A Spatial Clustering Approach for Efficient Landmark Discovery using Geo-tagged Photos

***Abstract*—Geo-tagged photos enable people to share their personal experiences while visiting various vacation spots through image sharing social networks like Flickr. The geo-tag information offers a wealth of information for capturing additional information on traveler behavior, trends, opinions and interests. In this paper, we propose a landmark discovery system that aims to discover popular tourist attractions in a city by assuming that the popularity of a tourist attraction is positively dependent on the visitor statistics and also the amount of tourist uploaded photos clicked on site. It is a known fact that places with lots of geo-tagged photos uploaded to Flickr are visited more often by social-media savvy tourists, who plan their trip based on others' experiences. We propose to build a system that identifies the most popular tourist places in a particular city by using geo-tagged photos collected from Flickr and recommend the same to the user. In this paper, we present the methodology of spatially clustering the geo-tagged images and present an analysis of algorithm performance in identifying landmarks and their popularity.**

Keywords: ***landmark ranking, spatial clustering, indexing, recommendation systems***

I. Introduction

Online photo sharing services such as Instagram and Flickr are gaining popularity due to the development and increased use of digital cameras and mobile phone cameras. In the Web 2.0 era, social media is a booming platform through which users communicate with each other or contribute to the information content on the Web. Volunteered Geopgraphic Information [1] is the information that has geographic content in it and is shared by users. VGI has been used for data analytics and knowledge discovery research in varied domains like tourism, disaster & crisis management, transportation, personalization and so on. The most common sources of VGI on the Web today are social networking sites like Flickr, OpenStreetMap, Twitter, Facebook, YouTube, Wikimapia, Foursquare etc. These services allow users to share pictures and associate each of those pictures with tags due to which, a huge collection of crowd-sourced and tagged photographs are available. A user can decide to attach geo-tags to an uploaded photo, to indicate where it was clicked. Most modern digital cameras automatically capture location information through an inbuilt GPS device, which can be automatically made available as geographical information, or alternatively this is specified by the users themselves. Geo-tags have been found to be very useful metadata format for injecting spatial information into Web applications.

Flickr being an online photo sharing service, allows users to upload their photographs. These photos are associated with metadata like the geo location at which the photo was captured, time at which the photo was clicked, size, camera that was used to click the photograph, title, text describing the photo, tags that are keywords for describing the photo and so on. This metadata helps in analyzing the activities of Flickr users. These images when plotted on a map can reveal the various places that have been visited by tourists. Using the spatial patterns from images, popular destinations of tourists can be discovered. In general, we aim to discover this information from geo-tagged Flickr photo data to develop an efficient travel recommendation system for providing the user with the most interesting (popular) landmarks.

The main motivation behind the proposed work is the fact that gathering information before an impeding vacation is still mainly a manual task focused on research and reading up online articles. When people have to travel to places that they've never visited, they will have several questions about planning the travel. Some help can be obtained from reading various travel magazines or by sharing their questions on social media. This process is not very efficient and requires a lot of manual work. It is a time consuming an tedious task for them to read several travelogues and summarize the reviews. Another disadvantage of consulting travel magazines is that they vary in language and makes it hard for the users to translate and interpret the information. Hence an automated trip planning service which can be customized to users' interests is desired. This service should also incorporate previous visitors' knowledge.

There are several online travel guides available currently and some of the popular ones are WikiTravel[1] and Yahoo Travel Guide[2]. WikiTravel provides users with information such as climate of the destination, popular landmarks in that area, places to shop in the region, good restaurants, different kinds of food that are unique to that place and also information about how to get there. One major disadvantage is that if WikiTravellers do not add information about any particular destination, other users will not find information about that destination. While WikiTravel depends on the users to share information, Yahoo Travel Guide provides information area wise. For each country, all the popular cities in that country are listed. Also, the information is provided by editors and not tourists who have visited those places. One of the major disadvantages of this system is that it only provides information about certain cities.

Practically, automated travel recommendation is a difficult task and depends on various factors like individual interests

[1]Wikitravel, wikitravel.org

[2]Yahoo Travel, yahoo.com/travel

of tourists, cost of the travel, duration, travel dates, amount of comfort desired by the users, etc. In our proposed system, we target two of the attributes for trip planning - location(city) and tourist attractions. Spacial clustering algorithms can be used to extract popular landmarks from the geo referenced images. Section II reviews related work on travel recommendation using geo-referenced photos. Section III gives the proposed approach to travel recommendation. Section IV provides the results and analysis of the work done. Section V provides a conclusion of the project and gives future work of the project.

II. Background and Related Work

Recommender systems, used for resource discovery and guidance in domains like music, books, movies, etc., and for personalization based on user preference are an area of active research interest. Sun et al. [2] proposed a system where geo-tag information was used to develop tag suggestion tools by finding tag correlation. These generated tags along with the geo-tags were used for other purposes like mining events and trends. In other works related to research in tourism, the collection of geo-tagged images and GPS history data is used to identify hotspots and landmarks, mine trajectory and movement and also plan itinerary and recommend trips [3], [4]. However, there is very little work in the area of automating the process of tour or trip planning. Kori et al. [5] proposed to use local blogs and generate travel guide, but users preferences were not taken into account to automate planning of trips.

Ji et al. [6] proposed a method to extract popular landmarks from blogs by using graphic models, while Zheng et al. [7] and Jaffe et al. [8] focused on visualizing, recognizing and describing a landmark by using geo-referenced images. Kennedy et al. [9] used the rich media available on the Web to generate representative sets of images for landmarks. Gao et al. [10] proposed the use of only tags associated with images in order to extract potential landmarks and also rank them according to their popularity among tourists.

Iwata et al. [11] used clustering techniques to discover sub-trajectories and Zheng et al. [12] proposed a HITS (Hypertext Induced Topic Search)-based model to mine interesting locations and travel patterns using GPS trajectory and considering the relationship between the user and the location. Route suggestion was addressed by Sun et al. [2], where DBSCAN clustering technique was first used to determine landmarks from a collection of geo-tagged images and then to provide a road based travel route between the extracted landmarks.

DBSCAN clustering is a density-based clustering approach and takes a lot of time to compute the clusters, especially when the data is very dense, as is the case with Flickr data. Our approach is to use an enhanced DBSCAN algorithm with R-tree indexing which results in much better performance and accuracy. We also employ effective noise elimination techniques to filter out unrelated photos from the collection and to identify photos uploaded by non-tourists (locals), and then cluster the resulting data to discover popular landmarks visited by tourists in six popular cities around the world. We also provide visualizations of the clustered data to highlight and verify the popularity of a landmark, as discovered by our algorithm.

III. Proposed System

In this section, we discuss in detail the various processes of proposed methodology. The system workflow is shown in figure 1.

A. Collecting geo-tagged images from Flickr

The first requirement of our work is to collect the photos posted on Flickr by tourists visiting various landmarks during their trip. We used certain APIs provided by Flickr, which support a limited number of operations like getting contacts, posting photos, get the list of favorites, getting photos from galleries etc. One of the services provided by Flickr that can be used to get publicly available geo-tagged photos is `flickr.photos.search`. Along with the photos, the tags associated with the photos and some metadata is also obtained. The metadata includes information about the owner of the photo, the date on which the photo was taken, the exact location at which the photo was taken (geo-tag) etc. This constitutes our initial data set which has to be processed further in order to obtain tagged photos that are relevant and also that can be attributed to only tourists.

B. Dataset filtering for Image-relevant tags

The main functionality of tagging is to describe photo content and is the easiest way of providing additional metadata to an image. These tags give us information about the photos like what kind of landmark it is, the name of the street on which the landmark is present on etc. Without the tags, computationally intensive image processing methods may need to be used in order to determine the content of an image. However, the downside of the tagging done by social media users is that, the tags often are very generic in nature like ”mySummer”, ”myTrip”, ”vacation” etc. Such tags do not give any information on the content of a photo and need to be identified and removed from the dataset. The problem in this is, the thousands of user uploaded, public photos per city available on Flickr and due to this, the identification of these generic tags cannot be done manually.

To find such generically tagged photos and to eliminate them from the dataset, we propose to use a natural language processing (NLP) technique called Tf-idf (term frequency - inverse document frequency). This technique is a statistical method that is used to determine how important a word is among a collection of words, in a document or in a collection of documents. Once all the unimportant tags for each photo are removed, the remaining tags can be used to extract information about the landmarks extracted after the clustering process. Here, we consider photos as documents and tags for each photo as the words in the document. Tf-idf is computed as given in equation (1).

$$Tfidf(tag_i) = tf(tag_i) \times idf(tag_i) \quad (1)$$

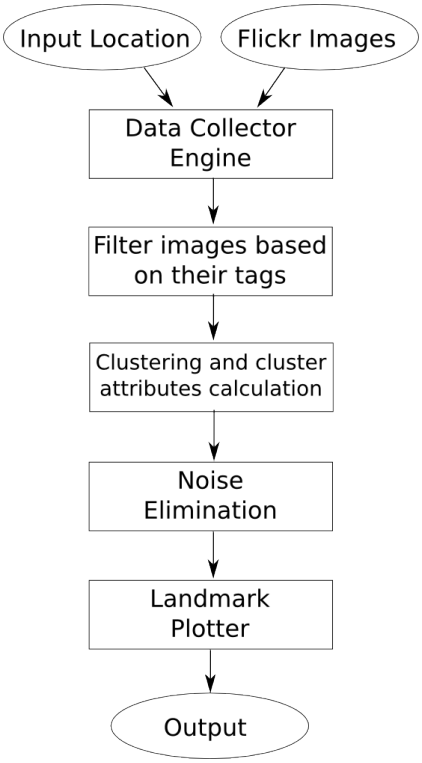


Fig. 1: *System Architecture Diagram*

The term frequency is calculated as either 1 or 0. It is 1 if a tag is present in a photo and 0 if it is not. Inverse document frequency is calculated using the equation (2).

$$idf(tag_i) = log_2(1 + \frac{D}{doc\_freq\{i\}}) \tag{2}$$

where tag_i represents the tag under consideration, *D* represents the total number of photos and *doc_freq*$\{i\}$ represents the number of photos that tag_i occurs in.

C. Clustering and Calculation of Cluster Attributes

Spatial clustering algorithm DBSCAN was used to group the images into clusters according to their geo-tags. DBSCAN algorithm requires two parameters - the neighborhood distance and the minimum number of points. By trial and error, a distance of 100 meters was used as the neighborhood distance and 20 was used as the minimum number of points. Since the distance between the locations where two different images were clicked is not known to us, we used the geo-locations of the images to compute the great circle distance between them using *haversine formula*. The haversine formula can be used to get the shortest distance between two points (landmarks) on the surface of a sphere (Earth), called the great-circle distance, using the longitude and latitude information [13].

Using this information, we perform DBSCAN clustering on the image data. Algorithm 1 shows the process of clustering the data [**?**]. Obtained clusters are further filtered using cluster attributes which are, number of users, the weight given to each user's content and total content weight for the cluster. These cluster attributes are calculated using the method proposed by Gao et al. [10] and are given below.

- *Number of users* (N_{user}) - For each cluster, (N_{user}) gives the number of users whose images are in that cluster. Greater the number of users, greater is the popularity of the region.
- *Content weight for each user* (CW_{user}) - This describes how much content each user provides to the cluster. Let CW_{user}(k) denote the content score for k^{th} user and N_{photo}(k) denote the number of photos provided by the k^{th} user. CW_{user}(k) is calculated using the formula (3).

$$CW_{user}(k) = log(N_{photo}(k) + 1) \tag{3}$$

- *Total content weight for the cluster TW)* - This describes the overall content provided by all users in the cluster. It is the sum of all CW_{user} values from all the users. It is given by the formula (4).

$$TW = \sum_{i=1}^{N-users} CW_{user}(k) \tag{4}$$

D. Clustering indexed data

Since the number of images that have to be clustered is large, DBSCAN takes a longer time to complete the clustering process. This is because of performing range queries for all the images in the data set. A range query is invoked to return all the images that are within a specific range of a particular image. If the images are not indexed according to their geo-locations, then a normal range query would require the program to traverse through all the images, compute their distances with respect to the given image and return those that satisfy the maximum radius condition. Hence, the run time complexity of DBSCAN without indexed data is $O(n^2)$. Using appropriate data structures for storing multidimensional data will reduce the time taken for clustering but might have a higher space complexity which is not a major concern.

Since one range query is performed for each point, while clustering n points, using an efficient indexing mechanism would reduce the overall runtime complexity. The clustering process can be improved by indexing images based on their geo-locations. However, we found that the common 1-dimensional indexes and hashes were not suited to our data, which is basically 2-dimensional. In this paper, we have explored two methods of indexing the spatial data, R-tree indexing and MongoDB's 2dsphere Index.

1) DBSCAN with R-tree indexing: R-tree or Rectangle-tree is a data structure used for indexing multi-dimensional data such as geographic co-ordinates. The basic working of R-trees is that nearby data objects are grouped together and represented within their minimum bounding rectangle in the next higher level of the tree [14]. So, each leaf of the tree are also rectangles and represents a single data object. R-trees are most efficient for retrieval of information. They support bounding box retrieval, nearest neighbor search and containment queries.

Algorithm 1 DBSCAN Algorithm with R-tree indexing

procedure DBSCAN($Data$, $epsilon$, $MinPoints$)
  $cluster \leftarrow 0$
  for each unvisited point P in $Data$ **do**
    mark P as visited
    $NeighborPoints \leftarrow$ REGIONQUERY$(P, epsilon)$
    if sizeof(NeighborPoints) $< MinPoints$ **then**
      mark P as noise
    else
      $cluster \leftarrow next\ cluster$
      EXPANDCLUSTER(P, $NeighborPoints$, $cluster$, $epsilon$, $MinPoints$)
    end if
  end for
end procedure

procedure EXPANDCLUSTER(P, $NeighborPoints$, $cluster$, $epsilon$, $MinPoints$)
  add P to $cluster$
  for each point P' in $NeighborPoints$ **do**
    if P' is not visited **then**
      mark P' as visited
      $NeighborPoints' \leftarrow$ REGIONQUERY$(P', epsilon)$
      if sizeof(NeighborPoints') $>= MinPoints$ **then**
        $NeighborPoints = NeighborPoints\ joined\ with\ NeighborPoints'$
      end if
    end if
    if P' is not in any cluster **then**
      add P' to $cluster$
    end if
  end for
end procedure

procedure REGIONQUERY(P, $epsilon$)
  return all points within P's $epsilon - range$
end procedure

Bounding box queries are useful in our context because they don't require the entire tree to be traversed. Based on the bounding box, sub trees are bypassed to move to the next sub-tree if they don't lie within the bounding box. An index was created for each of the cities where the images were indexed on their geo-tags. These indexes were serialized into index files for persistent storage so that they can be deserialized and used when the range queries were to be performed. For obtaining all the images that lie within a fixed radius of a particular image, the geographical co-ordinates of the given image is used along with the radius to get bounding box co-ordinates. These co-ordinates are then used for performing intersection queries on the indexes where only those images that lie within the bounding box rectangle are returned. Hence, using R-trees for indexing the Flickr images made clustering a lot faster. This is because most of the time during clustering is spent on performing range queries and indexing makes querying within a range run faster when compared to traditional DBSCAN.

2) MongoDB 2dsphere Index: MongoDB provides various indexing schemes including indexing for multi-dimensional data. One such index is the 2dsphere index which supports queries involving calculations on a sphere. It supports all kinds of geo-spatial queries like range, bounding box and nearest neighbor queries [15]. Using this indexing scheme gave interesting results as clustering took more time than the time taken by traditional DBSCAN without any indexing. This is because MongoDB takes a longer time for execution of 2dsphere index queries when compared to other type of queries.

E. Eliminating resident photos

The obtained set of images may contain photos that have been taken and uploaded by residents of that location and not tourists. Such photos have to be removed from the data set for accurate results. This is done using an entropy filtering method which is based on the fact that tourists stay in the city for a short period of time and upload photos that were taken within that span of time whereas residents may upload images which have been taken through out the year. Using this method, residents can be distinguished from the tourists among the users of the image set.

F. Determining cluster importance

Using the cluster attributes that were calculated, a cluster can be retained as a landmark or not. A cluster may be considered as a landmark only if it has a high importance factor. T_{CW} is the threshold value used to determine if the cluster is a landmark or not. If the cluster $cluster_i$ satisfies the condition represented by equation (5), it is considered to be a landmark.

$$TW(Cluster_i) > T_{CW} \tag{5}$$

G. Plotting landmark clusters

Once all noisy clusters are eliminated using cluster attributes, the median of all photos in each cluster was calculated to find the geo-location of the landmarks. This is plotted on a world map and city map in order to provide a visualization of the clustering results.

IV. Results and Analysis

A. Dataset

Flickr was queried for all public photos for 6 different popular tourist locations - Sydney, Paris, London, Singapore, New York and San Francisco. A total of 46,160 photos were collected. Among these 46,160 photos, 1,865 photos were eliminated as photos taken by residents. The statistics of the photo dataset from all six locations used in our work is given below.

- Total number of locations: 6

- Total number of photos: 46,160
- Total number of users: 4,324
- Total number of tourists: 4,301
- Total number of residents: 23
- Total number of photos taken by residents: 1,865
- Total number of photos taken by tourists: 44,295

For distinguishing tourists from residents, an entropy filtering method as described before was used. A threshold of 1.5 was used for the entropy. If the computed entropy was greater than this threshold, the user is labeled as a resident. Tf-idf was then used to eliminate common tags by using a threshold value of 2. So, all the tags that occur in more than one-fourth of the total photos will be discarded, as these would be generic tags. After removing photos based on tag analysis, DBSCAN algorithm was used to group the photos into clusters and cluster attributes were calculated. The threshold content score of a cluster that was used to eliminate noisy clusters was 8. A detailed statistics of the number of photos resulting after each step is shown in table I. The two indexing schemes for clustering yielded different results when compared to clustering on non indexed data which is tabulated in table II.

From table II, it can be seen that greater the number of photos to be clustered, greater is the clustering time. For cities that have similar number of photos, if the number of clusters is more, the time taken for traditional DBSCAN was more. So, we can conclude that clustering time depends on both the size of the data set and the number of clusters that can be formed. R-tree was found to be the most efficient indexing method for our data-set. A plot of the clustering time using the different approaches is shown in figure 2.

All the landmarks of the 6 locations were plotted on the world map and is shown in figure 4. Then, the filtered clusters for each of the six cities considered were plotted on the respective city map, again using Google Maps API. Figure 3a and 3b show the clusters that have been plotted for San Francisco and London respectively.

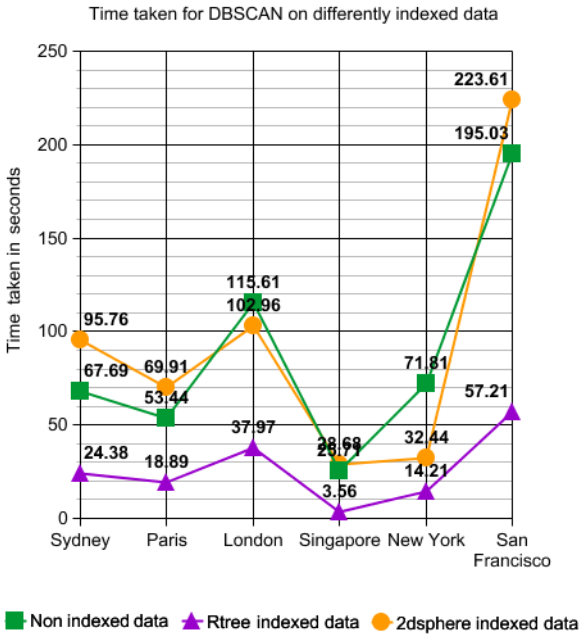


Fig. 2: *Clustering time analysis for different indexes*

V. Conclusion & Future Work

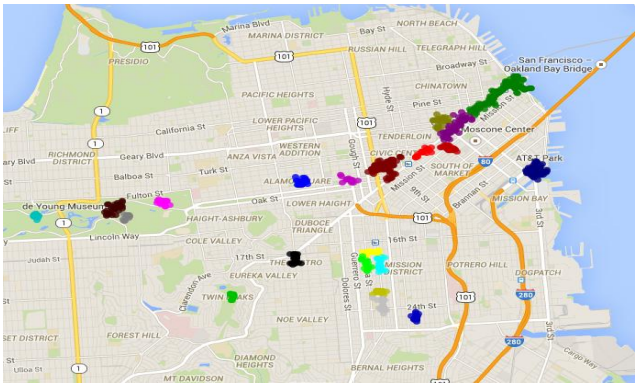
In this paper, we presented an efficient spatial clustering algorithm using R-tree indexing for landmark discovery from geo-tagged images. We collected public photos and their metadata for six different cities from Flickr. These photos were analyzed to distinguish tourists from residents and photos uploaded by residents were discarded as noise. The tags of the remaining photos were then analyzed using Tf-idf to

TABLE I: *Statistics of data pre-processing*

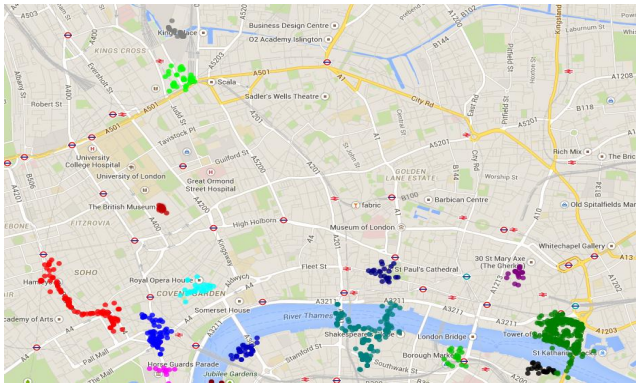
Location	No.of photos collected	No.of photos taken by tourists	No.of tourists' photos with tags	No.of photos remaining after tf-idf elimination	No.of clusters formed	No.of clusters considered	No.of photos in all clusters considered
Sydney	6249	6096	4775	4666	24	9	934
Paris	6600	6543	4248	4191	31	12	433
London	8050	7842	5889	5877	39	17	1084
Singapore	5656	4553	2726	2241	12	3	77
New York	8173	8165	5235	5234	17	5	123
San Francisco	11432	11096	8145	7785	62	22	1003

TABLE II: *Comparison of indexing methods*

Location	No.of photos	No.of clusters	DBSCAN on non-indexed data	DBSCAN on R-tree indexed data	DBSCAN on 2dsphere indexed data
Sydney	4849	24	67.69 seconds	24.38 seconds	95.76 seconds
Paris	4275	31	53.44 seconds	18.89 seconds	69.91 seconds
London	5889	39	115.61 seconds	37.97 seconds	102.96 seconds
Singapore	3199	12	25.71 seconds	3.56 seconds	28.68 seconds
New York	5243	17	71.81 seconds	14.21 seconds	32.44 seconds
San Francisco	8321	62	195.03 seconds	57.21 seconds	223.61 seconds



(a) Landmarks in San Francisco



(b) Landmarks in London

Fig. 3: Plotting the clusters on City maps

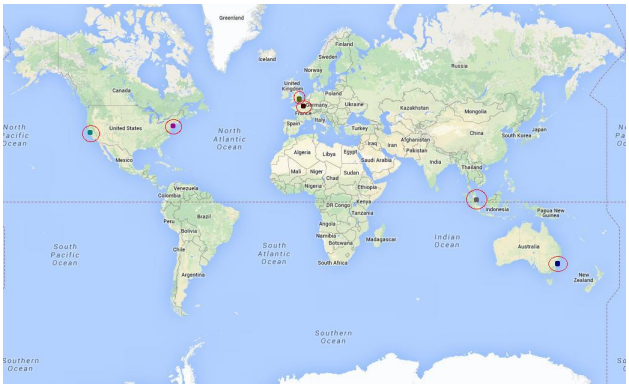


Fig. 4: *Landmarks of all six locations on world map*

eliminate more generic tags that occurred in most of the photos. DBSCAN clustering with R-tree indexing was then applied to the remaining photos and cluster attributes were calculated for each of the clusters. Using R-tree optimized the process of clustering due to faster data point retrieval. The clusters that satisfied minimum requirements were considered as landmarks and plotted on a location map using Google Maps API. As part of future work, we intend to incorporate an incremental clustering approach in order to avoid redundant clustering, so that the framework can be easily scalable, when new public photos are available from Flickr.

REFERENCES

[1] D. J. Coleman, Y. Georgiadou, J. Labonte *et al.*, "Volunteered geographic information: The nature and motivation of produsers," *International Journal of Spatial Data Infrastructures Research*, vol. 4, no. 1, pp. 332–358, 2009.

[2] Y. Sun, H. Fan, M. Bakillah, and A. Zipf, "Road-based travel recommendation using geo-tagged images," *Computers, Environment and Urban Systems*, 2013.

[3] S. Kisilevich, M. Krstajic, D. Keim, N. Andrienko, and G. Andrienko, "Event-based analysis of people's activities and behavior using flickr and panoramio geotagged photo collections," in *Information Visualisation (IV), 2010 14th International Conference*. IEEE, 2010, pp. 289–296.

[4] T. Rattenbury, N. Good, and M. Naaman, "Towards automatic extraction of event and place semantics from flickr tags," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2007, pp. 103–110.

[5] H. Kori, S. Hattori, T. Tezuka, and K. Tanaka, "Automatic generation of multimedia tour guide from local blogs," in *Advances in Multimedia Modeling*. Springer, 2006, pp. 690–699.

[6] R. Ji, X. Xie, H. Yao, and W.-Y. Ma, "Mining city landmarks from blogs by graph modeling," in *Proceedings of the 17th ACM international conference on Multimedia*. ACM, 2009, pp. 105–114.

[7] Y.-T. Zheng, M. Zhao, Y. Song, H. Adam, U. Buddemeier, A. Bissacco, F. Brucher, T.-S. Chua, and H. Neven, "Tour the world: building a web-scale landmark recognition engine," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 1085–1092.

[8] A. Jaffe, M. Naaman, T. Tassa, and M. Davis, "Generating summaries and visualization for large collections of geo-referenced photographs," in *Proceedings of the 8th ACM international workshop on Multimedia information retrieval*. ACM, 2006, pp. 89–98.

[9] L. S. Kennedy and M. Naaman, "Generating diverse and representative image search results for landmarks," in *Proceedings of the 17th international conference on World Wide Web*. ACM, 2008, pp. 297–306.

[10] Y. Gao, J. Tang, R. Hong, Q. Dai, T.-S. Chua, and R. Jain, "W2go: a travel guidance system by automatic landmark ranking," in *Proceedings of the international conference on Multimedia*. ACM, 2010, pp. 123–132.

[11] T. Iwata, S. Watanabe, T. Yamada, and N. Ueda, "Topic tracking model for analyzing consumer purchase behavior." in *IJCAI*, vol. 9, 2009, pp. 1427–1432.

[12] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *Proceedings of the 18th international conference on World wide web*. ACM, 2009, pp. 791–800.

[13] C. Robusto, "The cosine-haversine formula," *American Mathematical Monthly*, pp. 38–40, 1957.

[14] "R-tree," http://en.wikipedia.org/wiki/R-tree. [Online]. Available: http://en.wikipedia.org/wiki/R-tree

[15] "2dsphere indexes," http://docs.mongodb.org/manual/core/2dsphere/. [Online]. Available: http://docs.mongodb.org/manual/core/2dsphere/