IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

IMPLEMENTATION & ANALYSIS OF QUERY KEYWORD SUGGESTION USING PROXIMITY

[1]Prachi Shivaji Sable,[2]Prof. A. V. Mophare
[1]Student,[2]Assistant Professor
1Department of Computer Science and Engineering
[1]NBNSCOE, Solapur, India

Abstract: The Keyword suggestion in web search helps users to access relevant data without having knowledge in accordance with precisely expressing their queries or desired information. If we consider location of the user then we can recommend the user or produce results based on users current location. The keyword suggestion is the technique helps to retrieve documents which relates to information provided by user and user's location.

Search engine retrieves the documents, which are semantically relevant to the original query using KD graph and it have as result documents that correspond to objects near to the user's location. In this paper we have implementd and made analysis of location and proximity based keyword query suggestion framework using PHP and MySQL Database Along with use of XAMPP web Server. We have designed & also a weighted keyword-document graph, which grabs both the semantic relevance between keyword queries and the distance between the resulting documents and the query issuer's location.

***Index Terms* – KD Graph, PHP, MYSQL, XAMPP, Location, Proximity, Keyword, Query, Search Engine**

I. INTRODUCTION

In the Previous Paper we have mentioned that **Keyword suggestion** is most fundamental features of commercial web search engines. After submitting a Keyword query, the user may not be satisfied with the outcomes, so the Keyword suggestion module of the search engine suggests a set of m keyword queries that clarifies the user's search in the right path. Effective keyword suggestion methods are based on click information from query logs. New Keyword suggestions can be fixed according to their semantic relevance to the original keyword query. The first challenge of Keyword Suggestion framework is how to adequately measure keyword query similarity while considering the spatial distance factor. In accordance to previous query suggestion approaches Keyword Suggestion designs and uses a keyword-document bipartite graph (KD graph), which connects the keyword queries with their relevant documents. Diverse to all previous approaches which ignore locations, Keyword Suggestion adjusts the weights on edges in the KD graph to capture not only the semantic relevance between keyword queries, but also the spatial distance between the document locations and the query issuer's location the popular graph distance measure is applied for Personalized Page Rank (PPR) a.k.a. random walk with restart (RWR) distance on the KD graph, starting from the user supplied query, to find to set of m keyword queries with the highest semantic relevance to user supplied query and spatial proximity to the user location. In this Paper we have implemented and solved the problem using Partition Algorithm and Baseline Algorithm for producing result. The Problem statement is like following .

Problem Statement

To design the Location-aware Keyword query Suggestion framework, for suggestions relevant to the user's information needs that also retrieve relevant documents close to the query issuer's location

Objectives

- To search relevant document to the keyword using KD graph & current location aware suggestion using 'location aware edge weight adjustment' approach with help Partition-Based Algorithm.
- To show categories & images of suggested queries of nearby location using database, also show image or logo of particular thing on map using Google API.
- To conduct extensive experiments to evaluate the effectiveness and efficiency of Location aware Keyword Suggestion. The effectiveness of Location aware Keyword Suggestion framework compared to query suggestion that does not consider location.

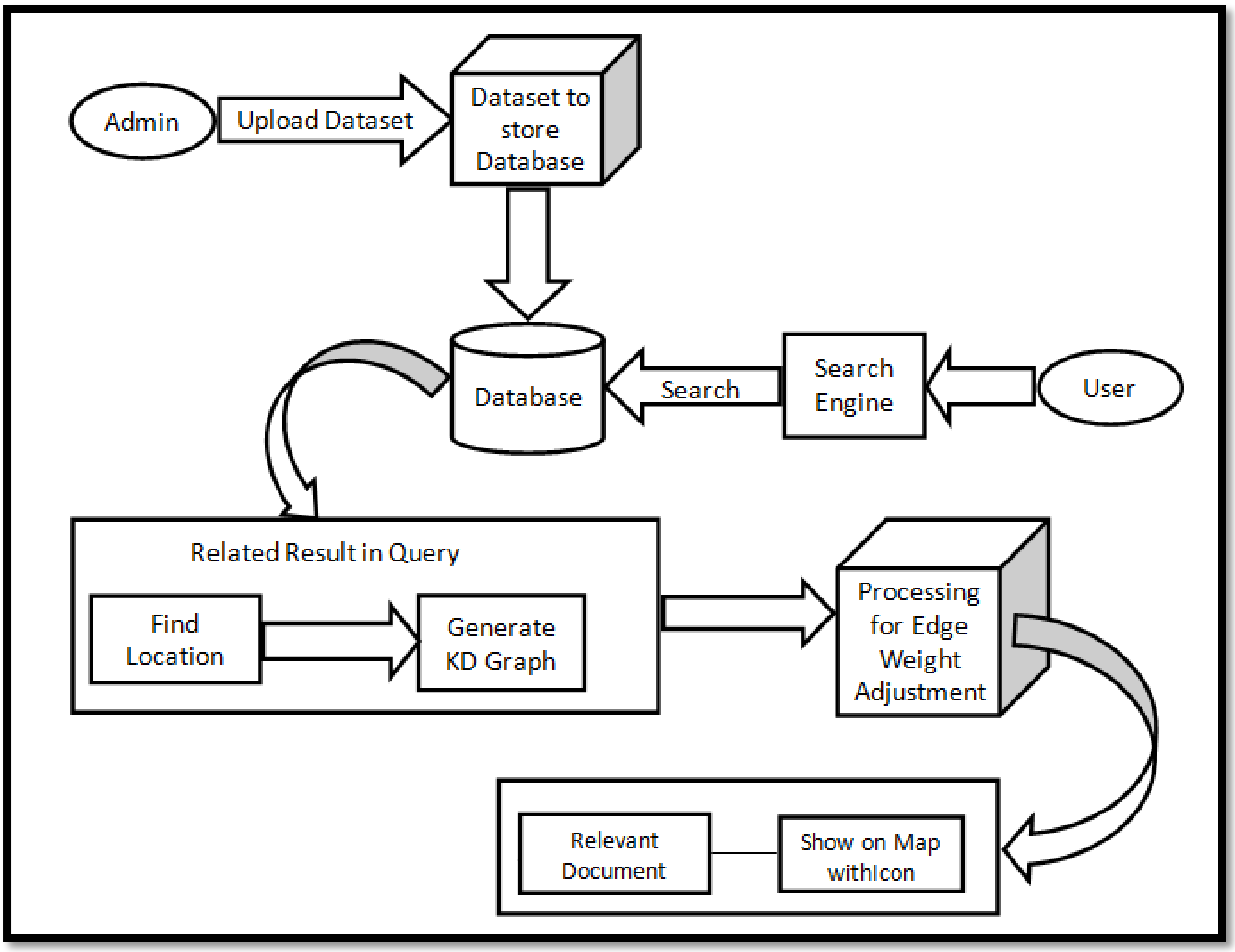
II. COMPARISON WITH EXISTING SYSTEM AND PROPOSED SYSTEM

S.N.	Parameter	Existing System	Proposed System
1	Database	Combined	Single
2	Database Name	Big Data	MySQL
3	Language	AJAX,JQUERY	HTML,PHP
4	Data Security	No	Yes
5	Storage Cost	Comparatively High	Comparatively Low
6	Computing Time	Slow	Fast
7	New Data Addition Facility	Automatic	Manual
8	Login Authentication	Not Required	Required
9	Search Method	Keyword Specific	Location based Keyword Specific

The existing System Works on the results showing from combining all the unnecessary data and we have removed it using our proposed system using Location Specific Keyword.

III. SYSTEM MODEL AND ALGORITHM OF PROPOSED SYSTEM

Our System woks on following flow model as shown in the following figure



Stepwise Description for the working of above model

1: Admin have to upload the dataset. Then preprocessing on this dataset and store all information into database.

2: User enters the query for searching the any document with location. Proposed system find the distance between the users enter query and document location. Find the distance of keyword and document.

3.Display Result: This distance display on map and recommend the place of the user.

ALGORITHM OF PROPOSED SYSTEM : Partition-Based Algorithm Algorithm

Algorithm details

K-means Algorithm

K-means algorithm is an iterative algorithm that tries to partition the dataset into K pre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group. It tries to make the intra-cluster data points as similar as possible while also keeping the clusters as different (far) as possible. It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster's centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum. The less variation we have within clusters, the more homogeneous (similar) the data points are within the same cluster.

K-means Clustering Method

If k is given, the K-means algorithm can be executed in the following steps:

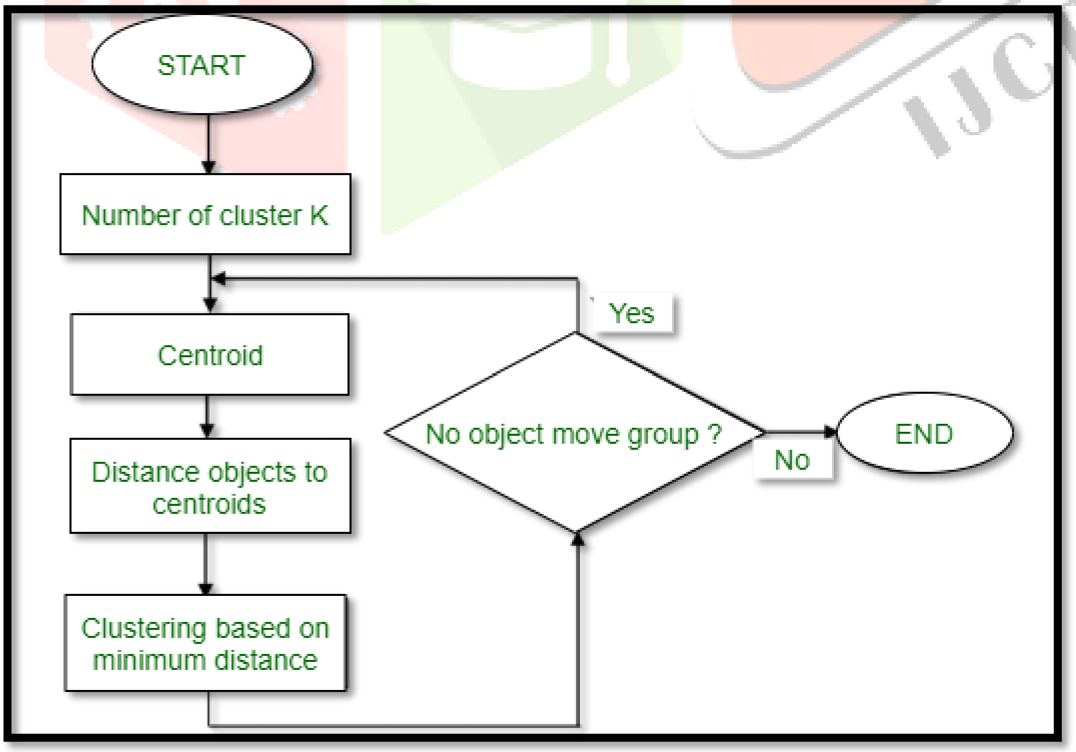
- Partition of objects into k non-empty subsets
- Identifying the cluster centroids (mean point) of the current partition.
- Assigning each point to a specific cluster
- Compute the distances from each point and allot points to the cluster where the distance from the centroid is minimum.
- After re-allotting the points, find the centroid of the new cluster formed.

Algorithm

```
  Input: Data set $X = \{x^{(1)}, x^{(2)}, \cdots, x^{(m)} | x^{(i)} \in \mathbb{R}^n\}$
  Output: Initial cluster centroids $\mu_{i=1,\cdots,k} \in \mathbb{R}^n$;
1 Choose two samples $x^{(i)}, x^{(j)}$ with largest distances as first two cluster
  centroids $\mu_1, \mu_2$,$X = X - \{x^{(i)}, x^{(j)}\}$;
2 for $i = 3, \cdots, k$ do // Choose other cluster centroids
3     Compute the sum of distance with all existing cluster centroids
      $D^{(j)} = \sum_{t=1}^{i-1} \|x^{(j)} - \mu_t\|^2$;
4     Set the sample $x^{(j)}$ with largest $D^{(j)}$ as the next cluster centroid $\mu_i$;
5     $X = X - x^{(j)}$;
6 end
7 return $\mu_1, \mu_2, \cdots, \mu_k$;
```

1. The first step when using k-means clustering is to indicate the number of clusters (k) that will be generated in the final solution.
2. The algorithm starts by randomly selecting k objects from the data set to serve as the initial centers for the clusters. The selected objects are also known as cluster means or centroids.
3. Next, each of the remaining objects is assigned to it's closest centroid, where closest is defined using the Euclidean distance between the object and the cluster mean. This step is called "cluster assignment step".
4. After the assignment step, the algorithm computes the new mean value of each cluster. The term cluster "centroid update" is used to design this step. Now that the centers have been recalculated, every observation is checked again to see if it might be closer to a different cluster. All the objects are reassigned again using the updated cluster means.
5. The cluster assignment and centroid update steps are iteratively repeated until the cluster assignments stop changing (i.e until *convergence* is achieved). That is, the clusters formed in the current iteration are the same as those obtained in the previous iteration.

Flow Chart


START
Number of cluster K
Centroid
Distance objects to centroids
Clustering based on minimum distance
No object move group ?
Yes
No
END

IV. IMPLEMENTATION

Language and Tools

Programming Language:**PHP**

Xampp Software for Server along with **Apache/Tomcat**

***METHODOLOGY** – GEO LOCATION*

The Geolocation API allows the user to provide their location to web applications if they so desire. For privacy reasons, the user is asked for permission to report location information.

WebExtensions that wish to use the Geolocation object must add the "geolocation" permission to their manifest. The user's operating system will prompt the user to allow location access the first time it is requested.

The Geolocation API is accessed via a call to navigator.geolocation; this will cause the user's browser to ask them for permission to access their location data. If they accept, then the browser will use the best available functionality on the device to access this information (for example, GPS).

Geolocation.getCurrentPosition(): Retrieves the device's current location.

Geolocation.watchPosition(): Registers a handler function that will be called automatically each time the position of the device changes, returning the updated location.

In both cases, the method call takes up to three arguments:

A mandatory success callback: If the location retrieval is successful, the callback executes with a **GeolocationPosition** object as its only parameter, providing access to the location data. An optional error callback: If the location retrieval is unsuccessful, the callback executes with a **GeolocationPositionError** object as its only parameter, providing access information on what went wrong. An optional object which provides options for retrieval of the position data. For further information on Geolocation usage, read Using the Geolocation API. Interfaces Geolocation The main class of this API — contains methods to retrieve the user's current position, watch for changes in their position, and clear a previously-set watch. **GeolocationPosition** Represents the position of a user. A GeolocationPosition instance is returned by a successful call to one of the methods contained inside Geolocation, inside a success callback, and contains a timestamp plus a **GeolocationCoordinates** object instance. **GeolocationCoordinates** Represents the coordinates of a user's position; a GeolocationCoordinates instance contains latitude, longitude, and other important related information. **GeolocationPositionError** A **GeolocationPositionError** is returned by an unsuccessful call to one of the methods contained inside **Geolocation**, inside an error callback, and contains an error code and message. **Navigator.geolocation** The entry point into the **API**. Returns a Geolocation object instance, from which all other functionality can be accessed.

SYSTEM DEMONSTRATION

RegistrationPage:

There are two types of registration. One, user registration in that userhastoenterFirst name,lastname,contact no, address,emailid&password then click on Sign Up button. After successful registration, all the data will be stored into database; another is Landmark registration in that landmark has to enter Place name, contact no, address, latitude & longitude, profile picture, document, city name, email-id & password. User & landmark can login only after successful registration.

Login Page:

Userhastologin intotheapplication for proceeding the application. User has to enter validEmailId and password.User can search data only after successful login.

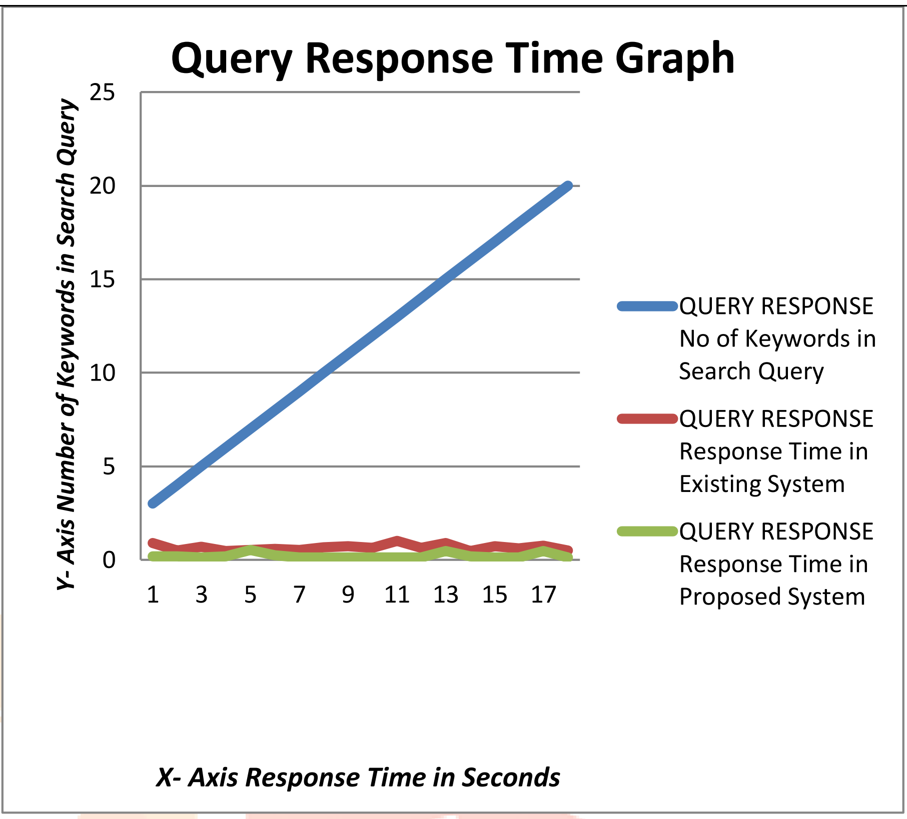
Response time:

Response time is the time an application server or API takes to respond to a user's request. The response time has a significant influence on the performance of an application. Response Time Testing measures the time taken for one system node to respond to the request of another. It is the time a system takes to reach a specific input until the process is over. For example, you have API, and you want to know exactly how much time it takes to execute it and return data. Response Time measures the server response of every single transaction or query.Response time starts when a user sends a request and ends at the time that the application states that the request has completed.

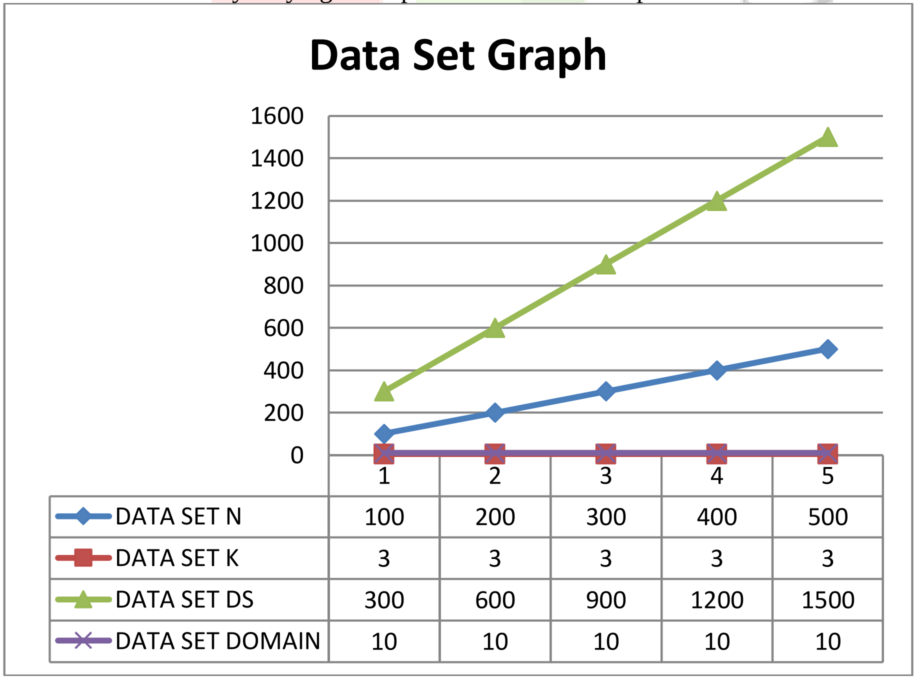
IV. RESULTS AND ANALYSIS

1.Response Time : Response time is very important factor in any System or Project,that means that after giving an input for processing ,how much time is required to process the output and the success of the system is depend on time.when the time factor is considerd the proposed system takes very less time as compared with existing system as shown in the following graph

QUERY RESPONSE TIME		
No of Keywords in Search Query	**Response Time in Existing System**	**Response Time in Proposed System**
3	0.89	0.18
4	0.49	0.18
5	0.71	0.12001
6	0.46	0.18
7	0.52	0.54
8	0.6	0.24
9	0.54	0.12001
10	0.67	0.12006
11	0.74	0.12001
12	0.64	0.12001
13	1.02	0.12001
14	0.63	0.12001
15	0.89	0.48007
16	0.47	0.18006
17	0.72	0.11999
18	0.62	0.12006
19	0.76	0.48007
20	0.49	0.12006



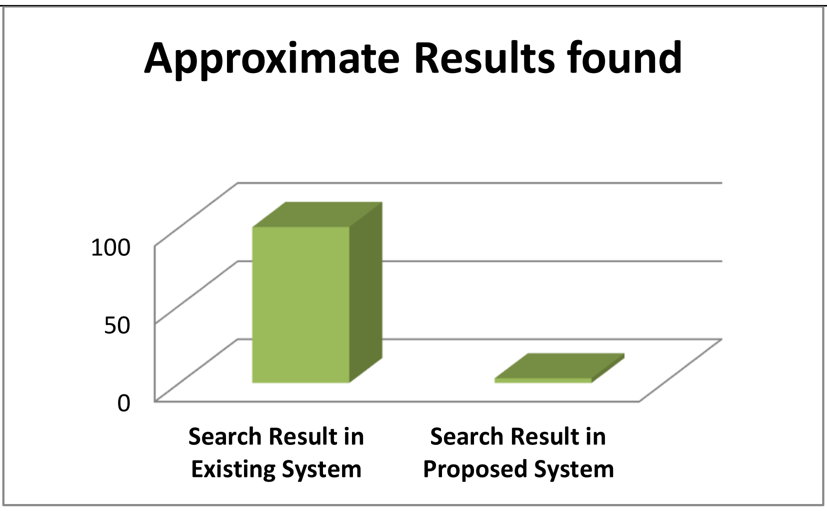
2. Datasets: We know that the Data was randomly generated. Each component is selected uniformly from [0-600]. Each point was randomly tagged with keywords.
The dataset is characterized by its following parameters
(1) size - N
(2) domain -, D
(3) dictionary size - U
(4) the number of keywords K associated with each point
We created datasets by varying these parameters for our empirical studies.



	1	2	3	4	5
DATA SET N	100	200	300	400	500
DATA SET K	3	3	3	3	3
DATA SET DS	300	600	900	1200	1500
DATA SET DOMAIN	10	10	10	10	10

3. Resultant Approximate Result :
Here After any Search is completed in existing system results generated are more than 100 or 1000 and the user gets confused which result is true or desired result ,but in proposed system results are very simple and accurate that will be very specific 2 or 3

entries the above graph shows the Approximate no of results for any keyword in correspondence with existing system and proposed system

Approximate Results found After Search		
Search Result in Existing System	100	
Search Result in Proposed System	3	

V. Acknowledgment

I profoundly grateful to Prof. A.V. Mophare for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion. I would like to express my deepest appreciation towards Principal Dr. S.D. Navale, Prof. B.R.Solunke, HOD and PG coordinator department of computer science & engineering. I must express my sincere heartfelt gratitude to all staff members of computer science & engineering department who helped me directly or indirectly during this course of work. Finally, I would like to thank my family and friends, for their precious support.

VI. References

[1] Z. Chen, H. T. Shen, X. Zhou, Y. Zheng, and X. Xie. Searching trajectories by locations: an efficiency study. In Proceedings of the 2010 ACM SIGMOD International Conference on Management of data, pages 255–266, 2010.

[2] H.-P. Hsieh and C.-T. Li. Mining and planning time-aware routes from check-in data. In Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, pages 481–490, 2014.

[3] W. T. Hsu, Y. T. Wen, L. Y. Wei, and W. C. Peng. Skyline travel routes: Exploring skyline for trip planning. In Mobile Data Management (MDM), 2014 IEEE 15th International Conference on, volume 2, pages 31–36, 2014.

[4] Q. Yuan, G. Cong, and A. Sun. Graph-based point-of-interest recommendation with geographical and temporal influences. In Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, pages 659–668, 2014.

[5] M. Ye, P. Yin, W.-C. Lee, and D.-L. Lee. Exploiting geographical influence for collaborative point-of-interest recommendation. In Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval, pages 325–334.

[6] Y.-T. Wen, P.-R. Lei, W.-C. Peng, and X.-F. Zhou. Exploring social influence on location-based social networks. In Data Mining (ICDM), 2014 IEEE International Conference on, pages 1043–1048. IEEE, 2014.

[7] H. Tong, C. Faloutsos, and J.-Y. Pan, "Fast random walk with restart and its applications," in Proc. 6th Int. Conf. Data Mining, 2006, pp. 613–622.

[8] Y. Fujiwara, M. Nakatsuji, M. Onizuka, and M. Kitsuregawa, "Fast and exact top-k search for random walk with restart," Proc. VLDB Endowment, vol. 5, no. 5, pp. 442–453, Jan. 2012.

[9] K. Avrachenkov, N. Litvak, D. Nemirovsky, E. Smirnova, and M.Sokol, "Quick detection of top-k personalized PageRank lists," in Proc. 8th Int. Workshop Algorithms Models Web Graph, 2011, vol. 6732, pp. 50–61.

[10] P. Berkhin, "Bookmark-coloring algorithm for personalized pagerank computing," Internet Math., vol. 3, pp. 41–62, 2006.

[11] PARTITION-BASED ALGORITHMS WOJCIECH WIECZOREK , https://link.springer.com/chapter/10.1007/978-3-319-46801-3_3

[12] Clustering Based K Means Algorithm https://www.geeksforgeeks.org/partitioning-method-k-mean-in-data-mining/

[13] Partitioning Algorithms used in Clustering - https://medium.com/analytics-vidhya/partitional-clustering-181d42049670