \begin{cases} 1, & i = j \\ \quad (1-\lambda)S_{Loc}(o_i, o_j) & \\ +(1-\lambda)S_{Doc}(o_i, o_j), & i \neq j \end{cases} \tag{13}$$

where $\lambda \in [0, 1]$ is the parameter to determine the weight of location similarity and semantic similarity in the location-semantic relationship measuring.

V. TOP-*K* TYPICAL AND RELEVANT OBJECT SELECTION

This section first discusses the challenges of top-k typical and relevant object selection problem, and then presents an approximation approach.

A. TOP-K SELECTION PROBLEM

As mentioned above, given a spatial keyword query q over spatial dataset D, the objective of the top-k typical and relevant object selection problem is to find a set of number k objects in D that are both typical and semantically related closely as possible to the given spatial keyword query.

The top-k typical and relevant object selection may be applied on large datasets, however, computing the exact answer set with the both highest typicality and semantic relevance to the given query needs quadratic time which is too costly for online query answering. Thus, it is necessary to develop an approximation algorithm which can rapidly provide the good approximations of exact answers.

B. APPROACH

This paper proposes an approximation approach, which consists of three steps, to resolve the top-k typical and relevant object selection problem over large datasets. The first step is to find a certain number of representative objects from the spatial database, and the second step is to order the remaining objects corresponds to each representative objects. The third step is to select the top-k typical and relevant objects based on these orders. The first and second steps are processed during offline phase, and the third step is processed in online phase.

Step 1 (Find Representative Objects): Based on the location-semantic relationships between different pairs of spatial objects, we provide a method, which is inspired by the typicality estimation algorithm proposed in [14], to find the representative objects. As pointed out in [14], the object that has the highest probability density can be selected as the representative/typical object for the set of closely related objects. Since Gaussian kernel function is commonly used in density estimation and is essential in typicality computation, it can be applied to compute the representative objects in this paper. Given a set of spatial objects $D = (o_1, o_2, \ldots, o_n)$, the probability density function $f(o)$ is approximated as follows,

$$f(o) = \frac{1}{n}\sum_{i=1}^{n} G_h(o, o_i) = \frac{1}{n\sqrt{2\pi}}\sum_{i=1}^{n} e^{-\frac{d(o,o_i)^2}{2h^2}} \tag{14}$$

where $d(o, o_i)^2$ is the location-semantic distance between spatial objects o and o_i, $G_h(o, o_i) = \frac{1}{n\sqrt{2\pi}} \sum_{i=1}^{n} e^{-\frac{d(o,o_i)^2}{2h^2}}$ is the Gaussian kernel (here, $h = 1.06\sigma^{-1/5}$ and σ is the standard deviation of the location-semantic distances between all pairs of spatial objects).

We then use the probabilistic density estimation to find the representative objects from the given spatial object dataset D. The procedure is described as follows.

i) randomly partitions D into several groups, each of which contains u spatial objects, and thus there would be n/u groups.

ii) computes the typicality of each object in the group by using formula (14) and then finds the object having the highest probability in each group. These objects are formed into a new group N.

iii) for the group N, repeats the step (i) and (ii) until the only one winner is obtained. The winner is treated as a candidate of representative objects.

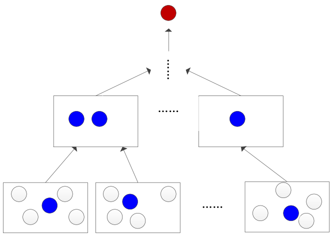


FIGURE 5. The strategy for finding the candidate of representative objects.

Figure 5 shows the processing procedure for finding the candidate of representative objects.

To guarantee the precision of representative object selection as much as possible, we repeat the above procedure v times and thus there would be v objects in the candidate set of representative objects. We then compute the typicality for each object in the candidate set in the scale of dataset D and then pick the object with the highest typicality as the representative.

To obtain the l representative spatial objects, the whole procedure described above should be repeated l times and thus the complexity of the algorithm is $O(lvun)$.

After this, when a user enters a spatial query, it needs to only compute the location proximity and semantic relevancy between the given query and representative object, which is used as a weighting parameter for top-k result selection.

Step 2 (Create Orders for Representative Objects): For each representative object $\overline{o}_i$, create an order τ_i of all remaining objects in D (except $\overline{o}_i$) in descending order, according to their location-semantic relationships to $\overline{o}_i$. The output of this step is a set of l orders. According to the output orders, each object o_j has a score that is associated with the position of o_j in each τ_i. The score of o_j in τ_i that corresponds to $\overline{o}_i$ is,

$$s(o_j|\overline{o}_i) = n - \tau_i(o_j) + 1 \tag{15}$$

where $\tau_i(o_j)$ represents the position of o_j in τ_i.

Step 3 (Select Top-k Typical Relevant Object): For a given spatial keyword query q, using the output of step 2, this step computes the set $q_k(D) \subseteq D$ with $|q_k(D)| = k$, such that $\forall q_j \in q_k(D)$ and $q_j' \in \{D - q_k(D)\}$ it holds that $score(o_j, q) > socre(o_j', q)$ with $score(o_j, q) = \sum_{i=1}^{l}(Sim_{LD}(q, \overline{o}_i)s(o_j|\overline{o}_i))$.

The Threshold Algorithm (TA) is employed to quickly evaluate the top-k objects for a given query [10]. The TA uses *Sorted* and *Random* modes to access the objects in the orders. The *Sorted* mode obtains the score of an object in an order by scanning the order of the objects from the top to down sequentially. The *Random* mode finds the score of an object in an order in one access by using an in-memory index structure. The in-memory index structure is a $n*l$ array, where n denotes the number of all objects and l represents the number of orders. The array stores the scores of all objects corresponding to different orders. For example, the element of $array[j][i]$ (where, $j = 1, \ldots, n, i = 1, \ldots, l$) represents the score of the j-th object in the i-th order. Clearly, when an object ID accessed by *Sorted* mode, the corresponding score of the object can be directly found in other orders by *Random* mode using the index.

Example 2: Figure 6 shows a toy example to illustrate the top-k selection processing procedure. In Figure 6(a), there are 5 objects in total and o_1, o_2, and o_3 are selected as the representatives, each of which corresponding to an order consists of the remaining objects (ranked according to their location-semantic relationships to the representative object) and their corresponding scores. Figure 6(b) shows the index array built for *Random* mode according to the orders in Figure 6(a).

The top-k result selection algorithm is shown in Algorithm 1, where the score of object o_j found in each order τ_i to query q is defined as:

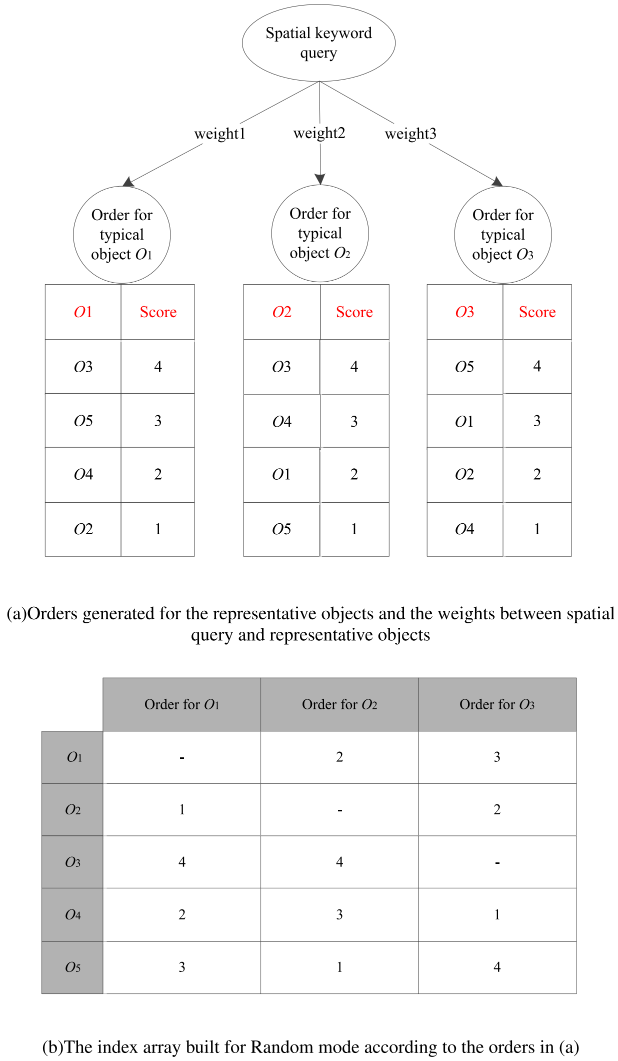
$$s(o_j, q) = Sim_{LD}(q, \overline{o}_i)s(o_j|\overline{o}_i) \tag{16}$$

where, the first factor is measured by Equation (13) and the second factor is computed by Equation (15). Note that, we use $s(o_j, q)$ to approximate the integration score of typicality and relevance of o_j to q. That means the object that is both relevant to the query and typical in the entire dataset would be scored high; on the contrary, the object having high relevancy to the query and low typicality in the dataset (or verse) may not get a high score.

The score of o_j in every other order can be found by using random access mode and all these scores are summed, resulting in the final score of o_j for given query q:

$$score(o_j, q) = \sum_{i=1}^{l} Sim_{LD}(q, \overline{o}_i)s(o_j|\overline{o}_i) \tag{17}$$

The termination criterion guarantees that no more retrieving object operations will be needed on any of the orders. This is accomplished by maintaining an array L which contains the scores of the last visited objects from all the orders by the end of the round-robin cycle. The sum of the scores in L represents the score of the very best object we hope to find


Spatial keyword query
weight1
weight2
weight3
Order for typical object O1
Order for typical object O2
Order for typical object O3

O1	Score
O3	4
O5	3
O4	2
O2	1

O2	Score
O3	4
O4	3
O1	2
O5	1

O3	Score
O5	4
O1	3
O2	2
O4	1

(a)Orders generated for the representative objects and the weights between spatial query and representative objects

	Order for O_1	Order for O_2	Order for O_3
O_1	-	2	3
O_2	1	-	2
O_3	4	4	-
O_4	2	3	1
O_5	3	1	4

(b)The index array built for Random mode according to the orders in (a)

FIGURE 6. TA-based top-*k* result selection processing.

in the data that is yet to be seen. If this value is no more that the object in the top-k buffer with the smallest score, the algorithm successfully terminates.

It should be pointed out that, to make sure the objects in the top-k answer set are relevant to the query as much as possible, we first select a certain number (much less than the number of all objects) of representative objects from the entire dataset, and then reserve only a small number of representatives that are closest to the given query. These reserved representatives and their corresponding orders are finally leveraged by TA to find the top-k result objects. The time complexity of algorithm 1 is $O(Ckl \log n)$, where n is the number of objects in the dataset, C is the number of representative objects selected from the entire dataset, l is the number of reserved representative objects, and k is the number of retrieved results.

Clearly, the result objects would be more typical than that obtained by traditional similarity-based query processing models since the top-k result objects selected by TA

Algorithm 1 The Top-k Selection Algorithm

```
Input: Order set $T_l = \{\tau_1, \dots, \tau_l\}$, spaitial keyword
       query $q$, number $k$.
Output: Top-$k$ typical relevant objects.
Let $B$ ={} be a buffer that can hold $k$ spatial objects
Let $L$ be an $l$ size array that is used to store the score the
last visited object of each order by the end of the current
round-robin cycle
repeat
for all $i \in \{1, \dots, l\}$ do
    Retrieve next object $o_j$ from $\tau_i$
    Compute $score(o_j, q) = Sim_{LD}(q, \overline{o}_i)s(o_j|\overline{o}_i)$ as $o_j$'s
    score
    Update $L[i]$ with score of $o_j$ in $\tau_i$;
    Get score of $o_j$ from other orders $\{\tau_k | \tau_k \in T_l$ and
    $k \neq i\}$ via random access
    $score(oj, q) \leftarrow$ summing up of all the retrieved
    scores of $o_j$ retrieved from all the orders
    Insert $\langle oj, score(o_j, q)\rangle$ into $B$ in descending order.
    $\lambda = \lambda + L[i]$
until $B[k-1].score \geq \lambda$
return $B$
```

are based on the orders corresponding to representative objects. We will next compare the efficiency and performance of our top-k typical relevant object selection method and the traditional spatial keyword querying method in experiments.

VI. EXPERIMENTS

This section introduces the experimental setup and reports the experimental results.

A. EXPERIMENTAL SETUP

The experiments are conducted on a computer running Windows 2010 with Intel i5-6300HQ 2.30GHz CPU, and 8GB of RAM. All algorithms are developed by Java.

Dataset: For our evaluation, we setup a real dataset containing 50,000 POIs extracted from Yelp [37]. Each POI has location and textual information. The location information is represented by longitude and latitude while the textual information is described by the *name* and *amenities* of POI. For the textual information, we first use the existing text segmentation and analysis tools such as AlchemyAPI, Jieba to extract the keywords and then transform the text document in to a set of keywords with associated *idf* weights. Table 5 lists the dataset properties.

B. ACCURACY OF DOCUMENT SEMANTIC SIMILARITIES

This experiment aims to evaluate the accuracy of our document semantic similarity measuring methods. To do this, we adopt the following user study strategy. We invited 100 students and they were partitioned into 10 groups. Each

TABLE 5. Properties of the Test Dataset.

Property	Yelp dataset
Total number of objects	50,000
Average number of unique keywords per object	7
Total number of unique keywords in dataset	89,648
Total number of keywords in dataset	627,536

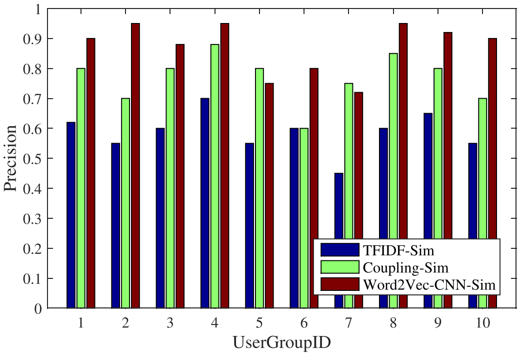


FIGURE 7. Accuracy of answers for Word2Vec-CNN-Sim, Coupling-Sim, and TFIDF-Sim.

group chose one document from the spatial dataset. For each selected document d_i, we generated a set D_i of 30 documents from dataset that are likely to contain a good mix of relevant and irrelevant documents in relation to the given document. Each set D_i is formed by mixing the top 10 documents returned by each method of Word2Vec-CNN-based similarity measure (short for Word2Vec-CNN-Sim), Keyword Coupling Relationship-based similarity measure (short for Coupling-Sim), and traditional TFIDF based similarity measure (short for TFIDF-Sim). Note that, if there were overlapped documents in D_i, we deleted them and randomly added some other documents into D_i. We empirically set the parameter $\beta = 0.2$ (resp. $\lambda = 0.5$) in Equation (10) (resp. Equation (13)) for measuring the keyword coupling relationship (resp. document similarity). Lastly, we presented the documents with their corresponding D_i's to each user group in our study. Each user group had to reach a consensus on marking the top 10 documents in D_i that are considered semantically related to d_i. We then measured how closely the 10 documents marked as relevant by the user group matched the 10 documents returned by each method.

The *Precision* metric is used to evaluate this overlap. Figure 7 shows the Precision of answers for Word2Vec-CNN-Sim, Coupling-Sim, and TFIDF-Sim, respectively. It can be seen that Word2Vec-CNN-Sim outperforms Coupling-Sim and TFIDF-Sim behaves the worst. The averaged Precision of Word2Vec-CNN-Sim and Coupling-Sim is 87.2% and 76.8%, respectively, while the TFIDF-Sim is 58.7%.

It also can be seen that, although the averaged precision of Word2Vec-CNN-Sim is higher than that of Coupling-Sim, the Coupling-Sim outperforms the Word2Vec-CNN-Sim in a few cases (user group 5 and user group 7). The reason is that the Word2Vec measure supposes the nearby/adjacent words/phrases (in a fixed window size) usually having the strong contextual relations while it is inefficient for capturing the intra-correlations between keywords that are far from each other in the same document and the inter-correlations between keywords across the different documents. In contrast, the keyword coupling relationship measuring method is independent of the sequence of the words/phrase in the documents, it considers both the frequency of co-occurrence of the keywords within the same documents and the inter-correlations between the keywords across the different documents. Therefore, in our following experiments, we will mainly use Word2Vec-CNN-Sim to measure the document semantic similarity and then leverage Coupling-Sim to adjust the special cases.

C. PRECISION OF TOP-K SELECTION ALGORITHM

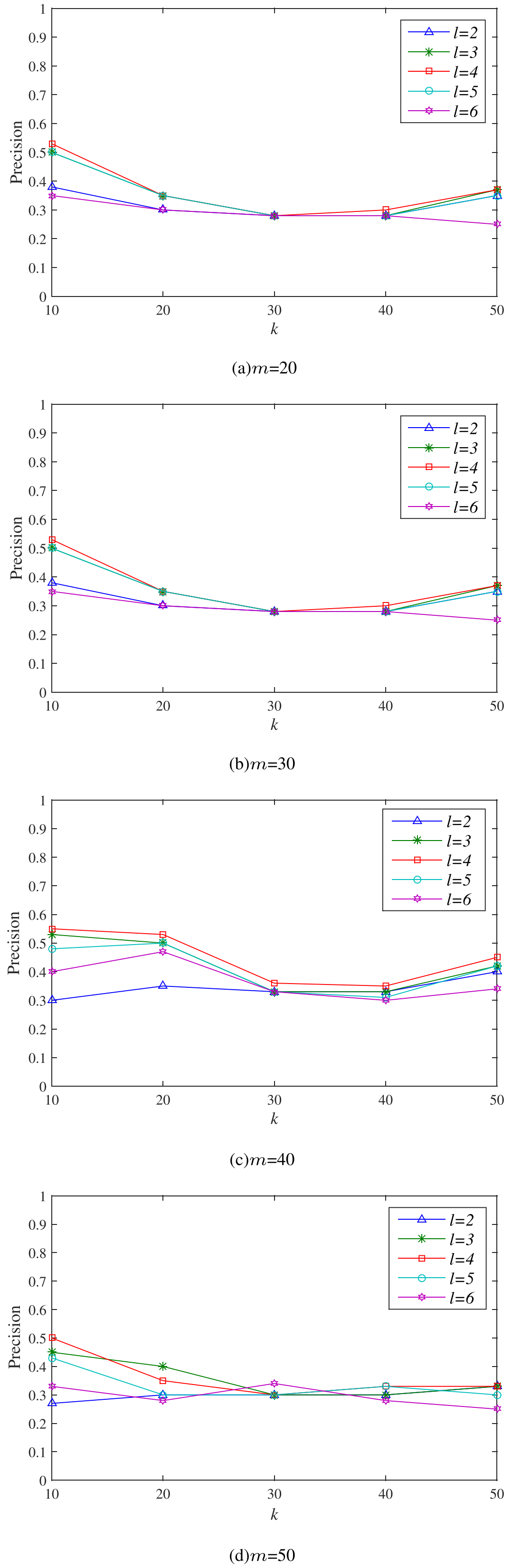
This experiment aims to test the precision of the top-k answer objects obtained by using our TA-based top-k selection algorithm over the orders that are generated in the offline time when compared with the top-k answer objects obtained by computing the location-semantic relationships of the given query to all spatial objects. To quantify this precision, we use $R(All, k)$ to denote the top-k objects returned by computing the location-semantic relevancy of the given query to all objects in the dataset, and $R(Rep, k)$ to denote the top-k objects returned using our top-k selection algorithm. The overlap of two top-k answer sets is measured using the Jaccard coefficient:

$$J(R(Rep, k), R(All, k)) = \frac{|R(Rep, k) \cap R(All, k)|}{|R(Rep, k) \cup R(All, k)|} \quad (18)$$

The coefficient falls into [0, 1] and the higher its value the more similar the two sets of answer objects are.

In this experiment, we randomly pick 10 objects from the Yelp dataset and then extract $2 \sim 4$ keywords from the text document associated to each object. The location information and extracted keywords of the selected object are combined to form a spatial keyword query. Furthermore, we use four parameters: n, m, l, and k, to character the dataset. Here, n is the number of spatial objects in the dataset, m the number of representative objects found from the dataset during the offline processing step, $l(l < m)$ the number of selected representative objects having the highest location-semantic relationships to the given query, and k the number of objects needs to be retrieved. Figure 8 shows the value of the coefficients (averaged over 10 test queries) for different values of k, when $l = \{2, 3, 4, 5, 6\}$ for each number of m ($m = \{20, 30, 40, 50\}$). The values of n are fixed to 50,000 (since there are 50,000 objects in Yelp dataset), and k is varied in $\{10, 20, 30, 40, 50\}$.

From Figure 8 we can see the overall coefficients (i.e., precision) of top-10 results corresponding to $m = 40$ is higher compared to that of other m. Furthermore, for the case of $m = 40$, the precision corresponding to different numbers of l are nearly identical when $l = \{3, 4, 5\}$, and especially the precision of top-10 results achieves 55% when $l = 4$, which means that when only a small number of orders are used to



(a)m=20

(b)m=30

(c)m=40

(d)m=50

FIGURE 8. Precision of our top-*k* selection method for different *l* and *m* when *k* varied.

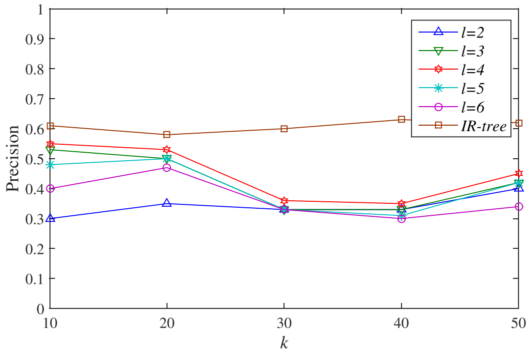


FIGURE 9. Comparison on Precision of IR-tree-based top-*k* selection method and our TA-based top-*k* selection method for different *l* when value *k* varied.

find the top-10 related typical objects, the information lost by looking at the orders of representative objects instead of computing location-semantic relationships of the given query to all objects in dataset is acceptable. It also can be seen that, the tendency of precision is not raised as the increase of number l (such as the precision corresponding to $l = 6$ is much lower than that of $l = \{3, 4, 5\}$ for each number of m on the test dataset). The reason is that the larger number of l the more orders which are inconsistent with the order of top-k result objects would be added into the TA scan list, and thus makes the precision becoming worse.

In this experiment, we also compared the precision of our TA-based top-k selection method with IR-tree-based top-k selection method. IR-tree is an efficient spatial keyword query index schema. It first builds the spatial index, and then, builds the text index for the objects in each group partitioned by the spatial index. Since IR-tree index cannot deal with the semantic querying, we compute the location-text relationship (rather than location-semantic relationship) for each pair of objects in the dataset by using the $tf \cdot idf$ weighting function and Cosine similarity for the text similarity measuring and then use TA algorithm to choose the top-k typical relevant objects. Figure 9 shows the comparison of the precision of our method and IR-tree-based top-k selection method. Note that we fix the number of m to 40 in this experiment since this value of m makes our method reaching the best performance.

It can be seen that the precision of IR-tree based top-k selection method steadily outperforms our method, the averaged precision from top 10 to 50 is 61%; While, our method achieves the best average precision (45%) from top 10 to 50 and best precision of top-10 results is 55% when $l = 4$. Although the precision of our method is lower than that of IR-tree index, our aim is to find both the typical and relevant objects to the given query with a certain precision. We next test the typicality of top-k objects returned by our method.

D. TYPICALITY OF THE TOP-K ANSWER OBJECTS

This experiment aims to verify the typicality of the top-k answer objects returned by using our method and

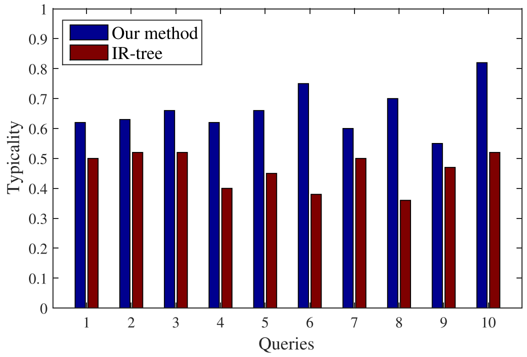


FIGURE 10. Comparison for the typicality of the top-*k* results returned by using our method and IR-tree-based top-*k* selection.

IR-tree-based top-k selection method, respectively. The measuring criteria of the typicality of top-k results is showed as follows,

$$Typicality(T) = \frac{\sum_{i=1}^{k} f(o_i)}{k} \qquad (19)$$

where T is the set of top-k answer objects, k is the number of objects needs to be retrieved, o_i is in T and $f(o_i)$ is computed by Equation (14). In this experiment, we set $k = 10$. To determine the candidate set for computing the object typicality by using Equation (19), we first obtain the union set of top-10 objects that are returned by using both our method and IR-tree-based top-k selection method, and then choose an object from the set whose location-semantic relationship to the given query is the furthest. After this, we take the location-semantic relationship of the selected object to the given query as a threshold, and then to obtain the candidate set of the objects having location-semantic relationships to the given query greater than the threshold. The higher the typicality of the top-k results indicates the more representative of the answer objects over the candidate set, and thus can improve the user recognition of the entire candidate set more efficiently. We take the typicality of the top-10 most typical objects from the candidate set as the baseline, and then compute the typicality of top-k results returned by IR-tree-based top-k selection and our method, respectively. Figure 9 shows the comparison of the typicality of top-10 answers returned by our method (resp. IR-tree-based top-k selection) for over 10 test queries.

From Figure 10, the typicality of top-10 results returned by using our method steadily outperforms the IR-tree index, and the averaged normalized typicality of top-10 results obtained by our method and IR-tree based method are 66% and 46%, respectively. The reason is that our method leverages TA and the orders corresponding to representative objects to find the top-k result and thus the typicality of the top-10 results is high. In contrast, IR-tree index uses Equation (1) to select and rank the results which usually makes the top-k results are similar to each other without the diversity and typicality. By integrating the comparison results of Figure 9 and Figure 10, we found that our method can achieve high typicality with a relative high precision, which can satisfy the user's needs in relevancy and typicality for the top-k results.

TABLE 6. Properties of the Test Dataset.

Datasets	10000 objects		30000 objects		50000 objects	
k	TA	R-tree	TA	IR-tree	TA	IR-tree
10	62	187	194	343	308	454
20	86	220	269	36	380	518
30	92	228	333	394	440	494
40	106	221	363	391	480	500
50	115	246	397	406	536	540

E. PERFORMANCE

This experiment aims to verify the performance of our TA-based top-k selection algorithm compared with IR-tree based top-k selection algorithm. We generate three different sizes of datasets which contain 10,000, 30,000, and 50,000 spatial objects, respectively. For the TA-based top-k selection algorithm, we fixed the number of l (i.e., the number of orders TA should be scanned) to 4 since our algorithm achieved the best precision when $l = 4$ which has been identified in the Experiment C. Based on these datasets, we test the execution time of our TA-based top-k selection algorithm and IR-tree based top-k selection algorithm for different k values ($k = \{10, 20, 30, 40, 50\}$). Table 6 presents the execution time (ms) of the two algorithms over datasets for different k values.

From Table 6 we can see that our method runs faster than IR-tree index over the datasets for different values of k. This demonstrated that our method can not only obtain the typical and relevant answers but also has better performance comparing with the IR-tree index, which indicates our algorithm can be well suitable for processing the large scale of dataset.

VII. CONCLUSION

This paper proposed a top-k spatial keyword query approach to address the typicality and semantic query problem. Our approach differs from the state-of-the art approaches in two aspects: (1) our approach considers the semantic relevancy between textual documents associated to spatial objects, especially we propose two document similarity measures, the keyword coupling relationship-based similarity measure and Word2Vec-CNN-based similarity measure, and the former is a good complement for the latter; (2) the probability density-based evaluation method is used to find the representative objects and the TA-based algorithm is leveraged to facilitate the top-k typical and semantically related answer object selection.

In the future, we will investigate how to analyze user temporal-spatio behaviors and to incorporate deep learning techniques for dealing with the personalization and diversity querying issues of spatial keyword query.

REFERENCES

[1] L. Alsumait and C. Dormeniconi, "Text clustering with local semantic kernels," in *Survey of Text Mining II*. London, U.K.: Springer, Apr. 2008, pp. 87–105.

[2] H. K.-H. Chan, C. Long, and R. C.-W. Wong, "On generalizing collective spatial keyword queries," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1712–1726, Sep. 2018.

[3] L. Chen, Y. Li, J. Xu, and C. S. Jensen, "Towards why-not spatial keyword top-κ queries: A direction-aware approach," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 4, pp. 796–809, Apr. 2018.

[4] L. Chen, X. Lin, H. Hu, C. S. Jensen, and J. Xu, "Answering why-not questions on spatial keyword top-κ queries," in *Proc. IEEE 31st Int. Conf. Data Eng.*, Apr. 2015, pp. 279–290.

[5] G. Cong and C. S. Jensen, "Querying geo-textual data: Spatial keyword queries and beyond," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Jun. 2016, pp. 2207–2212.

[6] G. Cong, C. S. Jensen, and D. Wu, "Efficient retrieval of the top-κ most relevant spatial Web objects," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 337–348, Jan. 2009.

[7] F. M. Choudhury, J. S. Culpepper, Z. F. Bao, and T. Sellis, "Batch processing of top-κ spatial-textual queries," *ACM Trans. Spatial Algorithms Syst.*, vol. 3, no. 4, pp. 13:1–13:40, Mar. 2018.

[8] Y. Dong, H. Chen, and H. Kitagawa, "Continuous search on dynamic spatial keyword objects," in *Proc. 35th Int. Conf. Data Eng.*, Apr. 2019, pp. 1578–1581.

[9] I. De Felipe, V. Hristidis, and N. Rishe, "Keyword search on spatial databases," in *Proc. IEEE 24th Int. Conf. Data Eng.*, Apr. 2008, pp. 656–665.

[10] R. Fagin, A. Lotem, and M. Naor, "Optimal aggregation algorithms for middleware," *J. Comput. Syst. Sci.*, vol. 66, no. 4, pp. 614–656, Jun. 2003.

[11] Y. Gao, J. Zhao, B. Zheng, and G. Chen, "Efficient collective spatial keyword query processing on road networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 2, pp. 469–480, Feb. 2016.

[12] H.-J. Hong, G.-M. Chiu, and W.-Y. Tsai, "A single quadtree-based algorithm for top-κ spatial keyword query," *Pervasive Mobile Comput.*, vol. 42, pp. 93–107, Dec. 2017.

[13] W. Hua, Z. Wang, H. Wang, K. Zheng, and X. Zhou, "Short text understanding through lexical-semantic analysis," in *Proc. IEEE 31st Int. Conf. Data Eng.*, Apr. 2015, pp. 495–506.

[14] M. Hua, J. Pei, A. W. C. Fu, X. Lin, and H.-F. Leung, "Top-κ typicality queries and efficient query answering methods on large databases," *VLDB J.*, vol. 18, pp. 809–835, Dec. 2009.

[15] L. Jing, M. K. Ng, and J. Z. Huang, "Knowledge-based vector space model for text clustering," *Knowl. Inf. Syst.*, vol. 25, no. 1, pp. 35–55, Jan. 2010.

[16] H.-J. Kim, J. Kim, J. Kim, and P. Lim, "Towards perfect text classification with Wikipedia-based semantic Naïve Bayes learning," *Neurocomputing*, vol. 315, pp. 128–134, Nov. 2018.

[17] Y. Lu, J. Lu, and C. Shahabi, "Efficient algorithms and cost models for reverse spatial-keyword K-nearest neighbor search," *ACM Trans. Database Syst.*, vol. 39, no. 2, pp. 573–598, May 2014.

[18] G. L. Li, J. Xu, and J. H. Feng, "Keyword-based k-nearest neighbor search in spatial databases," in *Proc. 21st ACM Conf. Inf. Knowl. Manage.*, Nov. 2012, pp. 2144–2148.

[19] Z. Li, K. C. K. Lee, B. Zheng, W.-C. Lee, D.-C. Lee, and X. Wang, "IR-tree: An efficient index for geographic document search," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 4, pp. 585–599, Apr. 2011.

[20] A. R. Mahmood and W. G. Aref, "Query processing techniques for big spatial-keyword data," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, May 2017, pp. 1777–1782.

[21] R. Mehrotra, S. Sanner, W. L. Buntine, and L. Xie, "Improving LDA topic models for microblogs via tweet pooling and automatic labeling," in *Proc. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2013, pp. 889–892.

[22] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proc. 7th Int. Conf. Learn. Represent.*, Mar. 2013, pp. 1–9.

[23] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, Sep. 2013, pp. 3111–3119.

[24] X. Meng, X. Zhang, L. Li, Q. Zhang, and P. Li, "Top-κ spatial keyword quer with typicality and semantics," in *Proc. Int. 24th Conf. Database Syst. Adv. Appl.*, Apr. 2019, pp. 244–248.

[25] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors forward representation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Oct. 2014, pp. 121–129.

[26] S. Park and S. Park, "Reverse collective spatial keyword query processing on road networks with G-tree index structure," *Inf. Syst.*, vol. 84, pp. 49–62, Sep. 2019.

[27] J. Qi, R. Zhang, C. S. Jensen, K. Ramamohanarao, and J. He, "Continuous spatial query processing: A survey of safe region based techniques," *ACM Comput. Surv.*, vol. 51, no. 3, Mar. 2018, Art. no. 64.

[28] J. Rocha-Junior, O. Gkorgkas, S. Jonassen, and K. Nϕåg, "Efficient processing of top-κ spatial keyword queries," in *Proc. 7th Int. Symp. Spatial Temporal Databases*, Jun. 2011, pp. 205–222.

[29] C. Salgado, M. A. Cheema, and M. E. Ali, "Continuous monitoring of range spatial keyword query over moving objects," *World Wide Web*, vol. 21, no. 3, pp. 687–712, Mar. 2018.

[30] X. Wang, Y. Zhang, W. Zhang, X. Lin, and Z. Huang, "SKYPE: Top-κ spatial-keyword publish/subscribe over sliding window," *Proc. VLDB Endowment*, vol. 9, no. 7, pp. 588–599, Jul. 2016.

[31] J. Wood, P. Tan, W. Wang, and C. Arnold, "Source-LDA: Enhancing probabilistic topic models using prior knowledge sources," in *Proc. Int. Conf. Data Eng.*, Apr. 2017, pp. 411–422.

[32] Y. Wu, J. Xu, L. Tu, M. Luo, Z. Chen, and N. Zheng, "Reverse collective spatial keyword querying," in *Proc. Int. Conf. Collaborative Comput.*, Dec. 2018, pp. 211–221.

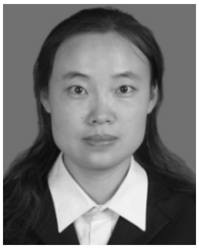
[33] D. Zhang, Y. M. Chee, A. Mondal, A. K. H. Tung, and M. Kitsuregawa, "Keyword search in spatial databases: Towards searching by document," in *Proc. 25th Int. Conf. Data Eng.*, Mar./Apr. 2009, pp. 688–699.

[34] C. Zhang, Y. Zhang, W. Zhang, and X. Lin, "Inverted linear quadtree: Efficient top κ spatial keyword search," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 7, pp. 1706–1721, Jul. 2016.

[35] K. Zheng, H. Su, B. Zheng, S. Shang, J. Xu, J. Liu, and X. Zhou, "Interactive top-κ spatial keyword queries," in *Proc. 31st Int. Conf. Data Eng.*, Apr. 2015, pp. 423–434.

[36] Z. Zhang and P. Zweigenbaum, "GNEG: Graph-based negative sampling for word2vec," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, Jul. 2018, pp. 566–571.

[37] *Yelp*. Accessed: Dec. 2017. [Online]. Available: https://www.yelp.com/dataset



XIAOYAN ZHANG was born in 1983. She received the M.S. degree from Northeastern University, China. She is currently pursuing the Ph.D. degree with Liaoning Technical University, China. Her research interests include spatial data analysis, city computing, and deep learning.

XIANGFU MENG was born in 1981. He received the Ph.D. degree from Northeastern University, China, in 2010. He is currently a Full Professor and a Ph.D. Supervisor with Liaoning Technical University, China. His research interests include spatial data management, recommender systems, and Web database query.

JINGUANG SUN was born in 1962. She received the Ph.D. degree from Liaoning Technical University, China, in 2006, where she is currently a Full Professor and a Ph.D. Supervisor. Her research interests include image processing, computer graphics, deep learning, and spatio-temporal data management.

QUANGUI ZHANG was born in 1978. He received the Ph.D. degree from the Beijing University of Technology. He is currently an Associate Professor with the School of Electronic and Information Engineering, Liaoning Technical University, Huludao, China. His current research interests include deep learning and recommended systems.

PAN LI was born in 1996. She is currently pursuing the master's degree with Liaoning Technical University, China. Her research interests include spatial keyword query and spatial recommendation systems.