Big Data Research ••• (••••) •••–•••

Contents lists available at ScienceDirect

Big Data Research

www.elsevier.com/locate/bdr

ELAN: An Efficient Location-Aware Analytics System ☆

Yaxiao Liu [a,*], Henan Wang [a], Guoliang Li [a], Junyang Gao [a], Huiqi Hu [a], Wen-Syan Li [b]

[a] Department of Computer Science, Tsinghua University, Beijing, China
[b] SAP Labs, Shanghai, China

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
Location-aware analytics
Spatial index
Incremental algorithm
Big data

ABSTRACT

We demonstrate an Efficient Location-Aware aNalytics system (ELAN), aiming to provide users with location-aware data analytics services. For each user-selected spatial region, ELAN can instantly identify the most important functionality features of the region (e.g., business zones and residential areas) by efficiently analyzing the user-generated content (UGC) within the region. For each feature, ELAN can efficiently calculate the spatial boundary of the functional zone (denoted by a convex hull) in order to help users better understand the feature and furthermore we can identify the influential range of a certain feature. ELAN has many real-world applications, e.g., choosing business locations and popular regions discovery. There are two main challenges in designing a location-aware data analytics system. The first is to achieve high performance, as the region may contain a large amount of location-based UGC data. The second is to support continuous queries as users may continuously change the region by zooming in, zooming out, and panning the map. To address these challenges, we propose effective spatio-textual indexes and efficient incremental algorithms to support instant location-aware data analytics. We have implemented and deployed a system, which has been commonly used and widely accepted.

1. Introduction

Nowadays, the rapid growth of Mobile Internet has witnessed the widespread use of location-aware services, e.g., Google map search and Yelp local search. Meanwhile, with the popularity of location-based social networks (LBSN), e.g., Foursquare and Qzone, users are generating more and more user-generated content (UGC) with location information, e.g., check-ins and reviews at venues. Obviously the location-based UGC data plays an important role in location-based services, and we can take advantage of them to provide users with various location-aware analytics services. Although traditional applications can improve the functionality of the online map, such as region-based keyword queries [1] and finding places with specific category within user selected regions [2], they cannot fully support online location-aware data analytics. Obviously we can instantly identify the most important functionality features of users' interested region, e.g., business zones and residential areas, to help users better understand the selected region, by efficiently analyzing the UGC data. To this end, in this paper we demonstrate an Efficient Location-Aware aNalytics system (called ELAN), which can instantly analyze location-based UGC data.[1]

Fig. 1 shows a screenshot of our ELAN system. Given a user-selected spatial region, ELAN efficiently analyzes the location-based UGC data in this region and shows the important functionality features (e.g., *attorney* and *restaurants*) to the user using a word-cloud-style interface. For each feature, e.g., *attorney*, the system calculates the spatial boundary of the functional zone to the feature, as illustrated in Fig. 1. ELAN provides a new strategy to discover the map – it is not only a simple tool to find location-based information, but also a way to explore the aggregated features of different regions on the map. ELAN has many real-world applications. (1) Map Navigation: Suppose a user is visiting a strange place and wants to know the most important functionalities in the region. Obviously it is difficult for her to get such information using traditional map applications. Alternatively, ELAN can show the functionalities on the map, and the user can browse and compare different regions, and choose the most interesting region. (2) Business Location Selection: If a businessman wants to open a new supermarket in a region, she can use our ELAN system to browse the map and check whether there are competitive supermarkets in the region and whether there are potential customers close to the

☆ This article belongs to Analytics and Applications.

* Corresponding author.
E-mail addresses: liuyaxiao12@mails.tsinghua.edu.cn (Y. Liu), whn13@mails.tsinghua.edu.cn (H. Wang), liguoliang@tsinghua.edu.cn (G. Li), gaojy11@mails.tsinghua.edu.cn (J. Gao), hhq11@mails.tsinghua.edu.cn (H. Hu), wen-syan.li@sap.com (W.-S. Li).

[1] Our system is available at http://tsingnus.cs.tsinghua.edu.cn/elan/ (Beijing) and http://tsingnus.cs.tsinghua.edu.cn/elan/usa.html (USA).

http://dx.doi.org/10.1016/j.bdr.2016.08.001
2214-5796/

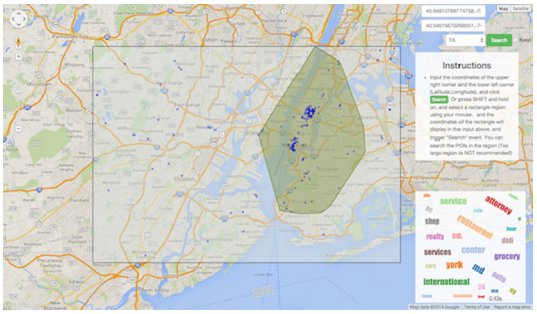

Fig. 1. A screenshot of ELAN.

region. (3) Public Facility Planning: ELAN can also provide suggestions for the government to do public facility planning. Suppose the government wants to build a sport center. The government can use ELAN to find the regions that are short of sport centers and have a large population.

It is rather challenging to design an efficient location-aware analytics system because (1) the user-selected region can be rather large and there may be a large amount of location-aware UGC data in the region; and (2) the user may continuously change the region (e.g., zooming in, zooming out, or panning the map) to make a comparison between different regions to find which region is more interesting. To address these two challenges, we propose effective index structures and efficient incremental algorithms to support instant location-aware analytics. We extend R-Tree [3] by incorporating textual information into the R-Tree nodes and propose KR-Tree. Compared with existing spatio-textual indexes, the objective of KR-Tree is to support location-aware analytics instead of location-based search, and KR-Tree only maintains some statistics spatio-textual information and has much smaller index sizes than existing spatio-textual indexes. When analyzing the features of each region, we calculate the importance of the features using the KR-Tree index instead of extracting all UGC data in the region, which can filter a large amount of irrelevant data. To support continuous queries, we propose efficient incremental algorithms that avoid many unnecessary computations. To demonstrate the effectiveness of our algorithms, we compare with the baseline algorithms, and the experimental results show that our method significantly outperforms the baseline approaches.

We have implemented and deployed a system based on some real-world datasets, which has been commonly used and widely accepted.

2. System overview

In this section, we first define the location-aware analytics query and a related problem – functional region analyzing, and then introduce the system architecture.

2.1. Location-aware analytics query

The underlying data in our system is location-based UGC data, which includes both textual descriptions and geographical locations, e.g., check-ins, points of interest (POIs), and user reviews at venues. Given a query region, the location-aware analytics query aims to identify the top-k important keywords sorted by keyword score (e.g., keyword frequency or inverse document frequency) from the UGC data in the region. Moreover, for each user selected keyword, we analyze all locations related to the keyword, and calculate the spatial boundary of the functional zone (denoted by the convex hull). We call this functionality *functional zone visualization*, which can help users better understand a region and a feature, and furthermore can be used to identify the influential range of

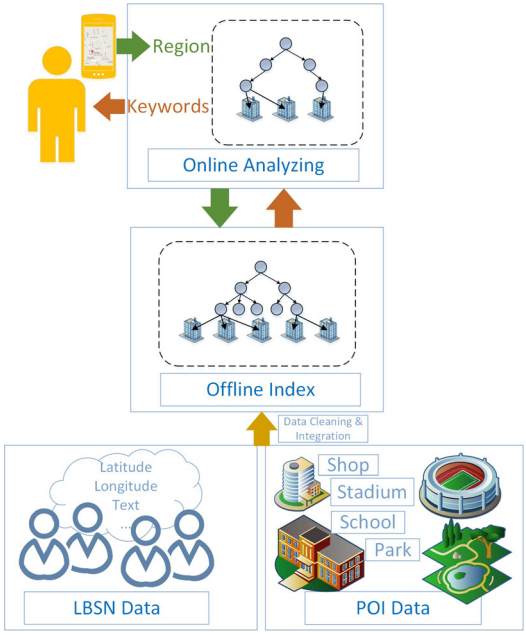

Fig. 2. Architecture.

a certain keyword or POI. The location-aware analytics query and functional zone visualization are two parts of the location-aware analytics system, and the system can extend its functions based on KR-Tree.

2.2. System architecture

Our system is composed of four main components – data cleaning and integration, offline index, online analytics and user interface, as shown in Fig. 2.

Data cleaning and integration. ELAN collects millions of real world POI data, and millions of user check-ins from the LBSNs (Tencent and Foursquare), cleans and integrates the unstructured data to structured data, which mainly includes locations and textual descriptions. For POI data, the text information is the description of the POI, e.g. name and category, and for user check-ins, it is the keywords of the check-ins.

Offline index. We propose KR-Tree as the offline index structure, which organizes locations using the R-Tree. In the internal and leaf nodes, we create textual indexes for location-aware UGC data in the region, which maintain the keyword statistics information. For each keyword, we keep its importance (or score) in the region. When analyzing the features of a region, we calculate the importance of the features using the index instead of the keywords of all locations. For each keyword, we also keep a list of data objects in the region that contain the keyword. When searching for locations related to a keyword, we both check the spatial information and the keyword, thus can filter irrelevant nodes in the KR-Tree, which can significantly improve the efficiency. KR-Tree shares the similar ideas with some other extended R-Tree structures, such as BR-Tree and IR-Tree in [4,5], but KR-Tree is designed to not only search objects with textual descriptions like other structures, but also efficiently and incrementally analyze the keywords of a region without retrieving all objects (see Sections 3.2 and 3.3).

Online analytics. The main task of answering a location-aware analytics query is to analyze the importance of the keywords appearing in the text information of the user-selected region. The top-k keywords are displayed to the user. In addition to the basic algorithm (see Section 3.1), we also propose an improved algorithm (see Section 3.2) based on the KR-Tree; and for the continuous queries, we propose an incremental algorithm based on

the improved algorithm (see Section 3.3). In the incremental algorithm, we distinguish the following three operations – zooming in, zooming out, and panning the map, and correspondingly design three algorithms, which can greatly improve the efficiency of the algorithm. The functional region analyzing shares the KR-Tree with location-aware analytics query, and searches locations, clusters, and convex hulls efficiently online (see Section 3.4).

User interface. To implement the user interface, we use AJAX, FastCGI and JSON to exchange data between user interface and the underlying system. Our system displays keywords within the user-selected region. When the user changes the region, e.g. zooming in, zooming out, and panning the region, the system updates the keywords based on the new region instantly. We show important keywords of a region using the word-cloud-style interface, and users can click the keyword and the system computes the spatial boundary of the functional zone to the selected keyword and displays the polygons on the map using Google Maps APIs.

3. Location-aware analytics algorithm

3.1. Basic algorithm

Given a user-selected region R, the basic algorithm first searches for all the leaf nodes in the R-Tree within the region, then scans the keywords of all locations in the leaf nodes, and sorts the keywords by the word score (Equation (1)) and finally returns top-k keywords.

$$Score = \frac{TF_{w \text{ in } R}}{TF_{\text{all words in } R}} * \log \frac{Count_{\text{all POIs}}}{Count_{\text{POIs containing } w}} \quad (1)$$

Equation (1) is similar to the term frequency and inverse document frequency (TF–IDF), but we treat the region as a document in the TF part, and treat the locations as documents in the IDF part. Since TF–IDF wins great reputation in search engines, we adopt the revised version as the keyword score, and other score equations maybe effective too.

Although the basic algorithm can analyze important keywords in a region, it has rather low performance because of the following two reasons.

(1) **Searching all locations & sorting all keywords.** The basic algorithm searches all locations in the region and extracts the keywords of each location online, and sorts all keywords by keyword score, which is time-consuming. To address this issue, we proposed the KR-Tree and use the threshold-based algorithm to improve the efficiency.

(2) **Analyzing from scratch.** When users are browsing the map, they may change the region frequently and continuously, and two search regions may have overlap. However the basic method has to compute the results from scratch. Intuitively, we can improve the efficiency by using an incremental algorithm to share the computations between different search regions.

3.2. Improved algorithm

As described in Section 2.2, each KR-Tree node contains a textual index including the keywords and the corresponding score in the region. To optimize the top-k retrieval process, for each KR-Tree node, we construct a sorted list of keywords in the textual index and their corresponding scores, where the keywords are sorted by their scores in a descending order. Given a user-selected region, we first identify all the nodes that are covered by the region. Then we utilize the sorted lists of these identified nodes to compute the top-k keywords as follows. A naive method is to enumerate every keyword, sum up the score of each keyword, and return the top-k keywords with the maximum combined scores. However this method requires to enumerate every keyword, which is rather expensive. To address this issue, we can use the threshold-based algorithm. We access the sorted list by the score in descending order and compute a lower bound based on seen keywords and utilize the bound to eliminate the unseen keywords with scores smaller than the bound. Obviously this method only scans some keywords in the sorted lists and avoids enumerating every keyword, and thus significantly improves the performance.

3.3. Incremental algorithm

Since users will continuously modify their queries to get expected results, we need to support the continuous queries efficiently. In general, users may possibly use three main operations on the map to do location-aware analytics – zooming in, zooming out and panning. Obviously, each time the system receives a location-aware analytics query, we can keep the nodes within the query region and utilize them to answer subsequent queries. In this way, when the system receives some similar location-aware analytics queries, the system can utilize the results of the recorded information from previous queries and avoid the search from scratch, which can improve the efficiency. We propose three algorithms with respect to the three operations.

Zooming in. When a user zooms in the map, the area of the region decreases and the new region must be a subregion of the original region. Given that the system has already recorded the last searched nodes, the new nodes which represent the new region is also a subset of the last searched nodes. Therefore, just by traversing the searched nodes and selecting the nodes that locates in the new region, the system can quickly return the new top-k results, and thus can avoid another time-consuming recursively search from the root node.

Zooming out. When a user zooms out the map, the area of the region increases and the original region becomes a subregion of the new region, which means it requires more nodes to represent the new region. The system can directly accept the recorded leaf nodes according to previous queries and from each of them to backtrack their parent nodes recursively to locate the new added leaf nodes. The algorithm also avoids a recursively search from the root node.

Panning. Panning the map is a more complicated process than zooming in and zooming out, but it shares the similar strategy with them. If the new region intersects with the original region, the system can directly use parts of the recorded leaf nodes which locate in the new region and then backtrack their parent nodes from them, and get all eligible leaf nodes. If the new region has no overlap with the original region, the system has to do a recursively search from the root node.

3.4. Functional zone visualization

We discuss how to support functional zone visualization. Obviously, there are many locations related to a keyword in a region which are distributed in the region, but how to find the subregions with the highest density? As we know, the locations of user check-ins are always different from the POIs, and how to get the most influential range of a POI? Functional zone visualization can solve the problems. We adopt four steps to analyze the functional regions.

(1) **Searching locations related to the keyword.** By using the KR-Tree, we can efficiently find the locations related to a keyword. In the search step, if the keyword inverted list of a node does not contain the keyword, we stop searching its sub-nodes, because the sub-nodes are not related to the keyword; if the inverted list contains the keyword, then we continuously search the locations related to the keyword, until all related locations are found.

Please cite this article in press as: Y. Liu et al., ELAN: An Efficient Location-Aware Analytics System, Big Data Res. (2016), http://dx.doi.org/10.1016/j.bdr.2016.08.001

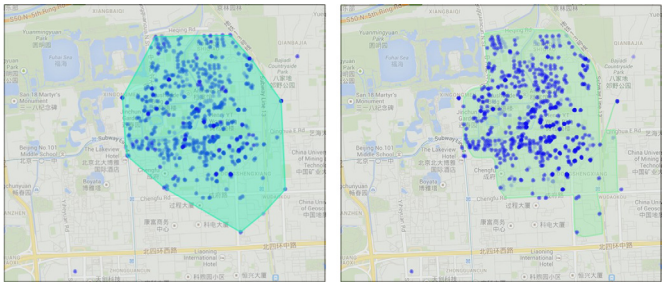

Fig. 3. Road-aware convex hull.

(2) **Clustering locations to groups.** Given a user-selected keyword, to find the sub-regions with the highest density to the keyword (i.e., the keywords are frequently appeared in the sub-regions), we cluster the locations related to the keyword by density. DBSCAN [6] was a density-based algorithm to discover clusters, and we adopt this algorithm as our clustering method. Since DBSCAN can only work with a given density, an important issue in clustering is how to set the density. If the density is too low, we can only get few large sub-regions; and if the density is too large, we can get many small sub-regions. As the density can be represented by the maximum distance between the locations, from the experimental study, we find that the default density can be set as 1/10 of the diagonal length of the rectangular region. Thus we can cluster the locations based on the density.

(3) **Computing the convex hull of a cluster of locations.** In step 2, we get some clusters of locations, but how to show them visually on the map? We formalize the problem as a convex hull problem, which finds the minimum convex polygon that contains the locations in a cluster. Graham [7] proposed an algorithm to find the convex hull of a simple polygon, called Graham Algorithm. However the algorithm does not consider the road information which is rather important in real-world applications, because in most cases, regions on the map are divided by the roads, thus we fit the convex hulls to the road network, and generate more rational sub-regions.

(4) **Mapping convex hull to road network.** We devise a road-aware algorithm to efficiently compute the convex hulls by taking road information into account when computing the convex hulls. In our previous work [8], we propose a real-time route recommendation method, and we can utilize this method to map the convex hulls to the road network as follows. We first find the shortest route between two nearest vertices of the polygon on the road network, and then replace the original polygon with the polygon on the road network. Our current system supports the road-aware convex hull construction in Beijing with the support of [8]. Fig. 3 shows the convex hull of the locations related to *Tsinghua University* in the user-selected region, while Fig. 3(a) shows the normal convex hull, and Fig. 3(b) maps the convex hull to the road network and can better describe the outline of Tsinghua University.

Heat-map-like method. In some particular conditions, the method above is too complicated and we may not need so accurate results, especially in influential range identify scenario, which is mostly used to recommend check-in POIs in LBSNs. The intuition idea is that the range who has most check-ins is the influential range of a POI, and a heat-map can represent the range visually, as shown in Fig. 4(a). Similar to the heat-map (a color overlay method on the map), we propose another functional zone visualization method – the heat-map-like method. Given a certain POI, firstly we get all check-ins using KR-Tree, and then calculate the score of each check-in location by counting the number of check-ins within a certain range (e.g. a circle with radius of 10 meters around the location) of the check-in location. Finally, the check-in

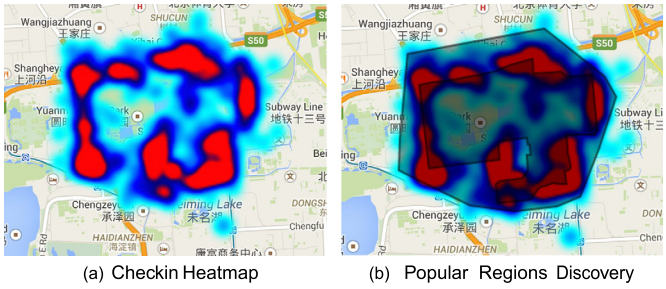

Fig. 4. Popular regions discovery. (The outlier convex hull is the influential range of the POI *Yuanmingyuan Park*, and the inner convex hull is the real-world boundaries of *Yuanmingyuan Park*.)

Table 1
Effectiveness evaluation.

User	1	2	3	4	5	6	7	8	9	10	Avg
Hits (%)	39	51	50	60	48	83	59	57	77	62	59

locations with relatively higher scores form a cluster, and we simply find the convex hull of the cluster (whose size is smaller than the input clusters of Step 3 above). The heat-map-like method is faster then the above method because it avoids iterative steps of the DBSCAN algorithm and generates smaller cluster than Step 2 above, while provides explainable influential ranges of the POIs, as shown in Fig. 4(b).

4. Experiment

We used millions of POIs in Beijing (1,228,736) and USA (9,700,676) to evaluate our methods, and each POI contains the point's latitude, longitude, address, category, description, and UGC. We conducted a user study to evaluate the effectiveness of ELAN and an experiment to verify the efficiency of our basic, improved, and incremental algorithms.

Effectiveness. To evaluate the effectiveness of ELAN, we first analyze the keywords of a particular region. We select a rectangular region ([40.84913799774758, −73.83293151855469] and [40.54615670268551, −74.36851501464844] as shown in Fig. 1), and get the following keywords – *restaurant, attorney, groccery, international, york, service* etc. Actually, this region contains the locations such as thousands of restaurants, hundreds of law offices and this region is located in the New York City, thus we can see the keywords appropriately represent the characters of the region.

To further verify the effectiveness of our system, we conducted a user study on Beijing POIs among 10 anonymous users who are familiar with Beijing. For each user, we randomly selected 20 regions in Beijing and in each region the system showed top-10 keywords, then the users marked up the keywords that match the characteristics of the region. Using the marked results as ground truths, we can compute the accuracy of our method and the result is shown in Table 1.

From Table 1, we can see on average about 59% keywords returned by our system can reasonably represent the features of a region and the results are satisfactory for most users.

Thus we can see that ELAN can correctly analyze the keywords of a region.

Precision of functional zone visualization. Functional zone visualization is an innovative application of location-aware analytics, and we evaluated the precision of functional zone visualization by comparing the regions that our method computed and the regions marked by real users (which was labeled by the Tencent company (qq.com)). As shown in Fig. 5, the colored regions are visualized by our method and the black regions are marked by real users in Tencent. We calculated the similarity of the regions to the same

keyword using the Equation (2). We used 63 well-known regions in Beijing to evaluate our method. The precision of our automatic functional zone visualization method was over 60%.

$$Similarity_{R1,R2} = \frac{Intersection_{R1,R2}}{Area_{R1} + Area_{R2} - Intersection_{R1,R2}} \quad (2)$$

Efficiency. To evaluate the efficiency, we conducted another experiment to compare the three analytics methods. We randomly selected 100 regions as the base region in Beijing, and for each region we zoom out (30 times continuously, Fig. 6(a)), zoom in (30 times continuously, Fig. 6(b)), and pan (30 times continuously, Fig. 6(c)), to make the map get larger, smaller, and same size but panned regions. The results are showed in Fig. 6. As shown in Fig. 6, we can see that the improved method and incremental method achieved rather high performance when we zoomed out, zoomed in, and panned the map, and significantly outperformed the baseline algorithm. In Fig. 6(d), on average, the incremental method was 3–5 times faster than the improved method, which in turn was 5–10 times faster than the basic method. For example, when we zoomed out the map (i.e., enlarge the region), the time of basic method increased greatly, but the other two methods only had a slightly increase, and they improved the efficiency by an order of magnitude.

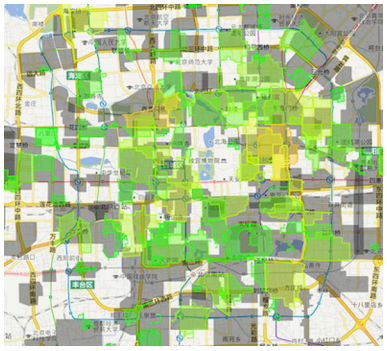

Fig. 5. Precision of functional zone visualization.

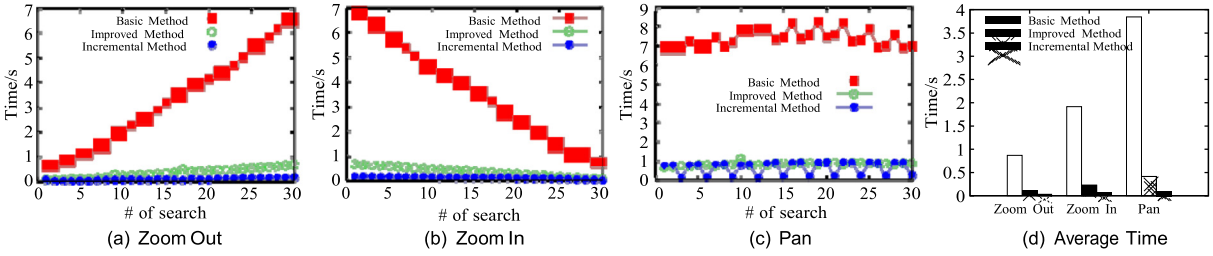

Fig. 6. Efficiency comparison.

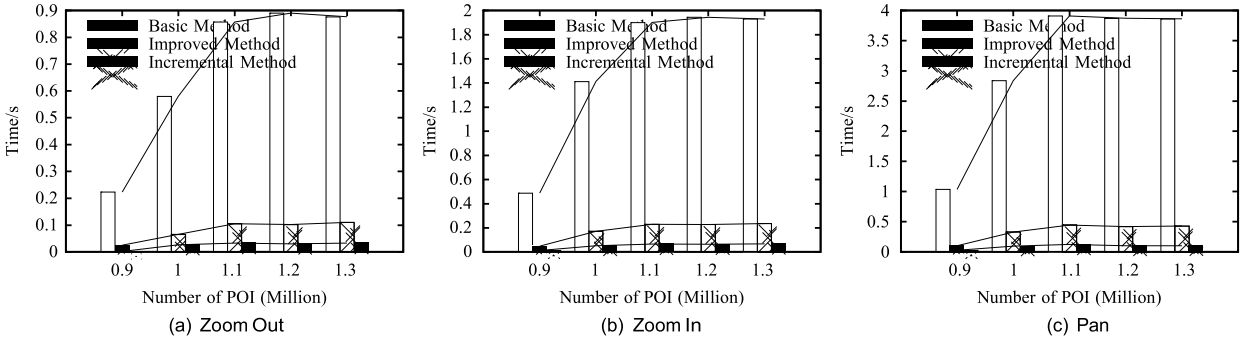

Fig. 7. Scalability experiment.

Scalability. To further evaluate the scalability of our methods, we conducted a series of experiments on different sizes of the dataset in Beijing. Fig. 7 shows the results. We can see that as the size of dataset increases, the time of the basic method consumes increases more quickly than the improved method and incremental method, and our methods have good scalability.

5. Applications

In this section, we discuss the applications of our system.

Business location selection. Businesses will regularly investigate detailedly to select the location for their new business, and traditional methods waste a lot of manpower and financial resources. For instance, assume a businessman wants to open a new bicycle store. Existing location-based systems cannot help her to find a good location. Alternatively, she can use our ELAN system. She first selects several regions and compares the differences of the characters of these regions based on the returned functionality features. Obviously the regions with keywords demanding for buying bicycles, e.g., *college* and *residential area*, may be better locations if *bicycle* is not a frequent keyword of the region, because in these regions the demand is greater than supply and the competition is not fierce. Theoretically, we can define the business location selection problem as follows. Given the type (keyword or category) of business and some candidate regions, the system recommends top-k regions for the type of business. The recommendation process includes three main steps: 1) Finding the other regions in the same city that contain the type of business and have similar area with the candidate regions. 2) Calculating the similarity of the regions found in step 1 with the candidate regions, and the similarity score is defined as the cosine similarity of the keyword vectors of the regions. 3) Selecting top-k regions of highest total similarity score as the business locations.

Public facilities planning. Public facility planning focuses on devising a planning standard, such as how many hectares of open space are required for a certain number of people in a district. ELAN can help users to compare the functionalities differences of different regions and provide suggestions on public facilities planning to the government, such as the number and location of schools, parks, supermarkets, etc. The public facilities planning

problem shares the similar definition with the business location selection problem, but we focus on public facilities and larger regions.

Travel recommendation. We will always meet the situation that when we are traveling to a new place, we need to find the restaurants, hotels, or entertainment locations along the road. On traditional online maps, we have to check all locations along the road, and make a complicated plan, but with our system, users just draw some regions along the road and get the characters along the road instantly. ELAN can be used for travel recommendation. We can support the users draw lines on the map, and ELAN automatically expands the lines to some nearby regions and integrates the keywords of all the regions.

High analyzing efficiency. We demonstrate that ELAN performs high efficiency using improved algorithm, and when users zoom in, zoom out or pan the region, ELAN can response to the new query more quickly because of the incremental algorithms. In both Beijing and the USA, no matter how large the selected regions are, ELAN can efficiently draw the word-cloud in less than 1 second in most circumstances.

Zone boundaries identification. Functional zone visualization can be used to automatically identify the zone boundaries. If we want to know the boundaries of Tsinghua University, the traditional method must mark the boundaries artificially, but with ELAN, our method can automatically identify the boundaries of Tsinghua University based on the location-based UGC data, as shown in Fig. 3. We will demonstrate more regions and keywords onsite.

Popular regions discovery. As we know, check-ins and tips can reflect the popularity of POIs. Traditional map applications cannot exactly identify an influential region of a POI and can just visualize the check-ins (or tips) using heat-map (see Fig. 4(a)). ELAN can not only visualize the popularity of the POI, but also can exactly identify the influential region of the POI (see Fig. 4(b)). Thus we can analyze the popularity of the regions more accurately with boundaries other than view them roughly using heat-map. Fig. 4 shows the popular regions discovery function of ELAN, which is available online now.[2]

6. Conclusion

In this demonstration paper, we demonstrated an efficient location-aware analytics system, which has many real-world map applications. We detailedly described the structure of the index, the improved algorithm, and the incremental algorithm. Our algorithm is not only efficient, but also can exactly extract keywords which can well reflect the characters of a region. We proposed the functional zone visualization algorithm, which can automatically discovery important features and identify the boundaries of the regions. ELAN has a widespread application and great prospects of research value and commercial value.

We can divide a region into some small regions. When we choose a relatively large region, such as a university, we can not only get the keywords throughout the region, in fact, by focusing on the degree of concentration of the keywords, we can find that this large region contains a number of small regions, while each one contains only one or a few keywords, such like cafeteria, dormitories, classrooms, playgrounds, etc. In this way, we can identify a small region of a single function. Of course, there are also some challenges, such as how to divide the region, how to measure the difference between two regions, and so on. To address these challenges, we will propose innovative ideas in our future work. Commercially, if the businesses fully understand the keywords of every region, they can choose the best location for their business. How to explore locations with the most business values through the identification of keywords of a region is also our future work. ELAN provides a new way of utilizing maps, and we can do much more in the future than what we can imagine now.

References

[1] R. Zhong, J. Fan, G. Li, K.-L. Tan, L. Zhou, Location-aware instant search, in: CIKM, 2012, pp. 385–394.
[2] J. Yuan, Y. Zheng, X. Xie, Discovering regions of different functions in a city using human mobility and pois, in: KDD, 2012, pp. 186–194.
[3] N. Beckmann, H. Kriegel, R. Schneider, B. Seeger, The *r**-tree: an efficient and robust access method for points and rectangles, in: SIGMOD, 1990, pp. 322–331, http://doi.acm.org/10.1145/93597.98741.
[4] G. Li, J. Xu, J. Feng, Keyword-based k-nearest neighbor search in spatial databases, in: CIKM, 2012, pp. 2144–2148, http://doi.acm.org/10.1145/2396761.2398590.
[5] Z. Li, K.C.K. Lee, B. Zheng, W. Lee, D.L. Lee, X. Wang, Ir-tree: an efficient index for geographic document search, IEEE Trans. Knowl. Data Eng. 23 (4) (2011) 585–599, http://dx.doi.org/10.1109/TKDE.2010.149.
[6] M. Ester, H. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: KDD, 1996, pp. 226–231, http://www.aaai.org/Library/KDD/1996/kdd96-037.php.
[7] R.L. Graham, F.F. Yao, Finding the convex hull of a simple polygon, J. Algorithms 4 (4) (1983) 324–331, http://dx.doi.org/10.1016/0196-6774(83)90013-5.
[8] H. Wang, G. Li, H. Hu, S. Chen, B. Shen, H. Wu, W. Li, K. Tan, R3: a real-time route recommendation system, Proc. VLDB 7 (13) (2014) 1549–1552, http://www.vldb.org/pvldb/vol7/p1549-wang.pdf.

[2] http://166.111.71.174:8000/regionalkeywords/heatvirtuallayer/ (Beijing).