Tagged Keyword Search in Multi-dimensional Datasets

P. Mohan babu[1], B. Vishnu Vardhan[2]

[1]M.Tech Scholar, [2]Assistant Professor

Department of CSE,
Prasad V Potluri Siddhartha Institute of Technology,
Kanuru, Vijayawada, A.P, India

***ABSTRACT*: Keyword-based search in text-rich multi-dimensional datasets facilitates many novel applications and tools. In this paper, consider objects that are tagged with keywords and are embedded in a vector space. For these datasets, Study queries that ask for the tightest groups of points satisfying a given set of keywords. propose a novel method called ProMiSH (Projection and Multi Scale Hashing) that uses random projection and hash-based index structures, and achieves high scalability and speedup. In this present an exact and an approximate version of the algorithm. Our experimental results on real and synthetic datasets show that ProMiSH has up to 60 times of speedup over state-of-the-art tree-based techniques.**

***KEYWORDS*: Projection and multi scale hashing (ProMiSH), nearest keyword search (NKS), locality sensitive hashing (LSH).**

1.INTRODUCTION

Objects (e.g., images, chemical compounds, documents, or experts in collaborative networks) are often characterized by a collection of relevant features, and are commonly represented as points in a multi-dimensional feature space. For example, images are represented using color feature vectors, and usually have descriptive text information (e.g., tags or keywords) associated with them. In this paper, consider multi-dimensional datasets where each data point has a set of keywords. The presence of keywords in feature space allows for the development of new tools to query and explore these multi-dimensional datasets. In this paper, study nearest keyword set (referred to as NKS) queries on text-rich multi-dimensional datasets. An NKS query is a set of user-provided keywords, and the result of the query may include k sets of data points each of which contains all the query keywords and forms one of the top-k tightest cluster in the multi-dimensional space. For a query Q = fa; b; cg, the set of points f7; 8; 9g contains all the query keywords fa; b; cg and forms the tightest cluster compared with any other set of points covering all the query keywords. Therefore, the set f7; 8; 9g is the top-1 result for the query Q. NKS queries are useful for many applications, such as photo-sharing in social networks, graph pattern search, geolocation search in GIS systems1 and so on. The following are a few examples.

1) Consider a photo-sharing social network (e.g., Facebook), where photos are tagged with people names and Fig. 1. An example of an NKS query on a keyword tagged multi-dimensional dataset. The top-1 result for query fa; b; cg is the set of points f7; 8; 9g. locations. These photos can be embedded in a high dimensional feature space of texture, color, or shape. Here an NKS query can find a group of similar photos which contains a set of people.

2) NKS queries are useful for graph pattern search, where labeled graphs are embedded in a high dimensional space for scalability. In this case, a search for a sub graph with a set of specified labels can be answered by an NKS query in the embedded space.

3) NKS queries can also reveal geographic patterns. GIS can characterize a region by a high-dimensional set of attributes, such as pressure, humidity, and soil types. Meanwhile, these regions can also be tagged with information such as diseases. An epidemiologist can formulate NKS queries to discover patterns by finding a set of similar regions with all the diseases of her interest. Formally define NKS queries as follows. Nearest Keyword Se. Although existing techniques using tree-based indexes suggest possible solutions to NKS queries on multi-dimensional datasets, the performance of these algorithms deteriorates sharply with the increase of size or dimensionality in datasets. Therefore, there is a need for an efficient algorithm that scales with dataset dimension, and yields practical query efficiency on large datasets. In this paper, propose ProMiSH (short for Projection and Multi-Scale Hashing) to enable fast processing for NKS queries. In particular, develop an exact ProMiSH (referred to as ProMiSH-E) that always retrieves the optimal top-k results, and an approximate ProMiSH (referred to as ProMiSHA) that is more efficient in terms of time and space, and is able to obtain near-optimal results in practice. ProMiSH-E uses a set of hash tables and inverted indexes to perform a localized search. The hashing technique is inspired by Locality Sensitive Hashing (LSH), which is a state-of-the-art method for nearest neighbor search in high-dimensional spaces. Unlike LSH-based methods that allow only approximate search with probabilistic guarantees, the index structure in ProMiSH-E supports accurate search. ProMiSH-E creates hash tables at multiple bin-widths, called index levels. A single round of search in a hash table yields subsets of points that contain query results, and ProMiSH-E explores each subset using a fast pruning-based algorithm. ProMiSH-A is an approximate variation of ProMiSH-E for better time and space efficiency. Evaluate the performance of ProMiSH on both real and synthetic datasets and employ state-of-the-art VbR_-Tree and CoSKQ as baselines. The empirical results reveal that ProMiSH consistently outperforms the baseline algorithms with up to 60 times of speedup, and ProMiSH-A is up to 16 times faster than ProMiSH-E obtaining near-optimal results.

1.1 EXISTING SYSTEM:

- Location-specific keyword queries on the web and in the GIS systems were earlier answered using a combination of R-Tree and inverted index.
- Felipe et al. developed IR2-Tree to rank objects from spatial datasets based on a combination of their distances to the query locations and the relevance of their text descriptions to the query keywords.
- Cong et al. integrated R-tree and inverted file to answer a query similar to Felipe et al. using a different ranking function.

Nearest Keyword Set:
Let D _ Rd be a d-dimensional dataset with N points. For any o 2 D, it is tagged with a set of keywords _(o)={v1; ::; vt} _ V, where V is a dictionary of Uunique keywords. For any oi, oj 2 D, the distance between oi and oj is measured by their L2-norm (i.e., Euclidean distance) as dist(oi; oj) = ||oi- oj||2. Given a set of data points A _ D, r(A) is the diameter A and is defined by the maximum distance between any two points in A,

$$r(A) = \max_{\forall o_i, o_j \in A} \|o_i - o_j\|_2.$$

1.2 DISADVANTAGES OF EXISTING SYSTEM:

- These techniques do not provide concrete guidelines on how to enable efficient processing for the type of queries where query coordinates are missing.
- In multi-dimensional spaces, it is difficult for users to provide meaningful coordinates, and our work deals with another type of queries where users can only provide keywords as input.
- Without query coordinates, it is difficult to adapt existing techniques to our problem.
- Note that a simple reduction that treats the coordinates of each data point as possible query coordinates suffers poor scalability.

1.3 PROPOSED SYSTEM:

- Consider multi-dimensional datasets where each data point has a set of keywords. The presence of keywords in feature space allows for the development of new tools to query and explore these multi-dimensional datasets.
- Nearest keyword set (referred to as NKS) queries on text-rich multi-dimensional datasets. An NKS query is a set of user-provided keywords, and the result of the query may include k sets of data points each of which contains all the query keywords and forms one of the top-k tightest cluster in the multi-dimensional space.
- In this propose ProMiSH (short for Projection and Multi-Scale Hashing) to enable fast processing for NKS queries. In particular, develop an exact ProMiSH (referred to as ProMiSH-E) that always retrieves the optimal top-k results, and an approximate ProMiSH (referred to as ProMiSH-A) that is more efficient in terms of time and space, and is able to obtain near-optimal results in practice.
- ProMiSH-E uses a set of hashtables and inverted indexes to perform a localized search.

1.4 ADVANTAGES OF PROPOSED SYSTEM:

- Better time and space efficiency.
- A novel multi-scale index for exact and approximate NKS query processing.
- It's an efficient search algorithms that work with the multi-scale indexes for fast query processing.
- Conduct extensive experimental studies to demonstrate the performance of the proposed techniques.

2 IMPLEMENTATION

MODULES:

- ❖ Admin Module
- ❖ User Module

MODULES DESCSRIPTION:

Admin Module

In this module, admin has to login with valid username and password. After login successful he can do some operations such as view all user, their details, add documents with details, add images with details, List All documents and images uploaded, List all users search history ,view similar group data, view similar user searched keywords in cluster, view keyword and query details and fetched ratio, view ratio results for keywords, generate r* tree for searched keyword (keyword as root and fetched results are sub roots), find IR tree for both images and documents(image or document category and its titles).

User Module

In this module, there are n numbers of users are present. User should register before doing some. After registration successful he can login by using valid user name and password. Login successful he will do some operations like view profile details, search for both documents and images using keyword and view the results, view search history, view all nearest keywords and fetched data.

Present an algorithm for finding top-k tightest clusters in a subset of points. A subset is obtained from a hashtable bucket as explained. Points in the subset are grouped based on the query keywords. Then, all the promising candidates are explored by a multi-way distance join of these groups. The join uses rk, the diameter of the kth result obtained so far by ProMiSH-E, as the distance threshold.
Explain a multi-way distance join with an example. A multi-way distance join of q groups {g1; …., gq} finds all the tuples {o1;I,…. ox;j , oy;k,……, oq,l} such that Vx, y: ox,j 2 gx, oy,k 2 gy; and ||ox,j – oyk||2 <_ rk. Fig.(a) shows groups {a, b, c} of points obtained for a query Q={a, b, c} from a subset F0. show an edge between a pair of points of two groups if the distance between the points is at most rk, e.g, an edge between point o1 in group a and point o3 in group b. A multi-way distance join of these groups finds tuples {o1, o3, o9} and {o10, o3, o9}. Each tuple obtained by a multi-way join is a promising candidate for a query.

Group Ordering

A suitable ordering of the groups leads to an efficient candidate exploration by a multi-way distance join. First perform a pair wise inner joins of the groups with distance threshold rk. In inner join, a pair of points from two groups are joined only

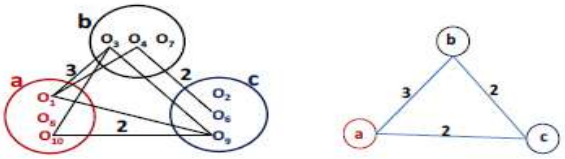

Fig (a) Pairwise inner joins (b) A graph representation

Above Fig.(a) a, b, and c are groups of points of a subset F0 obtained for a query Q={a,b, c} A point o in a group g is joined to a point o' in another group g0 if‖o-o'‖< _ rk. The groups in the order {a, c, b} generate the least number of candidates by a multi-way join. (b) A graph of pair wise inner joins. Each group is a node in the graph. The weight of an edge is the number of point pairs obtained by an inner join of the corresponding groups.

Complexity Analysis of ProMiSH:

In this section, first analyze the query time complexity and index space complexity in ProMiSH. Then discuss how ProMiSH prunes the search space.

Query Time Complexity:

Given a set of d-dimensional data points D, assume data points are uniformly distributed in the buckets of a hashtable, and keywords of each data point are uniformly sampled from the dictionary. Suppose D has N data points, each data point has t keywords, and the keywords are sampled from a dictionary of U unique keywords. Let Nv be the number of data points with keyword v. The expectation of Nv is computed as follows,

$$E[N_v] = \sum_{i=1}^{N}(1-(1-\frac{1}{U})^t) = N(1-(1-\frac{1}{U})^t).$$

Time complexity of ProMiSH-E. Let L be the index level applied in the index structure of ProMiSH-E, H(s) be the hashtable at scale s 2 {0, 1, ……..,L – 1}, B be hashtable size, and Nb;v be the number of data points with keyword v lying in a bucket b among B buckets. Suppose ProMiSH-E applies m random unit vectors. Since ProMiSH-E generates 2m signatures for each data point, the expectation of Nb,v under uniformity assumptions is estimated as below,

$$E[N_{b,v}] = \frac{2^m}{B}E[N_v].$$

3. RESULTS

Fig 1: Home Page

Fig 2: Admin Home

Fig 3: Here admin can add the documents with details & description.

Fig 4: Admin can view R tree Structure.. keyword and Results tree

Fig 5: Admin can IR tree Structure search results

Fig 6: User Home

Fig 7: User can View All Keywords and Related data

4. CONCLUSION & FUTURE WORK:

Proposed solutions to the problem of top-k nearest keyword set search in multi-dimensional datasets. Proposed a novel index called ProMiSH based on random projections and hashing. Based on this index, developed ProMiSH-E that finds an optimal subset of points and ProMiSH-A that searches near-optimal results with better efficiency. Our empirical results show that ProMiSH is faster than state-of-the-art tree-based techniques, with multiple orders of magnitude performance improvement. Moreover, our techniques scale well with both real and synthetic datasets. Ranking functions. In the future, plan to explore other scoring schemes for ranking the result sets. In one scheme, may assign weights to the keywords of a point by using techniques like tf-idf. Then, each group of points can be scored based on distance between points and weights of keywords. Furthermore, the criteria of a result containing all the keywords can be relaxed to generate results having only a subset of the query keywords. Disk extension. Plan to explore the extension of ProMiSH to disk. ProMiSH-E sequentially reads only required buckets from Ikp to find points containing at least one query keyword. Therefore, Ikp can be stored on disk using a directory-file structure. In this create a directory for Ikp. Each bucket of Ikp will be stored in a separate file named after its key in the directory. Moreover, ProMiSH-E sequentially probes HI data structures starting at the smallest scale to generate the candidate point ids for the subset search, and it reads only required buckets from the hash table and the inverted index of a HI structure. Therefore, all the hash tables and the inverted indexes of HI can again be stored using a similar directory file structure as Ikp, and all the points in the dataset can be indexed into a B+-Tree [36] using their ids and stored on the disk. In this way, subset search can retrieve the points from the disk using B+-Tree for exploring the final set of results.

REFERENCES

[1] W. Li and C. X. Chen, “Efficient data modeling and querying system for multi-dimensional spatial data,” in GIS, 2008, pp. 58:1–58:4.

[2] D. Zhang, B. C. Ooi, and A. K. H. Tung, “Locating mapped resources in web 2.0,” in ICDE, 2010, pp. 521–532.

[3] V. Singh, S. Venkatesha, and A. K. Singh, “Geo-clustering of images with missing geotags,” in GRC, 2010, pp. 420–425.

[4] V. Singh, A. Bhattacharya, and A. K. Singh, “Querying spatial patterns,” in EDBT, 2010, pp. 418–429.

[5] J. Bourgain, “On lipschitz embedding of finite metric spaces in Hilbert space,” Israel J. Math., vol. 52, pp. 46–52, 1985.

[6] H. He and A. K. Singh, “Graphrank: Statistical modeling and mining of significant subgraphs in the feature space,” in ICDM, 2006, pp. 885–890.

[7] X. Cao, G. Cong, C. S. Jensen, and B. C. Ooi, “Collective spatial keyword querying,” in SIGMOD, 2011.

[8] C. Long, R. C.-W. Wong, K. Wang, and A. W.-C. Fu, “Collective spatial keyword queries: a distance owner-driven approach,” in SIGMOD, 2013.

[9] D. Zhang, Y. M. Chee, A. Mondal, A. K. H. Tung, and M. Kitsuregawa, “Keyword search in spatial databases: Towards searching by document,” in ICDE, 2009, pp. 688–699.

[10] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, “Locality-sensitive hashing scheme based on p-stable distributions,” in SCG, 2004.