# Study on Similarity Search

Aakriti Singhal
*Department of Computer Science Engineering*
*MGM COET, Noida, Uttar Pradesh, India*

Elasha
*Department of Computer Science Engineering*
*MGM COET, Noida, Uttar Pradesh, India*

Medhavi Pandey
*Department of Computer Science Engineering*
*MGM COET, Noida, Uttar Pradesh, India*

**Abstract - Content-Aware Search can be defined as a method for searching a word or a phrase in such a way that we are able to extract wanted data from the search engine, example Google. It explores issues that are related to searching, refining and filtering of feature-rich, text and non-text data types. Content Based Similarity search explains that for given a query object we need to find similar objects, that is, objects with similar features (not exact features).This paper basically focuses on the various types of algorithms available for searching. The search can include both text and non-text data. Content based search system can be of different kinds-Full Similarity Search and Partial Similarity Search. There are various algorithms that work efficiently for the searching mechanism. Most of the algorithms are based on the nearest neighbour problem.**

**Keywords- Similarity Search, Partial Similarity Search, NN, K-n match, MQKNN, SBA, GSTBA**

## I. Introduction

As the data over the World Wide Web is growing at a very fast rate therefore it is very important that whatever a user enters as his query, the output should be relevant to what he is looking for. For this purpose, Similarity Search system is used that works for both text and non text data (image, video etc).

Most existing work have modelled this problem as the nearest neighbour (NN) problem, which considers the distance between the query object and the data objects over a fixed set of features. Nearest neighbour search (NNS), also known as Proximity search, is an optimization problem for finding closest points in metric spaces.

One of the drawbacks of the NN problem is that it leaves the partial similarities uncovered. Since in reality, it is difficult to match a query object in its entirety to the data object therefore the concept of partial similarity came into existence. A partial similarity models similarity search as matching the query object and the data objects in 'n' dimension, where 'n' is a given integer smaller than the dimensionality of the data space and these 'n' dimensions are determined dynamically to make the query object and the data objects match best.

The rest of the paper is organized as follows. Study on various algorithms of similarity search is explained in section II. Related work is given in section III. Concluding remarks are given in section IV.

## II.Studied Algorithm's

### *2.1. K-n match problem*

One of the drawbacks of NN problem was that it was based on fixed set of features so in order to overcome this issue the concept of partial similarity was proposed. To address this, the k-n match problem was designed which considers the distance between the query object and the data object given in n dimensions. The value of n is smaller than the dimensionality and these dimensions are determined dynamically to return the best matched answers.

The main points to be noted are that-

- Each dimension in the query list has to be sorted.
- The attributes are retrieved in ascending order of their differences to the corresponding attributes of the query.

The k-n match problem was implemented on an image database, COIL-100 that consist 100 images. 54 features like color histograms and moments were extracted for the purpose of similarity matching.

In the experiment, 42 images were used as query object and the result was a set of images with partial similarity.

One of the important parameters to be considered is the choice of value of n. If the choice is good, the result would be represented more accurately.

### *2.2. Frequent k-n match problem*

For finding objects of full similarity, the concept of frequent k-n match was developed. This was same as the k-n match problem but the only difference was in the value of n.

Here, the value of n could be varied within a certain range so that one could find out for every value of n and certain value of k (attribute), the objects that appeared most frequently in the answer set.

An example from the paper [1] could explain this better-

Suppose we are looking for objects similar to an orange. The objects are all represented by its features including color (described by 1 attribute), shape (described by 2 attributes) and other characteristics. When we issue a k-1 match query, we may get a fire and a sun in the answer set. When we issue a k- match query, we may get a volley ball and a sun in the answer set. The sun appears in both answer sets while none of the volleyball or fire does, because the sun is more similar to the orange than the others, in both color and shape.

The frequent k-n match problem is implemented on five real datasets taken from UCI machine learning repository. The dimensionality of the dataset varies from 4-34. To measure the effectiveness of this problem, a class stripping technique is used. By this technique every class is stripped and different techniques are used for the different classes to measure the similarity.By running some 100 queries randomly the most frequently appearing objects are selected in the answer set.

### *2.3. Multi-query K-nearest neighbor*

Multi-query K-nearest neighboralgorithm finds the k nearest neighbors for a given set of query points based on the Multi Query distance of a data point to query point. The average accuracy rate of this algorithm is 91.3% and that of frequent K-n match algorithm, which is 90.8%.

### *2.4. Sorting Based Algorithm (SBA)*

Sorting Based Algorithm (SBA) is an algorithm that sorts elements in a specific order and scans the sorted lists and recognizes identical sub sequences and output the relevant pairs. Radix algorithm is used to sort all the sub-strings in an alphabetical order.

### *2.5. Generalized Suffix Tree Based Algorithm (GSTBA)*

Generalized Suffix Tree Based Algorithm (GSTBA) uses suffix trees for similarity search. Suffix Trees is a compressed prefix tree containing all the sub-strings of the given text. Each edge in suffix tree is labeled with a symbol. A string is associated with every string in the tree.

### *2.6. Locality Sensitive Hashing (LSH)*

In this approach similarity search is done based on hashing. The main idea is to hash points from the data so that the probability of collision is greater for objects that are close to those that are far apart.

In locality sensitive hashing, the key approach is to hash the points using different hash functions, so that, for each function the probability of collision is higher for points closer to each other. The probability of collision is closely related to the distance between the points.

This method was implemented on two datasets-

- First one contains 20,000 histograms of colored images from COREL draw library.
- Second one contains 270,000 points of dimension 60 representing texture information of blocks of large aerial photographs.

The performance is measured on two parameters- speed and accuracy. The performance of LSH is measured by computing the minimal number of Indices needed to achieve a specified value of error.

- The info mentioned above can be illustrated in the tabular form as shown in Table 1 -

Table-1 Studied algorithms

| | **Title** | **Problem** | **Solution Algorithm** | **Result** | **Data Sets** |
|---|---|---|---|---|---|
| • | [1]Similarity search: A matching based approach | Partial Similarity | k-n match problem | Considers the matching between the query object and the data objects over 'n' dimensions | COIL 100 database that contains 100 images |
| • | [1]Similarity search: A matching based approach | Full similarity | Frequent k-n match problem | Finds out the set of objects which appear frequently in the result of a k-n match problem for 'n' values. | COIL 100 database that contains 100 images |
| • | [2]Similarity search approach to solve Multi query Problems | Nearest neighbor to multiple queries with possibly different degrees of importance | MQKNN ( Multi query k-nearest neighbor) | Calculate the difference between a given data point and a query point set, and design algorithms to efficiently retrieve K nearest data points to the query point set where, K is the number of data points required. | A dataset with normalized distributions from a synthetic data generator. |
| • | [3]Efficient algorithms for Similarity search | To identify substrings from each subsequence s. | SBA (Sorting Based Algorithm) | Sorts all the substrings in alphabetical order (so that similar subsequence comes closer) and scans through the sorted list to find similar subsequence and output the relevant pairs. | Set of string |
| • | [3]Efficient algorithms for Similarity search | String matching problem in sequence and in parallel | GSTBA (Generalized Suffix Tree Based Algorithm) | Identifying similar pairs of sequences in a given database | Data set of 2000 string with maximum length of string were 1000 and 2000 |
| • | [4]Similarity search in high dimensions via hashing | Similarity search in high dimension | LSH(locality sensitive hashing) | Higher running time than other searching algorithm by hashing points from the database so as to ensure a higher collision of similar objects | A histogram of 20,000 images from COREL |

Table 1 shows the various algorithms that have been used for similarity searching. These algorithms are tested on datasets and the results are verified. The table can be considered as a summarization of the different algorithms.

## III. RELATED WORK

This comparative result can be used to analyze the efficiency of the algorithms on various parameters like accuracy, response time etc. A paper can be presented that might contain a graphical as well as theoretical comparison of the result of the algorithms tested on a dataset.

## IV. CONCLUSION

The above table shows the various algorithms that have been developed for the purpose of similarity search. This table classifies the algorithm along with the problem for which they were developed and also the datasets on which they have been implemented.

## REFERENCES

[1] Anthony K.H.T., Rui Z., Nick K., & Beng C.O. "Similarity search: A Matching Based Approach" In VLDB, ACM, 2006.
[2] Yong S. "Similarity Search Approach To Solve Multi Query Problems", In
[3] S. Rajsekaran, Y.Hu, J. Luo, H. Nick, P.M. Pardalos, S. Sahani & G. Shaw " Efficient Algorithms For Similarity Search", In , pp. 1-10.
[4] Aristides G., Piotr I. & Rajeev M. "Similarity Search In High Dimensions via Hashing", In VLDB, pp. 520-529, 1999.

.