Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Finding map regions with high density of query keywords*

Zhi YU[1], Can WANG‡[1], Jia-jun BU[1], Xia HU[2], Zhe WANG[1], Jia-he JIN[1]

([1]*Alibaba-Zhejiang University Joint Institute of Frontier Technologies, Zhejiang Provincial Key Laboratory of Service Robot, College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China*)

([2]*Hangzhou Science and Technology Information Research Institute, Hangzhou 310001, China*)

E-mail: {yuzhirenzhe, wcan, bjj}@zju.edu.cn; huxia@hznet.com.cn; {wangzhe89, jinjiahe}@zju.edu.cn

Received Jan. 27, 2016; Revision accepted June 30, 2016; Crosschecked Nov. 3, 2017

Abstract: We consider the problem of finding map regions that best match query keywords. This region search problem can be applied in many practical scenarios such as shopping recommendation, searching for tourist attractions, and collision region detection for wireless sensor networks. While conventional map search retrieves isolate locations in a map, users frequently attempt to find regions of interest instead, e.g., detecting regions having too many wireless sensors to avoid collision, or finding shopping areas featuring various merchandise or tourist attractions of different styles. Finding regions of interest in a map is a non-trivial problem and retrieving regions of arbitrary shapes poses particular challenges. In this paper, we present a novel region search algorithm, dense region search (DRS), and its extensions, to find regions of interest by estimating the density of locations containing the query keywords in the region. Experiments on both synthetic and real-world datasets demonstrate the effectiveness of our algorithm.

Key words: Map search; Region search; Region recommendation; Spatial keyword search; Geographic information system; Location-based service

https://doi.org/10.1631/FITEE.1600043 **CLC number:** TP391

1 Introduction

A large amount of geographic object data is available on the Web such as shops, restaurants, and parking lots. More and more online businesses are incorporating the geographic data in their services. For instance, dianping.com, the largest website for location-based food and entertainment services in China with more than 90 million monthly active users, has now on their site more than 8 million local businesses. However, current location-based services offer only search for isolate locations on a map (known as points-of-interest, POIs for short), while frequently users are searching for regions instead. For example, when searching for electronics, we probably prefer a region densely populated with electronic shops to a single electronic store. Meanwhile, customers would have various demands not easily met in a single spot, e.g., buying Levi's Jeans while looking for some seafood at the same time. In these cases, traditional location-based services usually fail because a single spot can hardly satisfy such diverse requirements. To address this issue, we resort to finding regions instead of isolated locations. However, region search is a more challenging problem than location search, in that:

1. The keyword matching criterion for a region is different from that for a location. This is an even more complicated issue when the query consists of multiple keywords.

2. It is a non-trivial issue to determine the appropriate size and shape for a region of good match. A large region may contain more desired locations

‡ Corresponding author

* Project supported by the Zhejiang Provincial Natural Science Foundation of China (No. LZ13F020001), the National Natural Science Foundation of China (Nos. 61173185 and 61173186), the National Key Technology R&D Program of China (No. 2012BAI34B01), and the Hangzhou S&T Development Plan (No. 20150834M22)

ORCID: Can WANG, http://orcid.org/0000-0002-5890-4307

©Zhejiang University and Springer-Verlag GmbH Germany 2017

but may incur a higher cost of traveling to the region. A small region is easy to cover but options are usually limited.

3. Efficiency of the search algorithm is a critical issue because region ranking will usually involve more computation.

Although there are several density-based approaches, such as those proposed by Ester *et al.* (1996), Agrawal *et al.* (1998), and Guo *et al.* (2003), which use clusters as dense regions, these methods are not suitable for online region search, because a cluster is a set of POIs, which is not a real connected region in the 2D space. Finding a suitable region covering these point sets, such as Convex Hull, incurs too much computational cost for online services, which are expected to generate responses in a short latency time. Moreover, it is difficult to control the region size in cluster-based methods, since the spanning of data points is seldom considered in existing clustering algorithms. Thus, the regions for cluster-based methods may be too large or too small for users to visit. Some existing studies attempt to address this region size issue using grid-based methods (Schikuta, 1996; Agrawal *et al.*, 1998; Cheng *et al.*, 1999; Hinneburg and Keim, 1999). These methods divide the data space into a finite set of grid cells according to the distribution of each keyword, so there are different sets of grid cells for different keywords. While grid-based methods might be a good solution for single-keyword queries, it will be difficult for existing grid-based approaches to find dense regions in multi-keyword search.

To solve the above-mentioned problems, we propose a density-based search algorithm to find regions of interest (ROIs), i.e., regions densely populated with locations labeled with query keywords. By partitioning a map into small grids, neighboring grids densely containing target locations can be merged iteratively to form a larger region. By this way, we can efficiently retrieve the dense regions matching the query keywords. The algorithm is further extended to find top-k dense regions and to accommodate multiple query keywords.

Our main contributions are summarized as follows:

1. We propose a novel search algorithm for map regions best matching the user query.

2. We introduce a density-based matching criterion for finding good regions in map search.

3. To make the search more efficient, we develop a greedy algorithm that approximates the optimal results but incurs a much lower computational cost.

4. The extensions of our algorithm can well accommodate the top-k region search and multi-keyword query.

2 Related work

We review mainly work on map search and dense subgraph.

2.1 Map search

Map search, or spatial keyword search, is a central issue in a location-based service (LBS) and a geographic information system (GIS), and has been studied by the research community for quite a long time (Jones *et al.*, 2002; Zhou *et al.*, 2005; Chen YY *et al.*, 2006; Chen LS *et al.*, 2013; Mai *et al.*, 2013; Zhang *et al.*, 2013b; Ortega *et al.*, 2014; Son, 2014; Saoussen *et al.*, 2014) and urban computing supplies a lot of applications for the citizen (Wei *et al.*, 2012; Yuan *et al.*, 2012; Zhang *et al.*, 2013a). The key problem in spatial keyword search is to find relevant locations (POIs) measured simultaneously by keyword similarity and geographic distance (Joshi *et al.*, 2008; Leung *et al.*, 2011). The existing approaches use mainly a geo-textual approach to efficiently retrieve locations relevant to the query keyword. They can generally be categorized into three classes according to the spatial index they use: (1) R-tree based indices (Zhou *et al.*, 2005; Cong *et al.*, 2009; Li *et al.*, 2011; Wu *et al.*, 2012); (2) grid-based indices (Vaid *et al.*, 2005; Khodaei *et al.*, 2010); (3) space filling curve based indices (Chen *et al.*, 2006; Christoforaki *et al.*, 2011).

While these studies focus only on retrieving POIs in map search, Fan *et al.* (2012) proposed a search method SEAL that explicitly finds ROIs in a map. SEAL uses an ROI as query input to retrieve the most similar region from a predefined set of rectangular regions by considering both spatial overlap and textual similarity, and a new approach for uncovering locally characterizing regions has been proposed by Thomee and Rae (2013). In contrast, the dense regions in our study are similar to the ROIs in Fan *et al.* (2012) and Thomee and Rae (2013). Our work differs from these in that: (1) we use textual keywords instead of map regions as query input and

recommend top-k dense regions to users; (2) instead of searching from a predefined candidate set of regions, our method can search the whole map.

2.2 Dense subgraph search

Finding a dense region is a fundamental computational problem closely related with the dense subgraph problem. One type of this kind of problem aims to find the dense k-point set in the 2D space (Aggarwal *et al.*, 1989) or dense k subgraphs of topological graphs (Feige and Seltser, 1997; Feige *et al.*, 2001; Komusiewicz and Sorge, 2012). The former includes a k-variance problem, i.e., finding a set of k points with a minimum variance, which is similar to finding a region with high density on just k points, but the required variable k is useless for users because they cannot give a proper k before querying. Also, to resolve the k-variance problem efficiently, the k-order Voronoi diagrams method is the best choice (Shamos and Hoey, 1975; Aggarwal *et al.*, 1989; Aurenhammer, 1991), but it cannot achieve good performance when there are a large number of points with a large k, because there are too many points on the same circle to partition easily (Lee, 1982).

There also exist methods that retrieve a dense cluster as a dense region (Ester *et al.*, 1996; Agrawal *et al.*, 1998; Guo *et al.*, 2003). Generally, these methods use two different strategies in density-based clustering: a grid-based approach (Schikuta, 1996; Agrawal *et al.*, 1998; Cheng *et al.*, 1999; Hinneburg and Keim, 1999) and a neighborhood-based approach (Ester *et al.*, 1996; Ankerst *et al.*, 1999). Although multiple dense clusters matching query keywords can be retrieved in these methods, finding an appropriate region that covers all the POIs in a cluster remains a challenge. Also, in these methods the map space is divided into grid cells according to the distribution of POIs matching a query keyword. Since different keywords may produce different grid partition, it will be difficult to merge different partitions to accommodate multi-keyword query.

3 Dense region search

While previous studies on map search focus mainly on finding isolated locations that best match the query keyword, the objective of our dense region search is to find on a map a connected region most closely related with query keywords. By 'most closely related' we mean a region densely populated with, or close to, locations labeled with query keywords. As we are going to see, this relation between a region and the query keyword is best described by a region keyword density with both the size of the region and the distribution of labeled locations being considered. Thus, in this section, we will first introduce the region keyword relevance that measures how relevant a region is to a given query keyword. We then start with a simple dense region search problem of finding the densest region for one single query keyword. Then we extend our algorithm to accommodate multiple query keywords and top-k dense regions.

3.1 Region keyword relevance

Consider a map with locations labeled with different keywords (e.g., sea food, Levis). We represent each location l as a point on the 2D plane with coordinates $\{x, y\}$ derived from its longitude and latitude. For each keyword q_i $(i = 1, 2, \ldots, k)$ in a set of k keywords Q, we define the corresponding keyword location set $S^{\{i\}} = \{s_1^{(i)}, s_2^{(i)}, \ldots, s_{n^{(i)}}^{(i)}\}$ as the set of locations labeled with keyword q_i, with each $s_j^{(i)}$ being a keyword location for q_i. Fig. 1 illustrates an example with two keyword location sets. We say a connected region E is highly relevant to a keyword q_i if E is densely populated with corresponding keyword location $s_j^{(i)}$ or is close to many keyword locations $s_j^{(i)}$. If we use $\mathrm{Rel}(E, q_i)$ to denote the region keyword relevance between a connected region E and keyword q_i, a simple way of defining this region keyword relevance is as follows:

$$\mathrm{Rel}(E, q_i) = |E \cap S^{(i)}|, \tag{1}$$

that is, counting the keyword locations in region E.

However, if we take a deeper look at this keyword relevance, we will find that both keyword locations in region E and keyword locations outside but close to region E will contribute to this relevance.

Thus, instead of using the simple keyword location counting scheme in Eq. (1), we will use a definition based on a Gaussian field to take in the contribution of keyword locations outside but close to region E. We first define the relevance between a location l and a keyword location $s^{(i)}$ based on their Gaussian distance:

$$\mathrm{Rel}(l, s^{(i)}) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\|l - s^{(i)}\|^2}{2\sigma^2}\right). \tag{2}$$

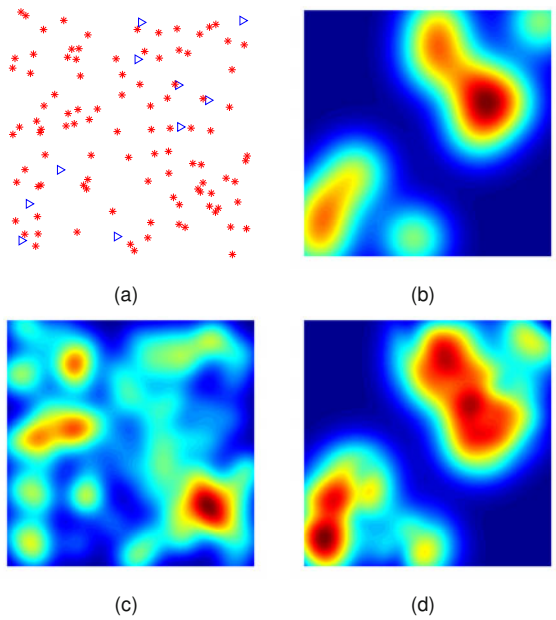

Fig. 1 Location keywords relevance: (a) two keyword locations (triangles and stars); (b) location keyword relevance for the triangle keyword; (c) location keyword relevance for the star keyword; (d) combined location keyword relevance for two keywords. References to color refer to the online version of this figure

The parameter σ in the above equation determines how much relevance a keyword location can propagate to its neighborhood. The simple keyword location counting scheme in Eq. (1) then corresponds to the special case $\sigma \to 0$, in which all the relevance is concentrated in a single location. We can then derive the location keyword relevance $\mathrm{Rel}(l, q_i)$ between a keyword q_i and a location l from Eq. (2):

$$\begin{aligned}\mathrm{Rel}(l, q_i) &= \sum_{s^{(i)} \in S^{(i)}} \mathrm{Rel}(l, s^{(i)}) \\ &= \frac{1}{\sigma\sqrt{2\pi}} \sum_{s^{(i)} \in S^{(i)}} \exp\left(-\frac{\|l - s^{(i)}\|^2}{2\sigma^2}\right). \end{aligned} \quad (3)$$

The region keyword relevance between a connected region E and keyword q_i is

$$\begin{aligned}\mathrm{Rel}(E, q_i) &= \int_E \mathrm{Rel}(l, S^{(i)})\, \mathrm{d}l \\ &= \frac{1}{\sigma\sqrt{2\pi}} \int_E \sum_{s^{(i)} \in S^{(i)}} \exp\left(-\frac{\|l - s^{(i)}\|^2}{2\sigma^2}\right) \mathrm{d}l. \end{aligned} \quad (4)$$

Note that the region keyword relevance $\mathrm{Rel}(E, q_i)$ depends on all the locations within or close to region E labeled with keyword q_i. This is important in the case of noisy data, which is a common case in a location-based service. If a shop changes its business and the keyword related with the location is not updated, a data error may occur. As we can see, the errors with a few locations in a region will be smoothed by other locations in this region. In other words, region-based search is more robust than location-based search in the case of noisy data. We summarize the symbols used in Table 1.

Table 1 Symbols and functions

Symbol	Definition
l	A point in the 2D plane
Q	A set of k keywords
q_i	The ith keyword
$s^{(i)}$	A location labeled with q_i
$S^{(i)}$	Keyword location set for q_i
$n^{(i)}$	Number of locations in $S^{(i)}$
E	A connected region in the 2D plane
$N_r(l)$	An r-radius neighborhood of l
$\boldsymbol{g}_{x,y}$	The x-row y-column grid region
$l_{\boldsymbol{g}_{x,y}}$	The center point of $\boldsymbol{g}_{x,y}$
$\mathbf{GR}^{(i)}_{x,y}$	Keyword relevance between $\boldsymbol{g}_{x,y}$ and q_i
GS	A set of grid regions
$\|A\|$	Number of points in set A
$\mathrm{Area}(E)$	Area size of E
$\mathrm{Rel}(E, q_i)$	Keyword relevance between E and q_i
$\mathrm{Den}(E, q_i)$	Keyword density between E and q_i
$\mathrm{Den}(\mathrm{GS}, q_i)$	Keyword density between GS and q_i

3.2 Single keyword dense region search

3.2.1 Approximate region density

Using the region keyword relevance in Eq. (4), we can formulate the problem of finding the densest region for one single query keyword as maximizing the following region keyword density between a connected region E and keyword q_i:

$$\mathrm{Den}(E, q_i) = \frac{\mathrm{Rel}(E, q_i)}{\mathrm{Area}(E) + C}, \quad (5)$$

where $\mathrm{Area}(E)$ is the area size of region E, and C is the regularization parameter that prevents region E from collapsing into a single location. As we are going to see, parameter C virtually controls the scope of the dense region retrieved in our search.

Given an area E, we can use Eqs. (4) and (5) to directly calculate the relevance and density for a region E on the corresponding set for query keyword q_i. However, the complexity will be prohibitive

for a large dataset. We next propose a single keyword dense region search algorithm that drastically reduces the computation complexity. The key is to use local approximation in a small neighborhood. That is, given an arbitrary location l, there is a small neighborhood region $N_\varepsilon(l)$ in which the location keyword relevance can be considered approximately constant:

$$\forall\,\alpha>0,\ \exists\,\varepsilon>0,\ \forall\,l'\in N_\varepsilon(l),\\ \|\mathrm{Rel}(l',s^{(i)})-\mathrm{Rel}(l,s^{(i)})\|<\alpha. \tag{6}$$

From the above equation we can derive the approximate region location relevance between $N_\varepsilon(l)$ and $s^{(i)}$ as

$$\begin{aligned}\mathrm{Rel}(N_\varepsilon(l),s^{(i)})&=\int_{N_\varepsilon(l)}\mathrm{Rel}(l',s^{(i)})\,\mathrm{d}l'\\ &\approx\int_{N_\varepsilon(l)}\mathrm{Rel}(l,s^{(i)})\,\mathrm{d}l'\\ &=\mathrm{Rel}(l,s^{(i)})\cdot\mathrm{Area}(N_\varepsilon(l)).\end{aligned} \tag{7}$$

Thus, the approximate region keyword relevance between $N_\varepsilon(l)$ and q_i is

$$\mathrm{Rel}(N_\varepsilon(l),q_i)\approx\mathrm{Rel}(l,q_i)\cdot\mathrm{Area}(N_\varepsilon(l)). \tag{8}$$

Given a region E, we can derive a partition for this region with pn small sub-regions $\{\Delta_1,\Delta_2,\ldots,\Delta_{\mathrm{pn}}\}$ of equal size δ:

$$E=\bigcup_j\Delta_j, \tag{9}$$

$$\forall\,a\neq b\Leftrightarrow\Delta_a\cap\Delta_b=\varnothing. \tag{10}$$

Also, the area size for E is $\mathrm{Area}(E)=\mathrm{pn}\cdot\delta$.

When δ is small, we can take Δ_j as an ε-neighborhood of a location l_i and we have

$$\mathrm{Rel}(\Delta_j,q_i)\approx\mathrm{Rel}(l_j,S^{(i)})\cdot\delta. \tag{11}$$

The region keyword relevance between E and q_i can be approximated as

$$\mathrm{Rel}(E,q_i)=\sum_j\mathrm{Rel}(\Delta_j,q_i)\approx\delta\cdot\sum_j\mathrm{Rel}(l_j,q_i), \tag{12}$$

and the corresponding region keyword density between E and q_i is

$$\mathrm{Den}(E,q_i)=\frac{\mathrm{Rel}(E,q_i)}{\mathrm{Area}(E)+C}\approx\frac{\sum_j\mathrm{Rel}(l_j,q_i)}{\mathrm{pn}+C'}, \tag{13}$$

where $C'=C/\delta$.

Eq. (13) indicates that the region keyword density of a region E can be well approximated by the fraction between the summation of region keyword relevance of pn sub-regions and the number of sub-regions plus a constant. This immediately suggests an efficient method for calculating the region keyword density for a region in a map by partitioning the map into small square grids of equal size.

3.2.2 Single keyword dense region search

Based on Eq. (13), we present a two-step dense region search method for a single-keyword query. In the preprocessing step, we partition the map into tiny square grids and for each keyword q_i calculate the keyword region relevance for each grid. In the search step, we propose a greedy search algorithm that retrieves a dense region from the map.

1. Preprocessing step. We map all the locations in the dataset into $[0,1]\times[0,1]$ by scaling and translation of their longitude and latitude. This area is further divided into $a\times a$ grids where a is a parameter determined by users. For each keyword q_i, we then calculate the location keyword relevance for the center of each grid. To further reduce the computational cost of the preprocessing step, we use only the locations within the scope of 3σ to each grid center in calculating its location keyword relevance. It is an inherent property of the Gaussian distribution that the relevance outside the scope of 3σ is negligible. The preprocessing step is summarized in Algorithm 1. The location keyword relevance between the query keyword q_i and the grids is stored in the relevance matrix $\mathbf{GR}^{(i)}$.

Algorithm 1 DRS preprocessing

Input: corresponding set $S^{(i)}$, parameters a and σ.
Output: relevance matrix $\mathbf{GR}^{(i)}$.

```
Obtain a zero a × a matrix GR^(i);
for each s^(i) ∈ S^(i) do
    for each g_{x,y} ∈ N_{3σ}(s^(i)) do
        TempRel = Rel(l_{g_{x,y}}, s^(i)) by Eq. (2);
        GR^(i)_{x,y} = GR^(i)_{x,y} + TempRel;
    end for
end for
return GR^(i);
```

2. Search step. To retrieve the dense region for query q_i, the search step engages a greedy algorithm

that starts with the grid of the highest location keyword relevance to q_i. The densest neighboring grid is merged into the current dense region at each iteration, and the neighboring grids are updated accordingly. By 'neighboring grids' we mean the grids to the top, bottom, left, and right of the current dense region. An example of neighboring grids is given in Fig. 2, where the current dense region is colored green and its neighboring grids are colored blue. In this example, the grid region marked with a yellow dot is the one with the highest location keyword relevance in the neighboring grids. It is then removed from the current neighboring grids and added to the current dense region. The two grids marked with black dots are then added to the neighboring grids. The region keyword density is calculated at each iteration as follows:

$$\mathrm{Den}(\mathrm{GS}, q_i) = \frac{\sum_{\boldsymbol{g}_{x,y} \in \mathrm{GS}} \mathbf{GR}^{(i)}_{x,y}}{|\mathrm{GS}| + C'}. \tag{14}$$

GS denotes the set of grids in the current dense region $\mathrm{GS} = \{\boldsymbol{g}_{x,y}\}$. The search step will stop if the region keyword density in Eq. (14) no longer increases and the current dense region is retrieved as the search result, which is

$$E = \bigcup_{\boldsymbol{g}_{x,y} \in \mathrm{GS}} \boldsymbol{g}_{x,y}. \tag{15}$$

We summarize the search step in Algorithm 2.

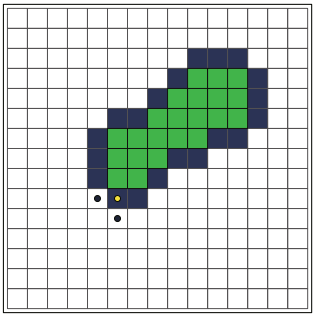

Fig. 2 A dense region and its neighboring grids. References to color refer to the online version of this figure

3.3 Extensions of DRS

3.3.1 Top-k dense region search

DRS retrieves only one dense region in a map. However, users probably prefer finding multiple dense regions to choose from. A side effect of using regularization parameter C in Eq. (5) is that the dense region containing the grid of the highest location keyword relevance may not be the region with the highest region keyword density in the map. That is, DRS is not guaranteed to find the densest keyword region. To overcome this limitation, we present an extension of the DRS that finds the top-k dense keyword regions.

The extension is to run DRS multiple times, with all the locations in the retrieved region being removed before the next run. To tackle the side effect introduced by the regularization parameter C, DRS is run $2k$ times and consequently $2k$ candidate regions are retrieved. These candidate regions are then re-ranked and the top-k regions are selected as the top-k dense regions. We summarize this extension in Algorithm 3.

Algorithm 2 DRS search

Input: relevance matrix $\mathbf{GR}^{(i)}$ and parameter C'.
Output: a dense region E and a set of grid regions GS.

Obtain an empty set GS and N;
Obtain an empty region E;
Label all grids UNFOUND;
Find the grid region $\boldsymbol{g}_{x,y}$ with the highest $\mathbf{GR}^{(i)}$;
$\mathrm{GS} = \{\boldsymbol{g}_{x,y}\}$;
$E = \boldsymbol{g}_{x,y}$;
Insert the neighboring grids of $\boldsymbol{g}_{x,y}$ into N;
Label $\boldsymbol{g}_{x,y}$ and neighboring grids FOUND;
OldDen = Den(GS) by Eq. (14);
while N is not empty **do**
  Find $\boldsymbol{g}_{x,y}$ in N with the highest relevance;
  NewDen = $\mathrm{Den}(\mathrm{GS} \cup \{\boldsymbol{g}_{x,y}\})$;
  if NewDen < OldDen **then**
    break;
  end if
  OldDen = NewDen;
  $\mathrm{GS} = \mathrm{GS} \cup \{\boldsymbol{g}_{x,y}\}$;
  $E = E \cup \boldsymbol{g}_{x,y}$;
  Insert $\boldsymbol{g}_{x,y}$'s UNFOUND neighboring grids into N;
end while
return E and GS;

3.3.2 Multi-keyword query

Sometimes people want to search dense regions using multiple keywords, e.g., using query keywords clothes, shoes, and basketballs to find shopping areas selling all the searched items. Multi-keyword query can be achieved by modifying only the preprocessing

Algorithm 3 DRS top-k region query

Input: relevance matrix $\mathbf{GR}^{(i)}$; parameters a, σ, C', and k.
Output: top-k regions $\{E_1, E_2, \ldots, E_k\}$.

```
for i = 1 to 2k do
    Calculate E_i and GS_a from Algorithm 2;
    for each s^(i) ∈ S^(i) ∩ E_i do
        for each g_{x,y} ∈ N_{3σ}(s^(i)) do
            TempRel = Rel(l_{g_{x,y}}, s^(i)) by Eq. (2);
            GR^(i)_{x,y} = GR^(i)_{x,y} − TempRel;
        end for
    end for
end for
Sort {E_1, E_2, ..., E_2k} by decreasing order of density, and remove bottom-k regions {E_{k+1}, E_{k+2}, ..., E_2k};
return {E_1, E_2, ..., E_k};
```

step of DRS. That is, to calculate the location keyword relevance for each grid, the relevance to all the query keywords should be considered. For this purpose, we use the harmonic mean function to calculate the multi-keyword relevance:

$$\mathrm{Rel}(E, \{S^1, S^2, \ldots, S^m\}) = \frac{m}{\sum_{i=1}^{m} \frac{1}{\mathrm{Rel}(E, S^{(i)})}}. \quad (16)$$

An example of the multi-keyword relevance for two keywords is shown in Fig. 1d. Using Eq. (16) in the preprocessing step, DRS and its top-k dense region extension can retrieve the dense region(s) related to multiple query keywords. We summarize the preprocessing step of multi-keyword query extension in Algorithm 4, while the search step is similar to those described by Algorithms 2 and 3.

We summarize the workflow of DRS and its extensions in Fig. 3, from which we can see that multi-keyword region search differs from single-keyword region search only in how the location keyword relevance is calculated. This makes the implementation of the extension easy.

Algorithm 4 DRS multi-keyword preprocessing

Input: m-keyword location set $\{S^{(1)}, S^{(2)}, \ldots, S^{(m)}\}$; parameters a and σ.
Output: relevance matrix $\mathbf{GR}^{\mathrm{all}}$.

```
for a = 1 to m do
    Calculate GR^(a) by Algorithm 1 using S^(a);
end for
for each g_{x,y} do
    GR^all_{x,y} = m / Σ_{i=1}^{m} 1/GR^(i)_{x,y};
end for
return GR^all;
```

4 Experiment results

In this section, we evaluate the performance of our algorithms on the synthetic dataset and a real world dataset. We begin with the description of the experimental setup.

4.1 Experimental setup

There are several parameters for our algorithm: variance σ for the Gaussian distribution, regularization parameter C, and the number of grid regions $a \times a$. We choose the parameter values as follows:

1. Each keyword q_i has a corresponding keyword location set $S^{(i)}$ with different distribution and location numbers $n^{(i)}$. In our experiment, we choose σ_i for each q_i as the mean of the distance between a location and its closest neighbor, i.e.,

$$\sigma_i = \frac{1}{n^{(i)}} \sum_{j=1}^{n^{(i)}} \|s_j^{(i)} - s_{j'}^{(i)}\|, \quad (17)$$

where $s_{j'}^{(i)}$ is the closest neighbor of $s_j^{(i)}$.

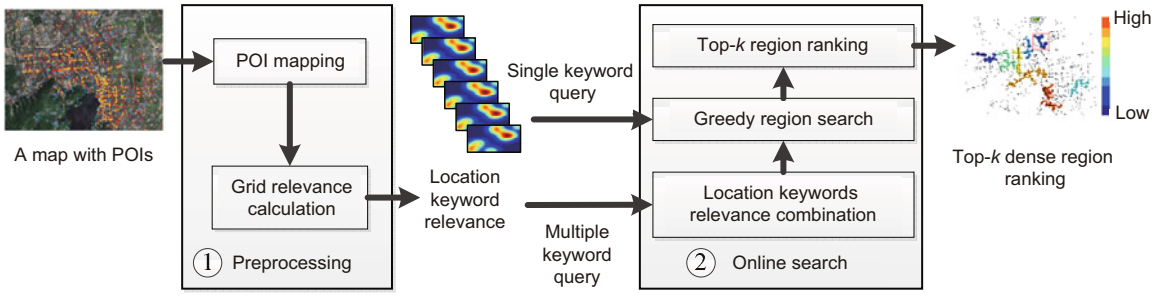

Fig. 3 The workflow of DRS and its extensions

2. The regularization parameter C used in Eq. (5) controls the size of the retrieved dense region. In our experiment, C is set to 0.01.

3. We divide the map used in our experiment into 1000×1000 grid regions, which is close to the screen resolution. The area for each grid region is therefore 1×10^{-6}.

4.2 Experiments on a synthetic dataset

In this subsection, we test the performance of our algorithm on randomly generated map data with a single dense region of arbitrary shape. We compare the accuracy of different algorithms in finding this dense region. As algorithm efficiency is a major concern in searching large map data, we also evaluate the time complexity in this subsection.

4.2.1 Datasets

We generate about 100 points in a small dense region of different shapes:

Rectangle: We generate 10×10 points in a 0.1×0.1 square region with even distribution (Fig. 4a).

Circle: We generate 11×11 points in a 0.1×0.1 square region with even distribution, and then remove all the points whose distances from the region center are more than 0.05 (Fig. 4b).

Street: We generate 2×50 points in a 0.02×0.5 rectangular region with even distribution (Fig. 4c).

Ro-street: We make a rotation and translation to the street data (Fig. 4d).

Cross: We generate 2×25 points in a 0.02×0.25 rectangular region and 25×2 points in a 0.25×0.02 rectangle with even distribution (Fig. 4e).

Ro-cross: We make a rotation and translation to the cross data (Fig. 4f).

To test the algorithms' ability in retrieving dense regions, we add random noises of 0, 100, 200, 300, 400, or 500 locations to $S^{(i)}$, respectively, and then run the algorithms. Fig. 4 shows the maps after adding 500 randomly generated locations. For each noise configuration, 10 test runs are conducted for each algorithm and the averages are calculated. The dense regions found by the test algorithms are shown in Fig. 4.

4.2.2 Performance evaluation

To quantitatively evaluate the experimental results, we use three metrics, Precision, Recall, and

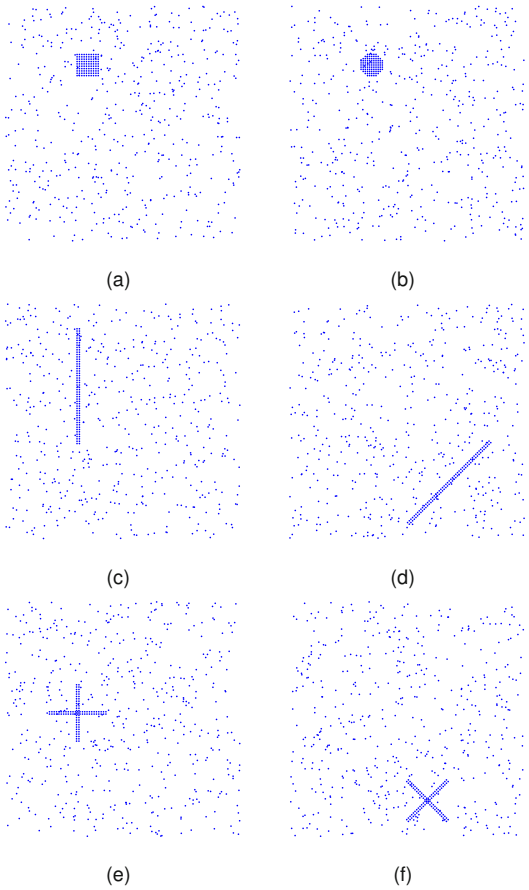

Fig. 4 Synthetic dataset: (a) rectangle; (b) circle; (c) street; (d) ro-street; (e) cross; (f) ro-cross

F_1-Score, to measure the region search performance. They are defined as

$$\text{Precision} = \frac{|E \cap S_{\text{in}}|}{|E \cap S_{\text{all}}|}, \tag{18}$$

$$\text{Recall} = \frac{|E \cap S_{\text{in}}|}{|S_{\text{in}}|}, \tag{19}$$

$$F_1 = \frac{2 \cdot \text{Precision}(E) \cdot \text{Recall}(E)}{\text{Precision}(E) + \text{Recall}(E)}, \tag{20}$$

where S_{in} is the set of locations in the dense region we have generated, S_{all} is the set of all locations we have generated, and $|A|$ is the number of elements in set A.

As we aim to present an efficient and effective algorithm, we also evaluate the average running time of each algorithm.

4.2.3 Comparison of different algorithms

To demonstrate the effectiveness of our DRS algorithm, we implement two other baseline algorithms, which are:

1. Minimum bounding rectangle (MBR): MBR finds the densest region from all the rectangular regions with edges parallel to the coordinate axis. It is provable that each edge of the rectangle shall be incident on a location. Therefore, the search space is the $O(n^4)$ rectangles with edges incident on locations. We use Eq. (1) to calculate the keyword region relevance for MBR, so the region keyword density for MBR is defined as

$$\mathrm{Den}(E,q)=\frac{|E\cap S|}{\mathrm{Area}(E)+C}. \tag{21}$$

2. Minimum bounding circle (MBC): MBC finds the densest region from all the circular regions. It is also provable that the edge of the circle must be incident on locations. As each circle in a plane can be uniquely determined by three points, or in this case by two ends of a diameter, the search space is $O(n^3+n^2)$ circles. The density for this method can also be expressed by Eq. (21).

4.2.4 Results

The results for different algorithms are shown in Fig. 5, with the grid region relevance calculated in the DRS preprocessing step shown in Fig. 6 and the regions retrieved shown in Fig. 7. Table 2 lists the average F_1-Score for the configuration of the 500 random locations. MBR has a tiny advantage over our DRS algorithm on

Table 2 Average F_1-Score for the case of 500 random locations

Region shape	F_1-Score		
	MBR	MBC	DRS
Rectangle	**0.9782**	0.9149	0.9615
Circle	0.9759	**0.9779**	0.9692
Street	**0.9873**	0.3784	0.9323
Ro-street	0.4234	0.3682	**0.9330**
Cross	0.6678	0.6341	**0.9193**
Ro-cross	0.7311	0.6472	**0.9208**

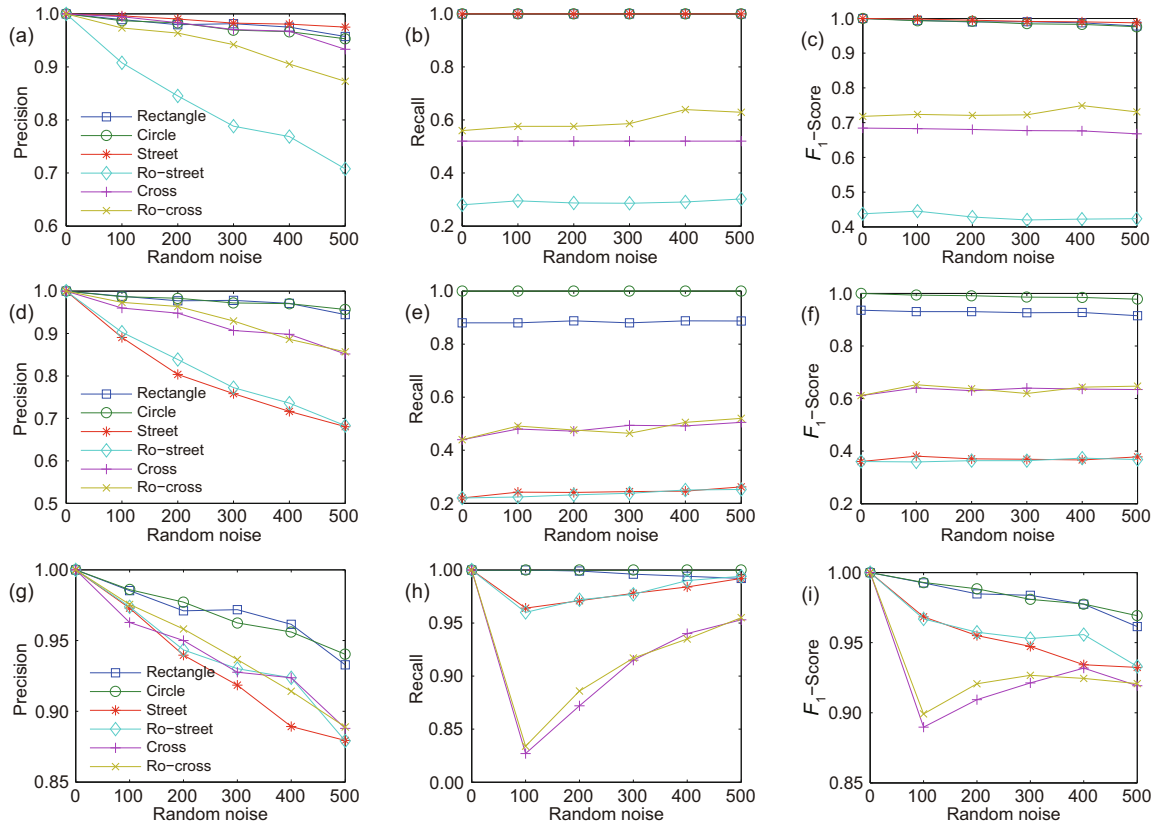

Fig. 5 Performance on the synthetic dataset with random noises of 0, 100, 200, 300, 400, and 500 locations: (a) MBR precision; (b) MBR recall; (c) MBR F_1-Score; (d) MBC precision; (e) MBC recall; (f) MBC F_1-Score; (g) DRS precision; (h) DRS recall; (i) DRS F_1-Score

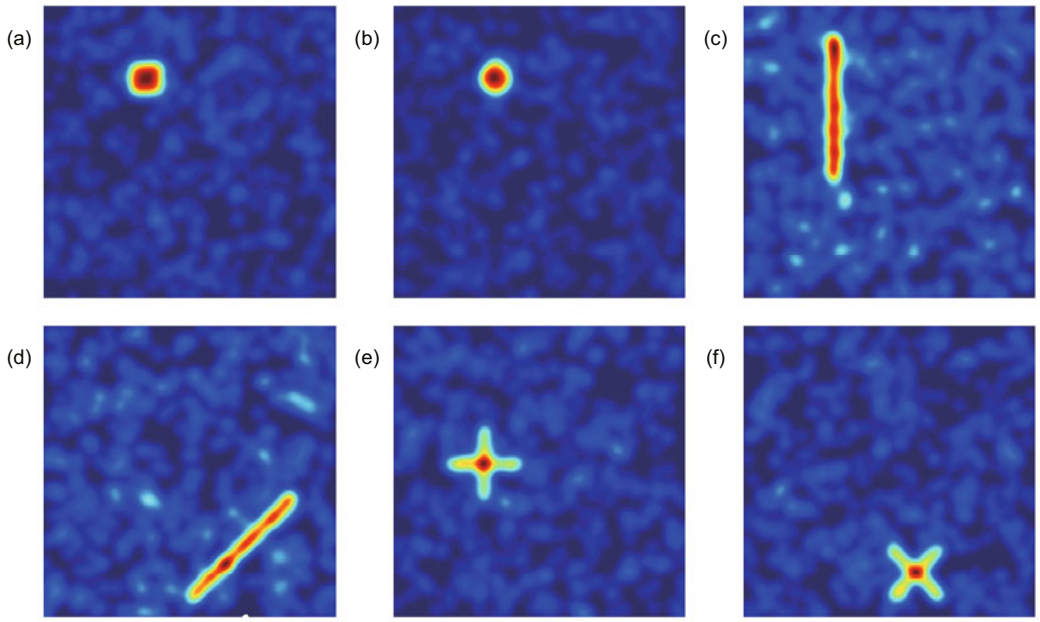

Fig. 6 Grid region relevance: (a) rectangle; (b) circle; (c) street; (d) ro-street; (e) cross; (f) ro-cross

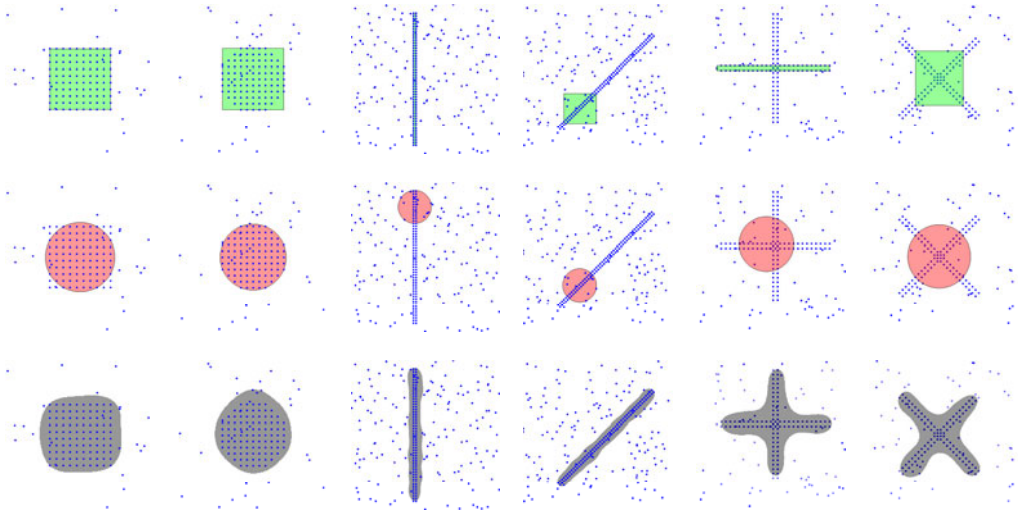

Fig. 7 Dense regions retrieved by MBR (green rectangle), MBC (pink circle), and DRS (gray region). References to color refer to the online version of this figure

regular shapes including rectangle, circle, and street, while its performance on other shapes is rather poor. The reason is that, MBR searches only the rectangles parallel to the coordinate axis. MBC is good at rectangle and circle, but fails in other shapes. Our DRS algorithm shows very good performance on all test shapes, proving its good ability in retrieving dense regions of arbitrary shapes.

Fig. 8 shows the time needed to retrieve the dense region using MBR, MBC, DRS preprocessing, and DRS search. We use a computer with a 4-core 3.20 GHz CPU and 16 GB memory. For the case of 500 random locations, MBR takes 1932.3 s for search and MBC takes 401.06 s on average, while DRS takes 15.83 s for the preprocessing step and 4.4484 s for the dense region search. The reason is that: in MBR, we need to calculate $O(n^4)$ rectangles and for each rectangle the time complexity for calculating the density is $O(n)$, making the time complexity $O(n^5)$ in total. We do some preprocessing by sorting the

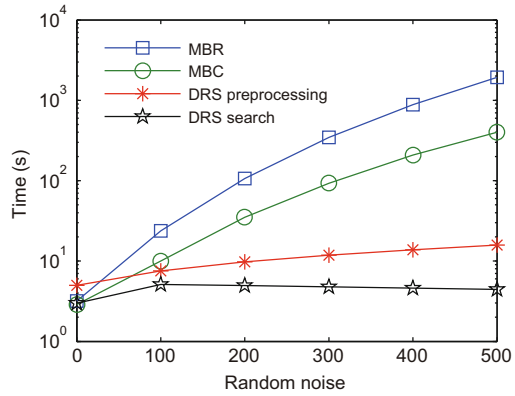

Fig. 8 Time cost of each method with different noises

locations by x and y coordinates respectively to reduce the time cost to $O(n^4)$, and the results are shown in Fig. 8. Similarly, we need to calculate the density of $O(n^3+n^2)$ circles, which is $O(n^4)$ in total. For DRS, however, the preprocessing step is to calculate the relevance for $a \times a$ grid regions, which takes $O(na^2)$ in total. Each iteration in the search step takes $O(\log a^2)$ to find a grid for the dense region, which contains $O(a^2)$ grids at most. So, the search step requires no more than $O(a^2)$ iterations, and therefore the time complexity is $O(a^2 \log a^2)$. In our experiment, we choose $a = 1000$ while n is about 600, so the time cost is about $O(n^2 \log n)$, which is much better than the others.

To summarize, DRS achieves better performance in a much shorter time.

4.3 Experiments on a real dataset

In this subsection, we compare different algorithms on a real-world dataset downloaded from dianping.com and analyze the performance of top-k and multi-keyword search.

4.3.1 Datasets

We use a real-world dataset downloaded from dianping.com, the largest website for location-based food and entertainment services in China. More than 20 000 restaurants, hotels, shops, etc., in the city of Hangzhou, together with 100 annotated keywords, are downloaded. In this dataset, 97 keywords are labeled on more than 10 locations, and the average number of labeled locations for a keyword is 218.2. Fig. 9 shows an example of the map with three different types of locations (red for snack, yellow for bakery, and blue for local cuisine). The map downloaded from Google Map covers an area of 13.0993 km $\times$ 9.8337 km. We map the locations to the first quadrant of the 2D coordinate system.

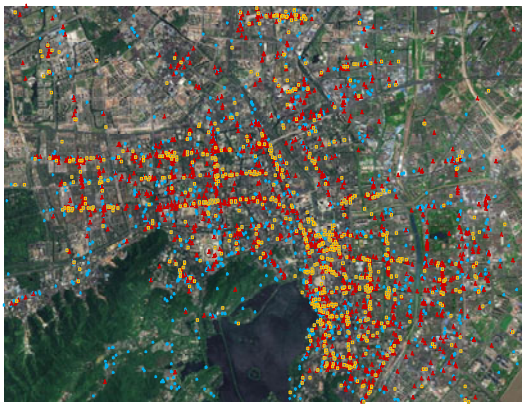

Fig. 9 Hangzhou map with three different types of locations (red for snack, yellow for bakery, and blue for local cuisine). References to color refer to the online version of this figure

4.3.2 Results and analysis

We run top-k DRS to search for 10 candidate regions using the keyword local cuisine. Fig. 10 shows the 10 retrieved regions as the results of re-ranking the top-20 candidate regions. We also show in Fig. 11 the results of running DRS 10 times, with locations in the retrieved region removed before the next run and results ranked by the retrieval order. While the retrieved regions in the two figures are almost identical (with the regions marked by red rectangles being different in the two figures), we notice that the rankings of dense regions in the two figures are quite different. The densest region in Fig. 11 (marked by blue rectangle) is the one containing the densest grid. However, after re-ranking, its location keyword density is surpassed by several other regions, as shown in Fig. 10. We conclude that using more candidate regions can improve the performance of the top-k query.

For the DRS multi-keyword search, we use all three keywords in Fig. 9 to query the map regions. The retrieved top-10 dense regions are displayed in Fig. 12. We can see that the regions retrieved, and ranked by multi-keyword relevance, are indeed densely populated with locations labeled with three keywords.

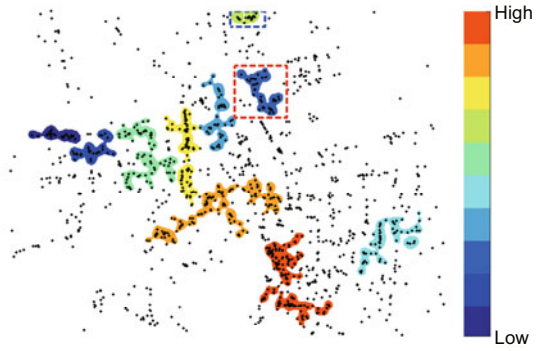

Fig. 10 Top 10 dense regions from 20 candidate regions

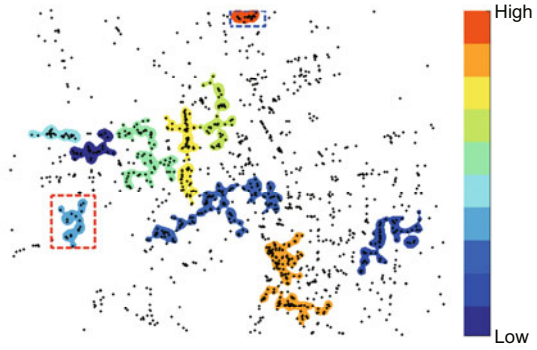

Fig. 11 Top 10 dense regions from 10 successive runs of DRS

5 Conclusions and future work

In this paper, we present a novel dense region search algorithm DRS for finding dense regions in location-based services. Different from traditional map search methods that consider mainly relevance between single locations with the query keywords, DRS takes into account the relevance and density for a region in the map. Results of experiments on synthetic and real social network datasets show that DRS achieves significant improvement in terms of dense region search accuracy and time cost. We believe that DRS can greatly enhance the value of a location-based search engine by using a region as a search result, which could give more information than a single location for users.

There are several interesting problems to be investigated in our future work:

1. Personalized region recommendation. If we know the current location of a user, it will be appropriate to consider the distance from his/her current location to the regions to be recommended. We can also take into account a user's preference and his/her historical visits in making a recommendation.

2. The application of region recommendation in social media. We can consider not only the location keyword relevance, but also the location's popularity, user rating, etc., by exploiting the power of social media.

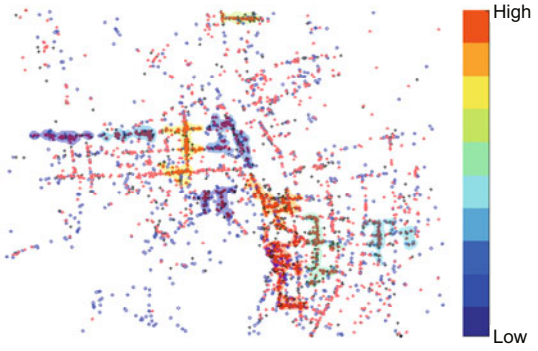

Fig. 12 Top 10 dense regions from three-keyword query

References

Aggarwal, A., Imai, H., Katoh, N., *et al.*, 1989. Finding k points with minimum spanning trees and related problems. Proc. 5th Annual Symp. on Computational Geometry, p.283-291. https://doi.org/10.1145/73833.73865

Agrawal, R., Gehrke, J., Gunopulos, D., *et al.*, 1998. Automatic subspace clustering of high dimensional data for data mining applications. *SIGMOD Rec.*, **27**(2):94-105. https://doi.org/10.1145/276304.276314

Ankerst, M., Breunig, M.M., Kriegel, H.P., *et al.*, 1999. Optics: ordering points to identify the clustering structure. *SIGMOD Rec.*, **28**(2):49-60. https://doi.org/10.1145/304182.304187

Aurenhammer, F., 1991. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Comput. Surv.*, **23**(3):345-405. https://doi.org/10.1145/116873.116880

Chen, L.S., Cong, G., Jensen, C.S., *et al.*, 2013. Spatial keyword query processing: an experimental evaluation. *Proc. VLDB Endowm.*, **6**(3):217-228. https://doi.org/10.14778/2535569.2448955

Chen, Y.Y., Suel, T., Markowetz, A., 2006. Efficient query processing in geographic web search engines. Proc. ACM SIGMOD Int. Conf. on Management of Data, p.277-288. https://doi.org/10.1145/1142473.1142505

Cheng, C.H., Fu, A.W., Zhang, Y., 1999. Entropy-based subspace clustering for mining numerical data. Proc. 5th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, p.84-93. https://doi.org/10.1145/312129.312199

Christoforaki, M., He, J., Dimopoulos, C., *et al.*, 2011. Text vs. space: efficient geo-search query processing. Proc.

20th ACM Int. Conf. on Information and Knowledge Management, p.423-432. https://doi.org/10.1145/2063576.2063641

Cong, G., Jensen, C.S., Wu, D.M., 2009. Efficient retrieval of the top-k most relevant spatial web objects. *Proc. VLDB Endowm.*, **2**(1):337-348. https://doi.org/10.14778/1687627.1687666

Ester, M., Kriegel, H.P., Sander, J., *et al.*, 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. Proc. 2nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, p.226-231.

Fan, J., Li, G.L., Zhou, L.Z., *et al.*, 2012. SEAL: spatio-textual similarity search. *Proc. VLDB Endowm.*, **5**(9):824-835. https://doi.org/10.14778/2311906.2311910

Feige, U., Seltser, M., 1997. On the densest k-subgraph problem. Technical Report, the Weizmann Institute, Rehovot.

Feige, U., Kortsarz, G., Peleg, D., 2001. The dense k-subgraph problem. *Algorithmica*, **29**:410-421. https://doi.org/10.1007/s004530010050

Guo, D.S., Peuquet, D.J., Gahegan, M., 2003. ICEAGE: interactive clustering and exploration of large and high-dimensional geodata. *GeoInformatica*, **7**(3):229-253. https://doi.org/10.1023/A:1025101015202

Hinneburg, A., Keim, D.A., 1999. Optimal grid-clustering: towards breaking the curse of dimensionality in high-dimensional clustering. Proc. 25th Int. Conf. on Very Large Data Bases, p.506-517.

Jones, C.B., Purves, R., Ruas, A., *et al.*, 2002. Spatial information retrieval and geographical ontologies an overview of the SPIRIT project. Proc. 25th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, p.387-388. https://doi.org/10.1145/564437.564457

Joshi, T., Joy, J., Kellner, T., *et al.*, 2008. Crosslingual location search. Proc. 31st Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, p.211-218. https://doi.org/10.1145/1390334.1390372

Khodaei, A., Shahabi, C., Li, C., 2010. Hybrid indexing and seamless ranking of spatial and textual features of web documents. *LNCS*, **6261**:450-466. https://doi.org/10.1007/978-3-642-15364-8_37

Komusiewicz, C., Sorge, M., 2012. Finding dense subgraphs of sparse graphs. Proc. 7th Int. Conf. on Parameterized and Exact Computation, p.242-251. https://doi.org/10.1007/978-3-642-33293-7_23

Lee, D.T., 1982. On k-nearest neighbor Voronoi diagrams in the plane. *IEEE Trans. Comput.*, **100**(6):478-487. https://doi.org/10.1109/tc.1982.1676031

Leung, K.W.T., Lee, D.L., Lee, W.C., 2011. CLR: a collaborative location recommendation framework based on co-clustering. Proc. 34th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, p.305-314. https://doi.org/10.1145/2009916.2009960

Li, Z.S., Lee, K.C., Zheng, B.H., *et al.*, 2011. IR-tree: an efficient index for geographic document search. *IEEE Trans. Knowl. Data Eng.*, **23**(4):585-599. https://doi.org/10.1109/tkde.2010.149

Mai, H.T., Kim, J., Roh, Y.J., *et al.*, 2013. STHist-C: a highly accurate cluster-based histogram for two and three dimensional geographic data points. *GeoInformatica*, **17**(2):325-352. https://doi.org/10.1007/s10707-012-0154-y

Ortega, E., Otera, I., Mancebo, S., 2014. TITIM GIS-tool: a GIS-based decision support system for measuring the territorial impact of transport infrastructures. *Exp. Syst. Appl.*, **41**(16):7641-7652. https://doi.org/10.1016/j.eswa.2014.05.028

Saoussen, K., Sami, F., Takwa, T., *et al.*, 2014. Tabu-based GIS for solving the vehicle routing problem. *Exp. Syst. Appl.*, **41**(14):6483-6493. https://doi.org/10.1016/j.eswa.2014.03.028

Schikuta, E., 1996. Grid-clustering: an efficient hierarchical clustering method for very large data sets. Proc. 13th Int. Conf. on Pattern Recognition, p.101-105. https://doi.org/10.1109/icpr.1996.546732

Shamos, M.I., Hoey, D., 1975. Closest-point problems. 16th Annual Symp. on Foundations of Computer Science, p.151-162. https://doi.org/10.1109/sfcs.1975.8

Son, L.H., 2014. Optimizing municipal solid waste collection using chaotic particle swarm optimization in GIS based environments: a case study at Danang city, Vietnam. *Exp. Syst. Appl.*, **41**(18):8062-8074. https://doi.org/10.1016/j.eswa.2014.07.020

Thomee, B., Rae, A., 2013. Uncovering locally characterizing regions within geotagged data. Proc. 22nd Int. Conf. on World Wide Web, p.1285-1296. https://doi.org/10.1145/2488388.2488500

Vaid, S., Jones, C.B., Joho, H., *et al.*, 2005. Spatio-textual indexing for geographical search on the web. Advances in Spatial and Temporal Databases, p.218-235. https://doi.org/10.1007/11535331_13

Wei, L.Y., Zheng, Y., Peng, W.C., 2012. Constructing popular routes from uncertain trajectories. Proc. 18th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, p.195-203. https://doi.org/10.1145/2339530.2339562

Wu, D.M., Yiu, M.L., Cong, G., *et al.*, 2012. Joint top-k spatial keyword query processing. *IEEE Trans. Knowl. Data Eng.*, **24**(10):1889-1903. https://doi.org/10.1109/icde.2011.5767861

Yuan, J., Zheng, Y., Xie, X., 2012. Discovering regions of different functions in a city using human mobility and POIs. Proc. 18th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, p.186-194. https://doi.org/10.1145/2339530.2339561

Zhang, F.Z., Wilkie, D., Zheng, Y., *et al.*, 2013a. Sensing the pulse of urban refueling behavior. Proc. ACM Int. Joint Conf. on Pervasive and Ubiquitous Computing, p.13-22. https://doi.org/10.1145/2493432.2493448

Zhang, Q., Kang, J.H., Gong, Y.Y., *et al.*, 2013b. Map search via a factor graph model. Proc. 22nd ACM Int. Conf. on Information and Knowledge Management, p.69-78. https://doi.org/10.1145/2505515.2505674

Zhou, Y.H., Xie, X., Wang, C., *et al.*, 2005. Hybrid index structures for location-based web search. Proc. 14th ACM Int. Conf. on Information and Knowledge Management, p.155-162. https://doi.org/10.1145/1099554.1099584