Answering Top-k Similar Region Queries[⋆]

Chang Sheng[1], Yu Zheng[2] Wynne Hsu[1], Mong Li Lee[1], Xing Xie[2]

[1]School of Computing, National University of Singapore, Singapore
{shengcha,whsu,leeml}@comp.nus.edu.sg
[2]Microsoft Research Asia, Beijing, China
{yuzheng,Xing.Xie}@microsoft.com

Abstract. Advances in web technology have given rise to new information retrieval applications. In this paper, we present a model for geographical region search and call this class of query *similar region query*. Given a spatial map and a query region, a similar region search aims to find the top-k most similar regions to the query region on the spatial map. We design a quadtree based algorithm to access the spatial map at different resolution levels. The proposed search technique utilizes a filter-and-refine manner to prune regions that are not likely to be part of the top-k results, and refine the remaining regions. Experimental study based on a real world dataset verifies the effectiveness of the proposed region similarity measure and the efficiency of the algorithm.

1 Introduction

In the geo-spatial application, a *similar region query* happens when users want to find some similar regions to a query region on the map. The application scenarios include

- Similar region search. Due to the limitation of knowledge, people may only be familiar with some places where they visit frequently. For example, people go to the nearest entertainment region which include malls for shopping and the restaurants for dinner. Sometimes, people wish to know the alternative places as the options for both shopping and dinners. Base on their familiar entertainment region, similar region query retrieves the regions that have the similar functions to their familiar entertainment region.
- Disease surveillance. Similar region search query is also useful in identifying the potential high-risk areas that are prone to outbreak of diseases. Many infectious diseases thrive under the same geographical conditions. By querying regions that are similar in geographical characteristics, we can quickly highlight these high-risks areas.

The traditional IR model might be applied to answer similar region queries: A direct application is to partition the map into a set of disjoined regions, represent the region by a vector of PoI categories, and utilize the vector space model

[⋆] Part of this work was done when the first author worked as an intern in MSRA

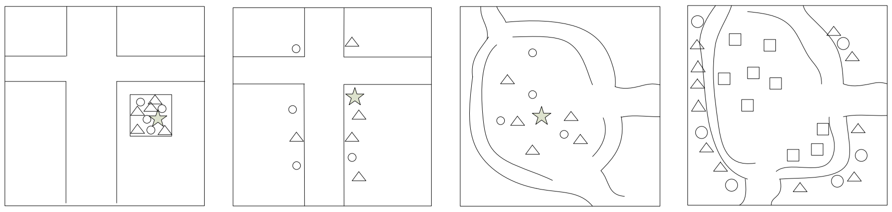

(a) Query region: Shopping mall (b) Cand. region 1: Shopping street (c) Cand. region 2: Shopping area (d) Cand. region 3: University town

Fig. 1. The first three plots show the distribution of five restaurants (triangles), four shops (circles) and one theater (star); The last plot gives a distribution of nine research institutes (rectangle) and many nearby restaurants (triangles) and shops (circles)

(VSM) [11] to evaluate the similarity of the regions. However, the traditional IR model is inadequate in supporting the "good" similar region queries due to two reasons. First, in traditional IR model, users are required to provide a set of keywords or terms to the search engine, and the search engine returns a list of texts which are relevant to the keywords. However, in similar region query, users only provide a query region instead of keywords. Second, similar region query searches the regions of similar region functionality, which is actually determined by the spatial objects (we call these spatial objects Point-of-Interests (PoIs) in the rest of paper) in the region and their spatial distribution in this region, i.e., *local distribution* or *distribution* in short. The traditional IR model does not take local distribution into account while computing the similarity.

For example, Figure 1 shows four local distributions, where the first three regions have the identical number and categories of PoIs, and the last region has different PoI categories from the first three regions. Given the query region shown in Figure 1(a), traditional IR model ranks Figure 1(b) and Figure 1(c) higher than Figure 1(d), because Figure 1(d) has different PoI categories. However, the traditional IR model could not distinguish the first three plots of Figure 1, which actually stand for three different region functionalities: Shopping malls are usually located in the communities as the entertainment centers; Shopping streets are located in the central business area for providing services to tourists; Shopping areas are located around the residential areas and the shops usually are groceries.

The above example highlights the importance of considering not only spatial objects categories but also their local distributions when answering similar region query. We present the problem for answering similar region query as follows.

Similar region query problem. Given a spatial map, a query region R_q, two coefficients to control the area of region μ_1 and μ_2, we aim to find the top-k most similar regions to R_q on the spatial map, such that 1) $\mu_1 \leq \frac{Area(R_i)}{Area(R_q)} \leq \mu_2$, R_i is a return region, and 2) any two return regions do not have large overlap [1].

[1] The degree of overlap is measured by the intersection ratio of two regions. In this paper, we set this ratio to be 0.8

In this paper, we focus on two main issues in tackling the similar region search problem. The first issue is to provide a proper definition for region similarity. While there have been extensive researches into defining the document similarity [1], to the best of our knowledge, there is no existing similarity measure for regions. In this paper, we propose a reference distance feature which is consistent with the human routines to compare region similarity. Accordingly, we extend the VSM model to the Spatial Vector Space Model (SVSM) by using the reference distance feature to capture the local distributions of spatial object categories.

Second, the search space in the region search problem is a continuous spatial map. Exhaustive search on the continuous spatial map is too expensive to provide quick response to users. To solve this problem, we provide a quadtree based approximate region search approach. The basic idea follows the filter-and-refine approach which is described as follows. We maintain a top-k region set and the similarity threshold to be a top-k region. We extract the representative categories from the query region and filter the quadtree cells that do not contain the representative categories. We further prune those cells that are not likely to be the top-k most similar regions. The remaining cells are remained as seeds to expand gradually. We insert the expanded regions into top-k regions if their similarity values are greater than the similarity threshold, and accordingly update the similarity threshold.

The remainder of this paper is organized as follows. Section 2 discusses the related work. Section 3 gives preliminaries. Section 4 presents the spatial vector space model. Section 5 proposes the quadtree-based region search approach. Section 6 reports our experiment results. Finally, Section 7 concludes this paper.

2 Related Work

Text retrieval is one of the most related problems. Conventional text retrieval focuses on retrieving similar documents based on text contents, and a few of similarity models, such as vector space model [11] and latent semantic analysis model [5], are proposed to compare the similarity. Recently, location-aware text retrieval, which combines both location proximity and text contents in text retrieval, receives much attention. To perform efficient retrieval, both document locations and document contents are required to be indexed in the hybrid index structures, such as a combination of inverted file and R*-tree [14], a combination of signature files and R-tree [7], DIR tree [3]. Our work substantially differs from location-aware text retrieval queries because we consider the relative locations of spatial objects and the returned results are regions which are obtained by the space partition index Quadtree.

Image retrieval [4], particularly content based image retrieval (CBIR) [10], is another related problem. CBIR considers the color, texture, object shape, object topology and the other contents, and represent an image by a single feature vector or a bag of feature vectors for retrieval. CBIR is different from the similar region query problem because CBIR focuses on either the content of whole image or the relationship from one object to another object, while we search the similar regions based on the local distribution of one category to another category. In addition, the image retrieval system searches the similar images from an image database, while our algorithm finds the similar regions on one city map which need to be properly partitioned during retrieval.

There are two existing approaches to select features from spatial data. The first approach is based on spatial-related patterns, such as collocation patterns [8] and interaction patterns [13]. Both patterns are infeasible to be employed in the similar region queries because they capture the global distribution among different PoI types, not the local distribution. The second approach is the spatial statistical functions test, like cross K function test [2]. In spite of theoretic soundness, this approach need long training time so that it is impractical to provide efficient response to the query.

3 Preliminaries

Suppose $\mathcal{P}$ is a spatial map, and $\mathcal{T}$ is a set of PoI categories $\mathcal{T} = \{C_1, C_2, \ldots, C_K\}$. Each PoI may be labelled with multiple PoI categories. For example, a building is labelled both as "Cinema" and "Restaurant" if it houses a cinema and has at least one restaurant inside. The PoI database $\mathcal{D}$ contains a set of PoIs. Each PoI in $\mathcal{D}$ is presented by a tuple $o = \langle p_o; \mathcal{T}_o \rangle$, where $p_o = (x_o, y_o)$ denotes the location of o, and $\mathcal{T}_o$ is a set of o's PoI categories.

A *region* R is a spatial rectangle bounded by $[R_{x_{min}}, R_{x_{max}}] \times [R_{y_{min}}, R_{y_{max}}]$ which locates in map $\mathcal{P}$. A PoI $o = \langle p_o; \mathcal{T}_o \rangle$ is said to *occur* in region R if $p_o \in R$. We use $\mathcal{D}^R = \{o | o \in \mathcal{D} \wedge p_o \in R\}$ to denote all PoIs which occur in region R, and $\mathcal{D}^R_{C_i} = \{o | o \in \mathcal{D} \wedge p_o \in R \wedge C_i \in \mathcal{T}_o\}$ to denote a set of objects with category C_i which occur in region R.

By modifying the concepts of TF-IDF measure in VSM, we define the *CF-IRF* as follows. The *Category Frequency (CF)* of the category C_i in region R_j, denoted as $CF_{i,j}$, is the fraction of the number of PoIs with category C_i occurring in region R_j to the total number of PoIs in region R_j, that is,

$$CF_{i,j} = \frac{\mathcal{D}^{R_j}_{C_i}}{\mathcal{D}^{R_j}} \tag{1}$$

The importance of a category C_i depends on the distribution of PoIs with category C_i on the entire map. Suppose we impose a $g_x \times g_y$ grid on the map. The *Inverse Region Frequency (IRF)* of category C_i, denoted as IRF_i, is the logarithm of the fraction of the total number of grids to the number of grids that contain PoIs with category C_i.

$$IRF_i = \log \frac{g_x \times g_y}{|\{\mathcal{D}^{R_j}_{C_i} | \mathcal{D}^{R_j}_{C_i} \neq \emptyset\}|} \tag{2}$$

With CF and IRF, the significance of a category C_i in region R_j, denoted as CF-IRF$_{i,j}$, is defined as follows:

$$\text{CF-IRF}_{i,j} = CF_{i,j} \times IRF_i \tag{3}$$

The information content of a region R_j is denoted as a vector

$$\overrightarrow{R_j} = (f_{1,j}, f_{2,j}, \ldots, f_{K,j}) \tag{4}$$

where $f_{i,j}$ denotes the CF-IRF value of category C_i in region R_j. We use $|\overrightarrow{R_j}|$ denotes the Euclidean norm of vector $\overrightarrow{R_j}$.

$$|\overrightarrow{R_j}| = \sqrt{f_{1,j}^2 + \ldots + f_{K,j}^2} \tag{5}$$

The *information content similarity* of two regions R_i and R_j is the cosine similarity of the corresponding feature vectors of R_i and R_j.

$$Sim(R_i, R_j) = \cos(\overrightarrow{R_i}, \overrightarrow{R_j}) = \frac{\overrightarrow{R_i} \cdot \overrightarrow{R_j}}{|\overrightarrow{R_i}| \times |\overrightarrow{R_j}|} \tag{6}$$

4 Spatial Vector Space Model

A similarity measure is desirable to evaluate the similarity of two regions. To be consistent with the human routines to compare region similarity, we propose the intuitive two level evaluation criteria as follows.

1. Do the regions have a significant overlap in their representative categories? This is the basic gist when users compare the similarity of regions. For example, the regions shown in Figure 1(c) and Figure 1(b) share three common categories, and the regions shown in Figure 1(c) and Figure 1(d) share two common categories. Therefore, the region pair {Figure 1(c), Figure 1(b)} is considered to be more similar than the region pair {Figure 1(c), Figure 1(d)}.
2. If two regions share some common representative categories, do the PoIs of the common representative categories exhibit similar spatial distribution? We observe that Figure 1(a), Figure 1(b) and Figure 1(c) all share the same representative categories, however they are not considered similar as the distributions of the PoIs for each category are drastically different in the three figures. In other words, two regions are more similar if they have not only the common representative categories but also the similar spatial distribution of PoIs. Given a query region of shopping mall, Figure 1(a) is more similar to this query region than Figure 1(b) and Figure 1(c).

CF-IRF feature satisfies the first level evaluation criterion but does not satisfy the second level criterion because it ignores the local distribution of the PoIs. This motivates us to propose a distribution-aware spatial feature and Spatial Vector Space Model (SVSM). In SVSM, a region R_j is represented by a **spatial feature vector** of n entries, $\overrightarrow{R_j} = (f_{1,j}, f_{2,j}, \ldots, f_{n,j})$ where $f_{i,j}$ is the i-th

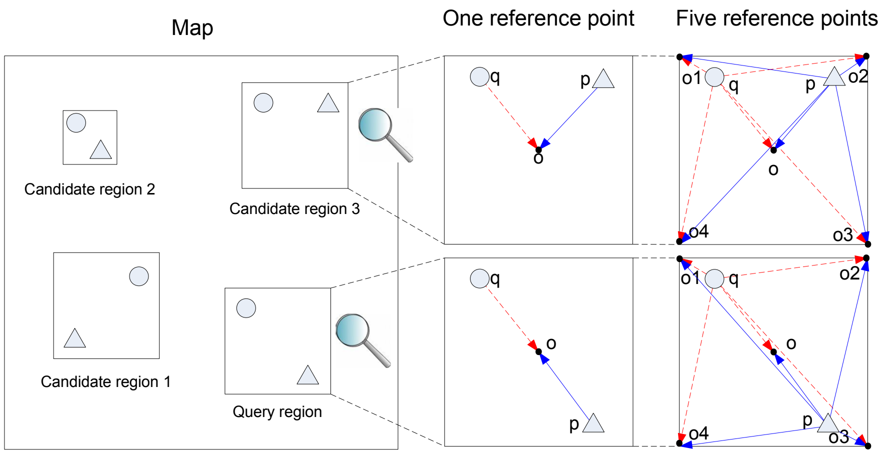


Fig. 2. An example of query region and its reference distance

spatial feature entry and n is total number of features or the dimension of the feature vector.

A desirable spatial feature will be insensitive to rotation variation and scale variation. In other words, if two regions are similar, rotating or magnifying one of the two regions will not affect their similarity. Figure 2 illustrates a query region and three candidate regions in a map. We consider candidate region 1 and 2 are similar to the query region because they are similar to query region after rotating by a scale variation (clockwise 270^o) or magnifying by a scale variation, respectively. In contrast, candidate region 3 is not so similar as candidate region 1 and 2 to the query region. Motivated by the requirements to minimize the effects of scaling and allow for rotation invariant, we introduce the concept of reference distance to capture the spatial distributions.

Average nearest neighbor distance can be employed to measure the local distribution in statistics domain [6], but it is expensive ($O(n^2)$ if no spatial index is used, where n is the number of PoIs). Therefore, we propose the concept of reference point. The *reference points* are user-specified points in a region to capture the local distributions of the PoIs in this region. The intuition behind reference points is based on the observation that most users tend to use some reference points for determining region similarity. For example, while comparing two regions which have one cinema each, users tend to roughly estimate the average distances from the other categories to the cinema, and compare the estimated distances of two regions. Here, the cinema is a reference point.

It raises an issue to select the proper number of reference points and their locations. We consider two extreme cases as follows. On one hand, one or two reference points are not enough to capture the distribution. For example, Figure 2 shows that one reference point cannot distinguish the distribution of query region and candidate region 3. On the other hand, a larger number of reference points will give a more accurate picture of the spatial distributions among the

PoIs, but at the expense of greater computational cost. In this paper, we seek the tradeoff between the two extreme cases. We propose that five reference points, including the center and four corners of the region, are proper to capture the local distribution. Figure 2 illustrates the five reference points, and the reference distances of two PoIs to the five reference points. The complexity to compute reference distance is $O(5 \cdot n)$, which is more efficient than nearest neighbor distance $O(n^2)$. Here, we do not claim that the selection of five point reference points is the best, but experiment results show that it is reasonable.

We now define the reference distance. Give a region R, a set of PoIs P, and five reference points O=$\{o_1, o_2, \ldots, o_5\}$. The distance of P to the i-th reference point $o_i \in O$ is

$$r(P, o_i) = \frac{1}{|P|} \sum_{p \in P} dist(p, o_i) \tag{7}$$

Assume region R has K different categories of PoIs. We use $r_{i,j}$ to denote the distance of PoIs with category C_i to the reference point o_j.The distance of K categories to the reference set O is a vector of five entries.

$$I = \{\overrightarrow{I_1}, \ldots, \overrightarrow{I_5}\} \tag{8}$$

where each entry is the distance of K categories to the reference point o_j, $\overrightarrow{I_i} = (r_{1,i}, r_{2,i}, \ldots, r_{K,i})$.

The similarity of two feature vector sets $I_{R_i} = \{\overrightarrow{I_{1,i}}, \ldots, \overrightarrow{I_{5,i}}\}$ and $I_{R_j} = \{\overrightarrow{I_{1,j}}, \ldots, \overrightarrow{I_{5,j}}\}$, is

$$Sim_r(I_{R_i}, I_{R_j}) = \frac{1}{5} \sum_{k=1}^{5} Sim(I_{k,i}, I_{k,j}) \tag{9}$$

We incorporate the rotation variation into similarity as follows. Given region R_j, we obtain four rotated regions R_{j1}, R_{j2}, R_{j3} and R_{j4} by rotating R_j 90 degree each time. The similarity of R_i and R_j is the similarity of R_i and the most similar rotated region of R_j, that is,

$$Sim_r(R_i, R_j) = arcmax\{Sim_r(I_{R_i}, I_{R_{jk}}), k = 1, 2, 3, 4\} \tag{10}$$

Lemma 1. *The reference distance feature is insensitive to rotation and scale variations.*

Proof: Based on Equation 10, we can derive that the reference distance feature is insensitive to rotation variation. Now we prove that the reference distance feature is insensitive to scale variation as follows. Assume R_j is obtained by scaling R_i by a factor σ. We have $\overrightarrow{I_{R_j}} = \sigma\overrightarrow{I_{R_i}}$ and $|\overrightarrow{I_{R_j}}| = \sigma|\overrightarrow{I_{R_i}}|$. Therefore, $Sim_r(R_i, R_j) = \cos(\overrightarrow{I_{R_i}}, \sigma\overrightarrow{I_{R_j}}) = \frac{\overrightarrow{I_{R_i}} \cdot \sigma\overrightarrow{I_{R_i}}}{|\overrightarrow{I_{R_i}}| \times \sigma|\overrightarrow{I_{R_i}}|} = 1$. $\square$

5 Proposed Approach

Given a query region R_q and two coefficients to control the area of region returned, μ_1 and μ_2, the naive approach to answer similar region queries is to

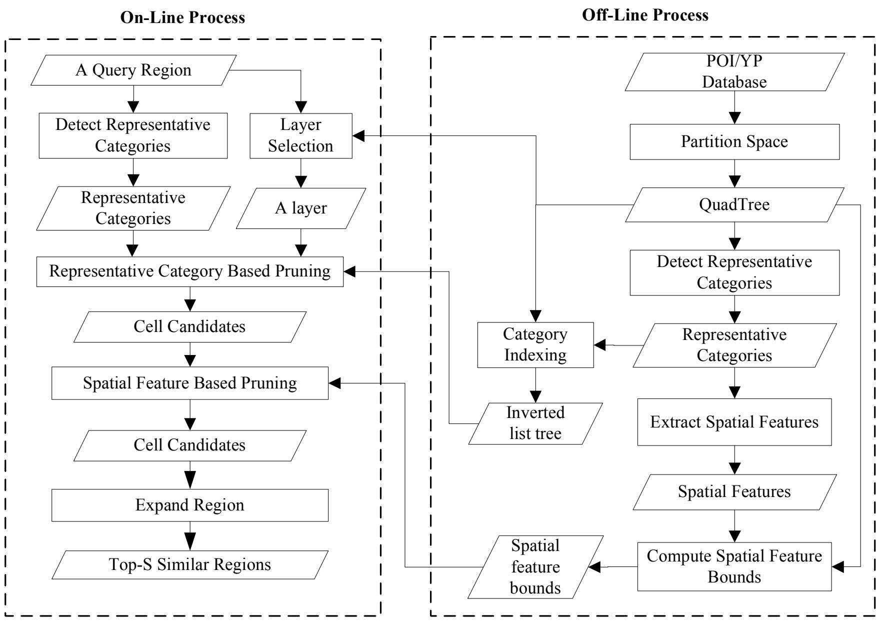


Fig. 3. Overview of system architecture

utilize a sliding window whose area is between $min\_area = \mu_1 \times area(R_q)$ and $max\_area = \mu_2 \times area(R_q)$. The sliding window is moved across the entire map and at each move, we compute the similarity between R_q and the sliding window. If we maintain a list of top-k regions having the k largest similarity values, then the time complexity of this naive approach in the worst case, given a $map_x \times map_y$ spatial map, is $O(k \times min\_area \times map_x \times map_y)$. This is because the total number of candidate regions is $min\_area \times map_x \times map_y$. For each candidate region we compute whether it overlaps with the existing top-k similar regions in $O(k)$ time complexity. Thus the overall time complexity of this algorithm is $O(k^2 \times min\_area \times map_x \times map_y)$, which is too expensive to provide a quick response to users.

To overcome the high complexity of the naive method, we propose a quadtree-based approximate approach. Figure 3 shows the system architecture overview of our approach. The architecture comprises an offline process and an online process. The offline process partitions the map into a hierarchical structure and builds a quadtree structure for quick retrieval of PoIs. The online process uses these index structures to perform region search queries efficiently. Given a query region, the system analyzes the shape and size of this region and determines the appropriate quadtree layer to initiate the similar region search process. At the same time, the system will compute the CF-IRF values to derive the representative categories of the query region. Once we know the starting level of the quadtree and the representative categories of the query region, we begin a filter-and-refine procedure to quickly reduce the search space that is unlikely to be in the top-k most similar regions.

5.1 Quadtree Structure

Given a PoI database and the map of this database, we partition the map and build a hierarchical quadtree structure [12] to facilitate the construction of multi-scale regions. In the quadtree, the root node indicates the whole map and each non-leaf node corresponds to one of the four partitioned cells from its parent's cell. At the lowest level, each leaf node corresponds to the partitioned cell with the smallest granularity. The depth of the quadtree depends on the smallest granularity requirement in applications. In our system, the leaf node is 100 meters by 100 meters, so the quadtree height is 10 for a city of 30 kilometers by 30 kilometers.

The quadtree structure enables an efficient handling of multi-granularity similar region queries. This is because we can adaptively select the different level of granularity by accessing the quadtree nodes at the appropriate level. For example, if the query region is the size of 200 meters by 200 meters and the parameter to control the minimal return region area μ_1=0.25, we perform the search on the leaf node because the leaf node area is no less than μ_1 times of the query region area.

The quadtree allows the effective region pruning by storing the key statistical information at each node in the quadtree. Each node maintains the lower bound and upper bound of feature entries defined as follows.

Definition 1. *The lower bound feature vector of a node B, denoted as $\overrightarrow{B}_{lb}$, is $(f_{1,lb}, f_{2,lb}, \dots, f_{n,lb})$, where $f_{i,lb}$ is the minimum i-th feature entry value of all descendant nodes of B.*

Definition 2. *The upper bound feature vector of a node B, denoted as $\overrightarrow{B}_{ub}$, is $(f_{1,ub}, f_{2,ub}, \dots, f_{n,ub})$, where $f_{i,ub}$ is the maximum i-th feature value of all descendant nodes of B.*

Each quadtree node maintains the minimum/maximum CF-IRF vector and the minimum/maximum reference distance vector. These bounds are useful for pruning the candidate regions as stated in Lemma 2.

Lemma 2. *Let $\overrightarrow{R_q} = (f_{1,q}, f_{2,q}, \dots, f_{n,q})$ to be the the feature vector of query region, δ to be the cosine similarity threshold of top-k regions. A node B can be pruned if for any feature entry $f_{i,q}$, we have $f_{i,ub} \cdot f_{i,q} \leq \frac{\delta}{n} \cdot |\overrightarrow{B}_{lb}| \cdot |\overrightarrow{R_q}|$.*

Proof: Let $f_{i,j}$ to be the i-th feature entry of region R_j where $R_j \in B$. Then $f_{i,lb} \leq f_{i,j} \leq f_{i,ub}$ and $|\overrightarrow{B}_{lb}| \leq |\overrightarrow{R_j}| \leq |\overrightarrow{B}_{ub}|$. Assume that $f_{i,ub} \cdot f_{i,q} \leq \frac{\delta}{n} \cdot |\overrightarrow{B}_{lb}| \cdot |\overrightarrow{R_q}|$. For the i-th feature entry $f_{i,j}$, we have $f_{i,j} \cdot f_{i,q} \leq f_{i,ub} \cdot f_{i,q} \leq \frac{\delta}{n} \cdot |\overrightarrow{B}_{lb}| \cdot |\overrightarrow{R_q}| \leq \frac{\delta}{n} \cdot |\overrightarrow{R_j}| \cdot |\overrightarrow{R_q}|$.

By summing up the inequalities, $\overrightarrow{R_j} \cdot \overrightarrow{R_q} = \sum_{p=1}^{n} f_{p,j} \cdot f_{p,q} \leq \delta \cdot |\overrightarrow{R_j}| \cdot |\overrightarrow{R_q}|$. So, we have $\cos(\overrightarrow{R_j}, \overrightarrow{R_q}) \leq \delta$, which means that any region R_j under B will not have a larger similarity than the top-k region similarity threshold. □

With Lemma 2, we can prune all nodes B that have no chance of satisfying the similarity threshold δ. For example, suppose the quadtree node B has four child nodes B_1, B_2, B_3, B_4. Each feature vector of child node has five entries.

$$\overrightarrow{B_1} = (0.1, 0.3, 0.1, 0.8, 0.0),\ \overrightarrow{B_2} = (0.1, 0.7, 0.2, 0.7, 0.0)$$
$$\overrightarrow{B_3} = (0.0, 0.3, 0.1, 0.8, 0.2),\ \overrightarrow{B_4} = (0.2, 0.4, 0.2, 0.6, 0.1)$$

So we have $\overrightarrow{B}_{lb}$ = (0.0, 0.3, 0.1, 0.6, 0.0) and $\overrightarrow{B}_{ub}$ = (0.2, 0.7, 0.2, 0.8, 0.2).

Let the feature vector of query region is $\overrightarrow{R_q}$ = (0.9, 0.1, 0.9, 0.1, 0.8) and δ = 0.95. We have $\frac{\delta}{n} \cdot |\overrightarrow{B}_{lb}| \cdot |\overrightarrow{R_q}|$=0.2468. The node B can be pruned because each feature entry product of $\overrightarrow{R_q}$ and $\overrightarrow{B}_{ub}$ is less than 0.2468.

In addition, we also construct an inverted tree index on the representative categories to facilitate similar region search. The root node of the inverted tree has K entries, where each entry corresponds to a category. Each category, say C_i, of a non-leaf node is associated with a child node that has four entries. The entry value is 1 if the corresponding partitioned region has the C_i as a representative category; otherwise the entry value will be 0. This inverted list tree is recursively built until it reaches a leaf node of the quadtree structure or all four entries have value 0. Based on this inverted tree index, we can quickly identify the cells that have similar categories to the query region.

5.2 Region Search Algorithm

In this section, we present the search strategy based on the quadtree structure. The basic idea is to compute the proper search level in the quadtree in which the buckets of search level will be greater than the minimal area of returned regions, and on the search level we select a few bucket as seeds to gradually expand to larger regions of proper size and large similarity value to the query region.

Algorithm 1 gives a sketch of the region search process. Line 1 computes the search level based on the granularity of query region. Line 2 extracts the representative categories from the search region R_q. The function *ExtractCategory* computes the CF-IRF values for each category on R_q and only maintains the top-m categories with the largest CF-IRF values. Line 3 adjusts the feature vector of R_q. The entries which correspond to the top-m representative categories remain and the other entries are set to be zero. Line 4 initializes the return region set to be an empty set and the similarity threshold δ to be 0. Line 5 calls procedure *SearchQTree* to search the similar regions.

Procedure *SearchQTree* recursively searches and prunes the candidate regions in quadtree. Line 8 is the validity checking for the top-k regions. A bucket is valid only if 1) it contains the CM representative categories, and 2) it cannot be pruned by Lemma 2. The inverted tree structure and the feature bounds of buckets facilitate the validity checking. If a bucket is valid and this bucket is higher level than l_{search} (Line 9), its child nodes need to be recursively detected further (Lines 10-11). Otherwise, Line 13 expands the valid buckets on l_{search} by calling the function *RegionExpansion*. Line 14 inserts the expanded region R to

Algorithm 1: RegionSearch(R_q, T, k, m)

```
input  : Query region $R_q$; Quadtree $T$; Number of return regions $k$; Number of
         representative categories $m$.
output: Top-$k$ similar regions
1  Compute the search level $l_{search}$ on $T$ based on $R_q$;
2  $CM$ = ExtractCategory($R_q$, $m$);
3  Adjust($\overrightarrow{R_q}$, $CM$);
4  $\mathcal{R} = \emptyset$; $\delta = 0$;
5  SearchQTree($\overrightarrow{R_q}$, $T.root$, $\delta$, $\mathcal{R}$);
6  return $\mathcal{R}$;

7  Procedure SearchQTree($R_q$, $B$, $\delta$, $\mathcal{R}$, $CM$)
8  if $B$ has $CM$ categories $\wedge$ $B$ cannot be pruned by Lemma 2 then
9      if $B.level < l_{search}$ then
10         foreach child node $B' \in B$ do
11             SearchQTree($R_q$, $B'$, $\delta$, $\mathcal{R}$);
12     else
13         $R$ = RegionExpansion($R_q$, $B'$);
14         $\mathcal{R} = \mathcal{R} \cup R$;
15         update $\delta$;

16 Function RegionExpansion($R_q$,$R$)
17 repeat
18     foreach $dir \in \{LEFT, RIGHT, DOWN, UP\}$ do
19         $R'' = expand(R, dir)$;
20         $dir = arcmax($ Sim($R_q$,$R''$));
21     $R' = expand(R, dir)$;
22 until Sim($R_q$, $R$) $\leq$ Sim($R_q$, $R'$);
23 return $R'$
```

the top-k region set $\mathcal{R}$, if R has no big overlap with the existing top-k regions or R has overlap with one existing top-k regions but R has a larger similarity value. Line 15 updates the similarity threshold δ based on the k-th largest similarity value in $\mathcal{R}$ currently.

The *RegionExpansion* function (Lines 16-23) treats a region as a seed, performs the tentative expansion in four candidate directions, and selects the optimal expanded region which gives the largest similarity value. The step width of each expansion is the cell side of the quadtree leaf node in order to minimize the scope of expansion, which eventually approach the local most similar region. The expansion stops if there is no increase in the similarity value (Line 22).

Finally, Line 6 returns the top-k regions. If the number of regions in $\mathcal{R}$ is less than k, we decrease the value of m by 1 in Line 2, and search the cells which share exact $m-1$ common representative categories. We repeatedly decrease the m value by 1 till the number of return regions in $\mathcal{R}$ reaches k.

6 Experiment Studies

In this section, we present the results of our experiments to examine the performance of similar region search. We first describe the experiment settings and the evaluation approach. Then, we report the performance on the region queries.

6.1 Settings

In our experiments, we use the Beijing urban map, which ranges from latitude 39.77 to 40.036, and longitude 116.255 to 116.555. The spatial dataset consists of the real world yellow page data of Beijing city in China. This dataset has two parts. The first part contains the persistent stationery spatial objects, such as the large shopping malls, factories, gas stations, land-marks, etc. The second part is the set of short-term and spatial objects which are updated from time to time, such as small restaurants and individual groceries. The total number of PoIs are 687,773, and they are classified into 48 major categories by their properties and functions.

We construct a quadtree for the Beijing urban map. The quadtree height is 10, and the cell side of quadtree leaf node is about 100 meters and the number of leaf nodes is 512×512. For each node of quadtree, we compute the lower bound and upper bound for the two features, namely category frequency and reference distance. We set μ_1=0.25 and μ_2=4, which means the return region areas range from one quarter to four times of query region areas.

As we are not aware of any existing work that support top-k similar region queries, we only evaluate two variants of the RegionSearch algorithm as follows. 1) VSM: It is a baseline algorithm based on the CF-IRF vector space model, and 2) SVSM: It is a spatial vector space model based algorithm that measures region similarity by the reference distance feature vector.

Given a query region, VSM and SVSM return the top-5 most similar regions respectively. Five users who are familiar with Beijing city score the return regions from 0 to 3 according to the relevance of the query region and the return regions. The final score of a return region is the average scores of five users. Table 1 gives the meanings of each score level.

Table 1. Users' scores for the return region

Scores	Explanations
0	Totally irrelevant
1	A bit relevant, with at least one common functionality with the query region
2	Partially relevant, the functionality of return region cover that of query region
3	Identically relevant, the return and query regions have the same functionality

We employ DCG (discounted cumulative gain) to compare the ranking performance of VSM and SVSM. The criteria DCG is used to compute the relative-to-the-ideal performance of information retrieval techniques. For example, given G =(2, 0, 2, 3, 1), we have CG =(2, 2, 4, 7, 8) and DCG =(2, 2, 3.59, 5.09, 5.52). The higher the scores computed by DCG, the more similar the return region. Please refer to [9] for the definitions of the cumulative gain (CG) and the discounted cumulative gain (DCG).

All the algorithms are implemented in C++ and the experiments are carried out on a server with dual Xeon 3GHZ processors and 4GB memory, running Windows server 2003.

6.2 Effectiveness study

We select three typical types of query regions as the test queries.

- The shopping mall. The shopping mall is one of the commercial community whose spatial points are clustered in small regions.
- The commercial street. The commercial street is another commercial community whose spatial points distributed along the streets.
- The university. The spatial points has a star-like distribution where the institutes are located at the center and other facilities such as hotels and restaurants are located around the university.

Each type of query region is given three query regions, which are listed in Table 2. We evaluate the average DCG values for each type of query region.

Table 2. The type and size (meter × meter) of nine query regions

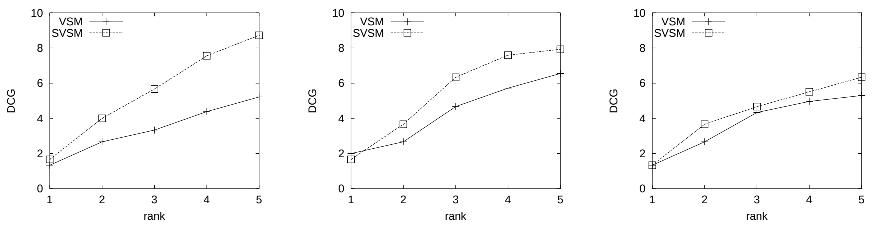
ID	Type	Query region	ID	Type	Query region	ID	Type	Query region
q_1	mall	150×150	q_4	street	150×600	q_7	university	1400×800
q_2	mall	100×300	q_5	street	200×500	q_8	university	1400×1100
q_3	mall	50×70	q_6	street	300×100	q_9	university	1200×1200

In order to find a proper number of representative categories, we run the two algorithms on different queries by varying the number of representative categories to be 3,5,10. We found that the average DCG curve of m=5 is better than the curves of m=3 and m=10. The result is consistent with our expectation because small m values are not enough to differentiate the region functionality, and large m values are likely to include some noise categories, both of which could affect the precision of return regions. We set m=5 in the rest experiments.

Figure 4 shows the average DCG curves for the three types of query regions. We observe that SVSM outperforms VSM for all of three query types, especially on shopping mall queries and street queries. This is expected because SVSM captures both the PoI categories and the local distribution in a region, which is consistent with human routines to evaluate the region similarity. In addition, SVSM did not have remarkable performance on the university queries. This is possibly because university query regions are larger than shopping mall queries and street queries, which results in the larger reference distances to decrease the contrast of spatial feature vector.

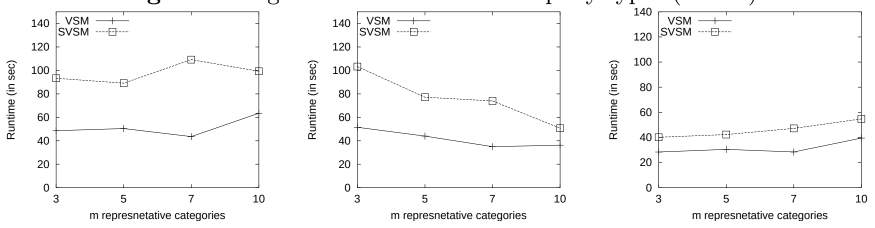
6.3 Efficiency Study

In this set of experiments, we study the efficiency of VSM and SVSM. Figure 5 gives the runtime for the three query types as m varies from 3 to 10. We see that



(a) Shopping mall query (b) Street query (c) University query

Fig. 4. Average DCG curves for three query types ($m = 5$)



(a) Shopping mall query (b) Street query (c) University query

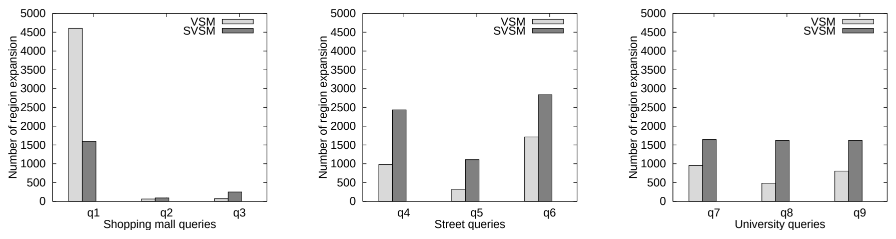
Fig. 5. Effect of m on runtime

both VSM and SVSM are scalable to m, but VSM is the faster than SVSM. This is because SVSM requires extra time to process the additional spatial features. We also observe that the runtime for SVSM decreases as m increases. This is because a larger number of representative categories lead to the pruning of more candidate regions. In addition, the runtime for the university queries is smaller than the other two query types since the runtime is determined by the area constraint of return regions. For large query regions, the search starts at the higher levels of the quadtree which have small number of candidate regions.

Next, we check the performance of pruning strategy. Since only the candidate regions which pass the validity test are expanded, we evaluate the pruning strategy by counting the number of region expansion operations. Figure 6 shows the number of region expansion performed for the three query types. In this experiment, the shopping mall queries and street queries have around 260,000 candidate regions, and the university queries have around 16,000 candidate regions. We observe that q_1 have more regions to be expanded than q_2 and q_3 (see Figure 6(a)). A closer look reveals that q_1 only contains three categories, hence many regions are considered as candidates. Figure 6(b) and Figure 6(c) show that less than 3,000 and 1,800 regions are expanded respectively, demonstrating the power of the pruning strategies.

7 Conclusion

In this paper, we introduce a similar region query problem in which spatial distribution is considered to measure region similarity. We propose the reference distance feature and spatial vector space model (SVSM) which extends the concept of vector space model to include reference distance features. We design a quadtree-based approximate search algorithm to filter and refine the search



(a) Shopping mall query (b) Street query (c) University query

Fig. 6. Number of region expansion

space by the lower and upper bounds of feature vectors. Experiments on the real world Beijing city map show that our approach is effective in retrieving similar regions, and the feature bounds are useful for pruning the search space. To the best of our knowledge, this is the first work on similar region search. We plan to investigate other types of spatial features for region similarity definition and hope to incorporate our techniques into the Microsoft Bing search engine.

References

1. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.
2. N. A. Cassie. *Statistics for Spatial Data*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1993.
3. G. Cong, C. S. Jensen, and D. Wu. Efficient retrieval of the top-k most relevant spatial web objects. *PVLDB*, 2(1):337–348, 2009.
4. R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, 40(2):1–60, 2008.
5. S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407, 1990.
6. D. Ebdon. *Statistics in geography*. Wiley-Blackwell, 1985.
7. I. D. Felipe, V. Hristidis, and N. Rishe. Keyword search on spatial databases. In *ICDE*, pages 656–665, 2008.
8. Y. Huang, S. Shekhar, and H. Xiong. Discovering colocation patterns from spatial data sets: a general approach. *IEEE Transactions on Knowledge and Data Engineering*, 16(12):1472– 1485, December 2004.
9. K. Jarvelin and J. Kekalainen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems*, 20:2002, 2002.
10. M. S. Lew, N. Sebe, C. Djeraba, and R. Jain. Content-based multimedia information retrieval: State of the art and challenges. *ACM Trans. Multimedia Comput. Commun. Appl.*, 2(1):1–19, 2006.
11. G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, 1975.
12. H. Samet. *The design and analysis of spatial data structures*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1990.
13. C. Sheng, W. Hsu, M. Lee, and A. K. H. Tung. Discovering spatial interaction patterns. In *DASFAA*, pages 95–109, 2008.
14. Y. Zhou, X. Xie, C. Wang, Y. Gong, and W.-Y. Ma. Hybrid index structures for location-based web search. In *CIKM '05*, pages 155–162, New York, NY, USA, 2005. ACM.