Identify Similar Research Papers Using Locality Sensitive Hashing

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Software Engineering

Submitted By
Divya Gupta
(Roll No. 801431007)

Under the supervision of
Dr. Shalini Batra
Assistant Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147001

June 2016

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "***Identify Similar research Papers Using Locality Sensitive Hashing***", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Software Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of "***Dr Shalini Batra***" and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

(Divya Gupta)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Dr Shalini Batra)

Assistant Professor
Computer Science and Engineering Department

Countersigned by

(Dr Deepak Garg)

Head

Computer Science and Engineering Department

Thapar University

Patiala

(Dr S.S.Bhatia)

Dean (Academic Affairs)

Thapar University

Patiala

Acknowledgement

No volume of words is enough to express my gratitude towards my guide, **Dr. Shalini Batra,** Assistant Professor, Computer Science and Engineering Department, Thapar University, Patiala, who has been very concerned and has aided for all the materials essential for the preparation of this thesis report. She has helped me to explore this vast field in an organized manner and provided me with all the ideas on how to work towards a research oriented venture.

I am also thankful to **Dr. Deepak Garg**, Head of Computer Science and Engineering Department and **Mrs Rupali Bhardwaj**, P.G. Coordinator, for the motivation and inspiration that triggered me for the thesis work.

I would also like to thank the staff members and my colleagues who were always there in the need of the hour and provided with all the help and facilities, which I required, for the completion of my thesis work.

Most importantly, I would like to thank my parents, friends and the almighty for showing me the right direction out of the blue, to help me stay calm in the oddest of the times and keep moving even at times when there was no hope.

Divya Gupta

(801431007)

Abstract

Identifying the research papers of a particular domain is a tedious and time consuming job for an academician and a researcher. Lot of e ort can be saved if all papers related to a particular domain can be combined in a single group. It will not be feasible to manually cluster the similar type of papers on the basis of topics, key words or abstract. This thesis presents an approach of clustering similar type of papers using Locality Sensitive Hashing (LSH) , a probabilistic data structure which adds similar type of documents in a single bucket by spiting the input text into shingles and using min-hashing, a variant of Jaccard similarity to generate signature matrix. Our work explores how similar research papers can be clustered by considering the title of the paper, keywords and abstract of the paper. Experimental analysis shows that using LSH majority of the papers of similar domain are categorized into one bucket in less time.

In particular, we interpolate the sensitive hashing for the abstract with authors, keyword and journal of the paper. The basic methodology we adapt is to turning of a document into vector model is done by shingling and homogeneity among sets, is intended using Jaccard similarity. By penetrating shingles we build Characteristic matrix, which engender signatures for each document by a technique called ”minhashing” is used to diminish the size of the matrix.

Table Of Contents

List of Figures

List of Tables

List of Abbreviations

NLP	Natural language Processing
NNSS	Nearest neighbor Similarity Search
BMS	Best Match Similarity
ANN	Approximate Nearest Neighbor
APSS	All Pair Similarity Search
BoW	Bag of Words
CM	Characteristic Matrix
D	Document
DM	Distance Measure
DS	Document Set
J-DIS	Jaccard Distance
J-SIM	Jaccard Similarity
LCS	Longest Common Subsequence
LSH	Locality Sensitive Hashing
S	Shingle
SM	Signature Matrix
SN	Shingle Number

Chapter 1

Introduction

Searching in an era where data is present in form of large information repositories becomes increasingly important, where the objects contained do not own any defined order, for example huge collections of images, audios and other sophisticated digital objects. Similarity Search is the most general phrase used for a span of procedures which share the principle of finding (big data) spaces of entities where doing comparison between any pair of objects is the only available method similarity search.

Each database maintains different outlook of an entity, and resolving entities based on particular attributes is a major issue. Retrieving useful information from research papers datasets is time consuming and identifying similar type of data items has become an important task in wide areas of application including pattern recognition, data mining, ip aliasing, *etc*. In similarity search problem, objects (e.g., documents, images) are collected from various data sources and one is required to find the nearest (most similar) object corresponding the user's query. Different algorithms and approaches containing methods based on rules, pair-wise classification, clustering approaches, *etc.*, which were earlier used for similarity search are not applicable to massive data sets. Due to the exponentially growing number of published research results, searching for pertinent papers can become a very difficult task for researchers. Several solutions have been presented by academia and research community such as recommender systems [1] or dedicated search engines such as Google Scholar, *etc*. but still there is lot of scope for improvement.

In the world of technology, data on the browsers is increasing exponentially and because of this finding similar research papers from the set of large collections of papers becomes a very tedious task for research scholar students, scientists, professors. We need an approach which aids to solve this problem efficiently

1.1 Approaches to Similarity Search

Today everything we see, read, write, hear and measure is present in the form of digital format. Various approaches are available for finding similarity between objects. Some of the frequently used approaches them are:

1.1.1 Metric space

The metric space approach focus on searching the coherent ways for locating user-relevant information in large number of objects, where the similarity between objects is quantified by using a pair wise distance measures. In this approach data is collected in domains and then distance functions, also known as metric functions, are applied on the data. The metric space has an advantage that no specific conditions are required for representation of data. Different metric space postulates are used such as non-negativity, symmetry, triangle inequality, identity, positiveness between the pair of objects according to parameters and properties [2].

1.1.1 Vector Spaces

In vector spaces, high dimensional data represents the similarity between data and can be evaluated by measuring the distances [3] between the two vectors using Euclidean distance, Jaccard distance, *etc.* dependent on their properties and functions. The applications where one performs similarity search represent data as high dimensional vectors. In vector space model, text documents are represented in the form of algebraic model where vectors are taken as identifiers such as index terms. It helps in filtering of information, information retrieving and indexing of the documents. Similarity between any two sets of texts or images is done by comparing the features extracted by them, for example, color histogram, keywords, single phrases, an image feature can be represented as a vector

1.2 Types of similarity

Basically similarity refers to psychological closeness or presence of two physical representations. Similarity exists in many different forms, some are content based and

some are context based forms. Various similarity types are:

1.2.1 Lexical Similarity

In linguistics, lexical similarity can be defined as the measure of degree or amount of similarity between two given languages. The lexical similarity of 1 (or 100%) means the universal identity among vocabularies and lexical similarity of 0 shows that there are no common words [4]. It depends on the variations in word list of the documents and it can only be symmetrical.

1.2.2 Semantic similarity

Given two sentences, the similarity measurement determines how identical the meaning or definition of two sentences is. The higher the semantic score the more similar the semantic of the sentences. For finding semantic similarity, each sentence is first partitioned into a list of tokens and then words are stemmed. Finding the proper meaning for every word in a sentence is known as Word Sense Disambiguation [5]. Finally, likeness of the sentences is computed on the basis of similarity of the set of words. For example, the concepts are characterized as the nodes of a DAG in a semi-ordered set. Semantic relation between words or set of words can be depicted with the help of various processes such as a vector space model.

1.2 Importance of Similarity

Identifying required information from the massive data sets is a tedious task. Similarity search is required for classify different things and entities in the real world, for selecting the required option good recommender systems are required. Some of the important areas of application of similarity search are:

1.2.1 Classification

Similarity-based classification is a boon for problems in bioinformatics, computer vision, retrieval of information, NLP, and a wide range of other fields. The functions which calculate similarity may be asymmetric. To resolve this problem several approaches are used. A popular approach to find similarity-based classification [6] is to treat the given

similarities as inner products and treat dissimilarities as intervals in some Euclidean space.

1.3.2 Plagiarism

Plagiarism is defined as the "unfair appropriation" and "thieving and paperback" of another author's language, ideas, thoughts and the representation of them as one's own genuine work. Plagiarism is process which is mostly used by professional institutes to maintain the authentication of data. It occurs when an individual defer or presents the verbal or written work of different person as his or her own language.

1.3.3 Recommendation Systems

Recommender System [7] is defined as a subclass of filtering of information and predicting the choice or preference of different users on different items. Many large-scale commercial and popularized social websites recommends the products and people to connect to each other according to predicted choices and similarity in ratings. For recommending good items or predicting the choice of user, finding similarity among the products is important. For example, to find research papers of same domain we need to find similarity among them. This aids in the field of education and research analysis.

1.4 Variations of Similarity

A similarity search problem contain two elements: an environment E that defines the space for search, the distance function, and the correspondence function to use, i.e. E = {D; L : δ(p; q); h}, and a query Q that defines data points which are to be qualify in D. If the relationship between distances and scores is given, one can resolve similarity search problems by only contemplate the distances. Once the distance is found, the scores are only evaluated for the objects in the set of answer.

1.4.1 Range Similarity Search

In range similarity search, main purpose is to collect all those points of data that resides in the neighborhood of a query point. The closeness is thereby defined with a radius r. A drawback of the range similarity search is that it cannot give a clear picture of how to

choose value of r. If value of r is negligible or very less, the answer set may return no elements. If r is too large, objects are returned.

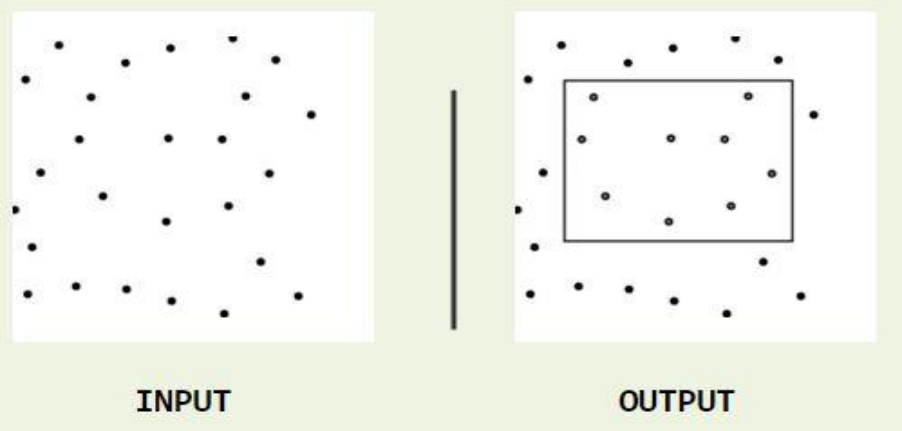


Figure 1.1: Range Similarity Search

1.4.2 Nearest Neighbor Similarity Search

Nearest Neighbor Search limits the size of the answer set, but not the distances/scores of the objects. This query follows from the relationship between scores and distances. The best values for k are selected [9], the best ones which have a good score with respect to the query of all other objects are given as results. But the disadvantage of this query is that it presents the results to the user regardless of the query.

Figure 1.2 : Example of Nearest neighbor Similarity Search

1.4.3 Best Match Similarity Search

Best Match Similarity Search is a junction of NN-search and range search which reduces the above problem where the user can restricts both the magnitude of the answer and the quality of the value of the returned objects. Hence, the resulting set is full with at most the k best matches.

1.4.4 All pair similarity search problem

In all pairs similarity search problem [10] the problem is to search all the sets of items whose similarity values are above the predefined value. Today, in data sciences many applications like recommender systems, search engines and digital libraries require to measure this problem of databases as these applications have large sample of presentations in a high dimensional space, that are actually scarce.

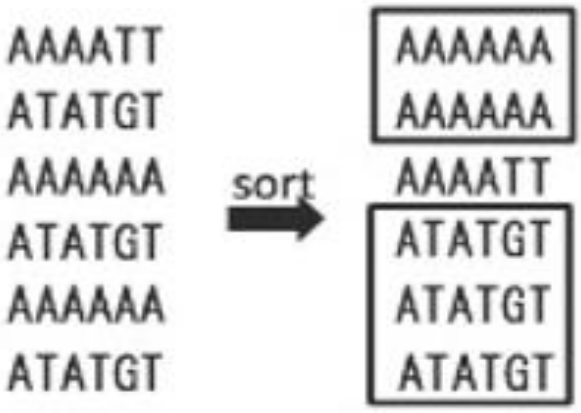


Figure 1.3: Example of All Pair Similarity Search

1.5 Distance Measures

Given a collection of data points in the form of sets, called as space, a function d(x1, y1) that takes two points as arguments in the space and produces a real number as output, is called as a distance measure[11] and satisfies the following preconditions:

1. d(x1, y1) $\geq$ 0 (non- negative distances)
2. d(x1, y1) = 0 if and only if x1 = y1 (distances are positive, except for the distance from a point to itself).
3. d(x1, y1) = d(y1, x1) (if distance is symmetric).

4. d(x1, y1) ≤ d(x1, z1) + d(z1, y1) (the triangle inequality property describes the length of shortest path from one point to another).

Various distance measures are used to observe and evaluate the nearness of the points in some given space. Some of the commonly used distance measures are given below.

1.5.1 Jaccard Distance

The division between the intersection and the union of the pairwise variables between two objects is called Jaccard similarity [11]; used to calculate similarity score between two sets is a common index for binary variables

The Jaccard distance can also be defined as a distance which measures *dis*similarity between two sample sets *i.e.* (A,B) is complementary to the Jaccard coefficient and is calculated by subtracting the Jaccard coefficient from 1 as given in Eq.(1) or by dividing the difference of the modulus of the union and the intersection of two sets by the size of the union:

$$Dist(A,B) = 1 - \frac{|A \bigcap B|}{|A \bigcup B|} = \frac{|A \bigcup B| - |A \bigcap B|}{|A \bigcup B|} \quad (1)$$

Different constraints of a distance measure are:

- If A and B are both empty, J(A,B) = 1

$$\mathbf{0 \leq J(A,B) \leq 1}$$

- The cardinality of union and intersection of two given sets can never be identical at the same time except the case when both union and intersection are same.
- Jaccard Distance is strictly positive.
- The results come after the union and intersection of two sets are always symmetric.
- Jaccard Similarity always follows triangular inequality, so Jaccard Distance also satisfies it.

1.5.2 Euclidean Distances

It is one of the most commonly used distance measure. A n-dimensional Euclidean space is considered where points are represented in the form of vectors of n real numbers. Then the Euclidean distance [12] between any two points x and y, d (x, y), is:

$$d([x_1, x_2, \ldots, x_n],\ [y_1, y_2, \ldots, y_n]) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \qquad (2)$$

Constraints on Euclidean distance measure are:

- The Euclidean distance between any two points can never be negative, because there is a positive square root in the distance equation
- If xi = yi for all i, then the distance is zero.
- Symmetric relation follows because of $(x_i - y_i)^2 = (y_i - x_i)^2$
- The triangle inequality requires a good deal of algebra to verify.

Other distance measures are also there which have been used for Euclidean spaces. For any constant value of r, we can interpret the L_r-norm to be the distance measure as given in eq(3):

$$d([x_1, x_2, \ldots, x_n],\ [y_1, y_2, \ldots, y_n]) = (\sum_{i=1}^{n} |x_i - y_i|^r)^{1/r} \qquad (3)$$

1.5.3 Hamming Distance

The Hamming Distance is a count used to present the difference between two strings represented in the form of binary numbers. The Hamming Distance is used for the Boolean vectors i.e. which contain only ‘0’s or ‘1’s. It is a small package with wide range of formulas used in information analysis. Basically, Hamming's formulas [13] permits computers to detect and correct errors on their own. The number of objects in which the two items distinguish is the hamming distance between them. Important properties of hamming distance are:

- The Hamming distance can never be negative.
- Hamming distance is 0, if and only if the vectors are same.

- The distance which is observed as a result does not depend on the vectors
- Hamming distance is symmetrical.

The triangle inequality is also followed because if a is the number of elements in which x and y vary, and b is the amount of component in which y and z differ, then x and z cannot be different in more than (a +b) components.

1.5.4 Cosine Distance

This metric is commonly used when one wants to calculate similarity between two documents. Since it might be plausible that uncommon words exist in the random documents in large amount, it is not fruitful to use the other methods of observing similarities (namely the Euclidean Distance and the hamming distance discussed earlier).The possibility that two documents do not split the maximum part is very high and does not give a wanted measure for determining similarities. In Cosine distance [14], normalized dot product of the two documents is calculated where the attributes (words in the documents or pixels of image) is considered as a vector. By calculating the cosine similarity, the user wants to observe the cosine of the angle between the two objects (x, y):

$$similarity(x, y) = cos(\theta) = \frac{x \cdot y}{||x|| * ||y||} \quad (4)$$

Cosine distance is calculated by subtracting the similarity from 1 as in eq(5)

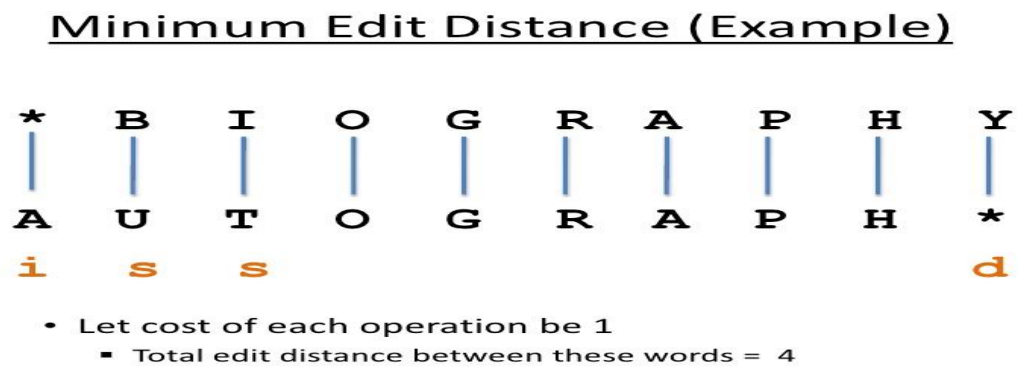
$$\text{Cos Dis } (x, y) = 1 - \text{Similarity}(x, y) \quad (5)$$

- As the values are taken in the range of (0,180), so distance can never be negative.
- If the two vectors are in the same direction means they do not share any attributes (or words) the angle between two vectors is 0.
- When the angle between the objects is 0 degrees the cosine similarity is 1
- Vectors in cosine distance follows symmetric relation
- Cosine distance also obeys triangle inequality.

1.5.5 Edit Distance

Edit distance is used when the data points are stored in the form of strings. Suppose there are two strings p =p1,p2,p3……p_n and q=q1,q2,q3…….q_m ; distance between them is counted as the least number of times the pushing and popping of a single characters is done to convert p to q or q to p . One more way to define edit distance is to compute LCS of p and q i…e…LCS (P, Q) then edit distance is defined as:

LENGTH (P) +LENGTH (Q) – 2* LENGTH (LCS(P,Q)) (6)



1.6 Structure of the Thesis

The remaining part of the thesis is organized in the following order:

- **Chapter 2** gives the details of related work or literature review in the corresponding field.
- **Chapter 3** presents the problem statement, gap analysis between the existing work with the desired targeted work, a concise statement of the problem.
- **Chapter 4** defines the methodology in detail and why particular technique is used
- **Chapter 5** shows the design techniques and all the steps which tells how to solve the defined problem
- **Chapter 6** provides experiments and implementation of the proposed approach.
- **Chapter 7** concludes the thesis and provides future scope of the work

Chapter 2

Literature Review

Finding similar research papers is a very time consuming process. If the data is present in less dimensional form then the nearest neighbor search algorithm give better results but as the world is scaling to big data, the data is growing enormously. With the growing data, handling big data computations have become an important issue. Much research has been done to find similarity among objects within defined time and space.

In [16] Cornelis explores a technique called as language modeling techniques to find similarity in big data. The methodology followed is to collect the data contained in the abstract by constructing new data points within the range with language models for the keywords and journal with the author of the paper. This strategy is further improved by finding titles and additionally interpolating with the results of models. Latent Dirichlet Allocation, an adaptation technique in which the keywords provided by the authors are used to guide the process.

In [17] Simpson presented a method for finding identical words with syntactic similarity of two strings. An algorithm based on data present in dictionary is used to capture the semantic similarity between two sentences known as dictionary based algorithm, which is strongly dependent on the WordNet semantic dictionary. A trustable score that provides the semantic dependencies between the definitions of two sentences is semantic similarity. It is tedious to acquire a better accuracy results because the correct semantic meanings are completely defined only in a respective context.

In [18] Tiridur proposed approach based on association similarity and uses Hierarchical agglomerative algorithm for making the cluster of documents. By using Correlation similarity, the cluster of every document is created and similarity between the documents is calculated. In HAC algorithm, the cluster is grouped level by level and to give rank to the cluster ranking technique is followed. Depending on the data of the document the

cluster evaluations are presented by grouping the relevant data.

In [19] goama presents an approach of finding similarity between text by dividing them into three approaches; String-based, Corpus-based and Knowledge-based similarities. The similarity measures between words, sentences, paragraphs and documents is a significant element in various works such as automatic essay, translation of machine summarization of text, information retrieval, clustering of documents.

In [20] Milenko a blocking approach is used to deal with big data where entities are compared pairwise with 1,000 hospitals from 100 cities. By using blocking technique some si4milar items are missing, the solution is to apply a blocking algorithm that reduces the difference of the set of identical pair data and those fulfill the criteria in blocking, and increase the intersection. Algorithms based on hash are used. Another powerful technique is MinHash or locality sensitive hashing, which uses measures of distance.

In [21] Romes gives Canopy Clustering, along with two values of thresholds and with a distance metric is chosen. By picking a random mention, a canopy is created using the first value of threshold and all the discussed issues are removed from where the centroid distance is very less than the second value of threshold. This approach is creative because in more than one canopy mentions can be included, and therefore minimize the opportunity that a perfect match can be excluded by approach of blocking.

In [22] Kołcz presents a different approach for finding a document of a particular domain by using novel near-duplicate method where each document is introduced as a real-valued sparse k-gram vector, different values of weights are used to optimize the range of similarity functions such as the Jaccard coefficient or cosine similarity In addition to this, vectors are linked to less number of hash-values as signatures of documents by using the locality sensitive hashing scheme for better similarity computation.

.

In [24] Schnitzler presents a filtered and refined methodology to increase the speed of searches in nearest neighbor with the Kullback–Leibler divergence for multivariate Gaussians. This is very creative and different approach where combination of features and similarity estimation is used in the field of automatic recommendation of music to predict the similar types of music.

In [25] Josephson suggest that although LSH is an efficient approach for finding similarity but still it cannot be considered practically because it's search quality vary according to several parameters which are dependent on data. So tuning of parameters and observing their performance is preferred by him. To resolve this, a statically model of Multi-Probe LSH is used to predict the average quality of search and latency for a given sample of datasets.

In [26] Cowell proposed a method to ensure the amount of the memory used when LSH is applied on different videos, image systems and text documents. Bin splitting is the approach which is used with the design of bands of Locality Sensitive Hashing. It ensures that it is used by considering only 0.04% of the occupied bins and try to lessen the amount of recall that increases when bin splitting is used.

Lee in [27] put forward an algorithm to decrease the number of false negatives in local sensitive hashing. The items are permitted to be hashed in large number of buckets if the item appears at the farthest boundary of the bucket. The algorithm produces a better accuracy but it takes high computation cost and memory space.

Chapter 3

Problem Statement

3.1 Gap Analysis

With the growing volume of data and increasing number of users on the internet, searching of required entity or a particular thing is become a tedious task. In today's scenario finding homogeneous items and patterns is one of the most deliberated problems in wide range of fields of computer science like machine learning, data mining and pattern recognition. Some of the areas which need improvements are:

- **Linear search:** Database may contain objects in terabytes and each object described by a vector that contains hundreds of dimensions. Therefore, a solution is desired that does not depend on a linear search of the database. Data is scattered on the browsers in the form of structured, unstructured or semi-structured data. Finding the resulting query from big data is becoming a very time consuming process because linear search is done on the scattered data and partioning of the data is performed if the data is sorted.
- **Accuracy**: Partioning algorithms works well only for less dimensional data but as the data increases exponentially, the efficiency and accuracy of these algorithms decreases.
- **Space complexity and time complexity**: needs to be reduced in case of massive data sets. To resolve this problem one option is to uses Locality Sensitive Hashing (LSH) which is a probabilistic data structure and has an important property that "distance is maintained for similar items

3.2 Problem Statement

By examine the above stated problems and research gaps, we need a technique that maintains the accuracy of results and reduce both time and space complexity. Locality sensitive hashing reduces the data points from a high-dimensional space to a low-dimensional subspace. It produces projections from a number of unlike directions and keeps record of the nearest points.

3.3 Objectives

- To study various similarity search techniques available in the literature.
- To find similarity between research papers using shingling approach and Locality Sensitive Hashing.
- To test and validate the proposed scheme for efficient and accurate search by varying various parameters.

3.4 Methodology

Main focus of the thesis is to improve accuracy and time complexity of similarity search for high dimensional data. Methodology followed is:

- Shingling is used to convert the data into form of vectors
- Minhash algorithm has been applied for 500 research papers. Shingles sizes for the text files which contain data are taken as k=9 as for large documents is there.
- Locality Sensitive Hashing is used for finding exact matching pairs so that research papers of same domain come in the same bucket.

Chapter 4

NNS (Locality Sensitive Hashing)

4.1 Nearest Neighbor Search

Nearest neighbor search (NNS), also called proximity search or closest point search is useful for finding similar kind of objects in database. It is an optimization problem to search the closest neighbors in high-dimensional data as shown in figure 4.1. In NNS:

- "Input is a set A of p points in a D-dimensional metric space M = (B, d) and a query point Q ∈ B, find the point p ∈ P closest to Q" [28].
- M is considered to be a *d*-dimensional Euclidean space and distance is measured by Manhattan distance or Euclidean distance

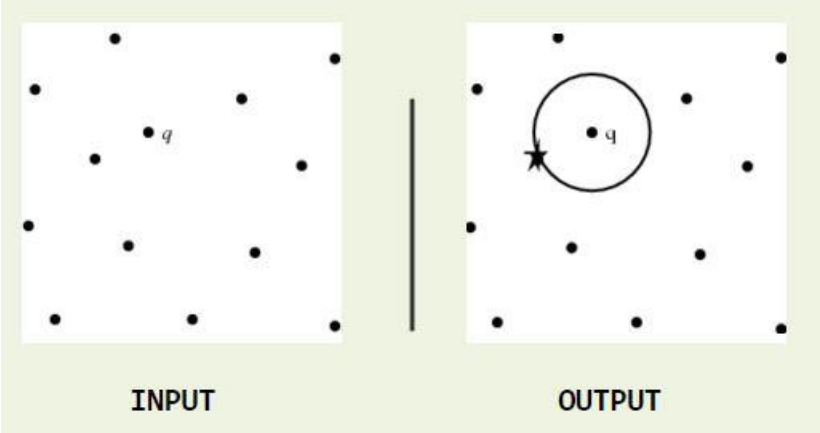


Figure 4.1: Example of Nearest Neighbor Search

4.2 Applications of NNS

- Pattern recognition (particularly for character recognition)
- Cluster analysis
- Computer vision - e.g. content-based image retrieval
- Duplicate detection: Web page, Image retrieval
- Plagiarism detection

- Databases
- Coding theory
- Data compression
- Internet marketing (like contextual advertisement)
- DNA sequencing
- Spell checking(suggesting correct spelling)

4.2.1 Duplicate Detection

Detection of duplicate images and web pages is one of the major applications of Nearest Neighbor search. Many MNC's and large commercial websites detect duplicity or similarity for security purpose. Similarly, finding similarity between *research papers* is becoming a difficult task for students or beginners and scientists who try to explore different research ideas in various fields. As research in various fields is increasing more and more research papers are added on the search engines. Various methods of Nearest Neighbor search try to solve this problem. But among all the methods of NNS, Locality Sensitive Hashing is one of the hashing based technique which provides the accurate results in less time. Such applications can help students to refine their research and help them in identifying the research papers of their domains easily and efficiently.

4.3 Variant of NN Problems

Nearest neighbor search algorithm finds the closest point according to the query in different ways. Some of the ways are explained below:

4.3.1 k-Nearest Neighbor Search

- It is a non –parametric algorithm used for regression and classification.
- In k-NN classification the output is in the form of class membership. If k=1 then object is given to the single closest member of class [29].
- In k-NN regression the output is the value of the property of the object [30].
- Disadvantage of this algorithm is it is quite sensitive to the structure of local data.

4.3.2 Approximate Nearest Neighbor Search

Definition: "Given a query point q, there exist a point p such that || q - p || ≤ r where p returns p' such that || q – p' || ≤ (1 + €) r.

- ANN is used in applications where accuracy of results is not as important as the speed. Approximate Nearest Neighbor (ANN)[31] is a searching methodology which reduces the amount of accuracy.
- It can be used to resolve the exact nearest neighbor issue by choosing all approximate closest neighbors and selecting the nearest point.
- This methodology does not guarantee to return the actual nearest neighbor in every case although it is advantageous in terms of speed or memory.

4.5 Methods of NNS

There are different methods for solving the various problems in Nearest Neighbor Search. All of them have different complexity in terms of space and time. Some of them are explained below:

4.5.1 Linear Search

The running time complexity of linear search is O(Nd) where N is the total number of elements in set S and d is the dimensionality of metric space M.

4.5.2 Space Partitioning

- In space partitioning, the space (Euclidean space) is divided into disjoint sets (partition of sets); this approach is also known as spatial access or spatial index methods.
- KD-Tree [32], BSP trees are the data structures used to store the objects in space partitioning in case of geometric series.
- Suffer from either space or query time complexity that is exponential in d.

4.5.3 Hashing Based

- Locality-Sensitive Hashing
- Spectral Hashing

4.6 Locality Sensitive Hashing

The term ‘locality-sensitive hashing’(LSH) was introduced in 1998, it is a name given to randomized hashing framework for efficient Approximate Nearest Neighbor (ANN)[33] used to search similar items in huge amount of data. In this technique, a family of functions called MinHash, are combined with banding technique to eliminate the disjoint pairs at minimum distance from the pairs at maximum distance [34]. Min hashing convert large documents into a matrix of small signatures and give the expected similarity of any record pairs and LSH finds identical pairs efficiently by adding the pairs which are most plausibly similar in the same bucket thus avoiding identification of each and every combination of documents.

In case of general hashing, exact matched or closed (near) items are hashed plausible in different locations according to linear probing, quadratic probing *etc*., but in Locality Sensitive Hashing similar items maintain their nearness even after hashing (mapping), as shown in figure 4.2

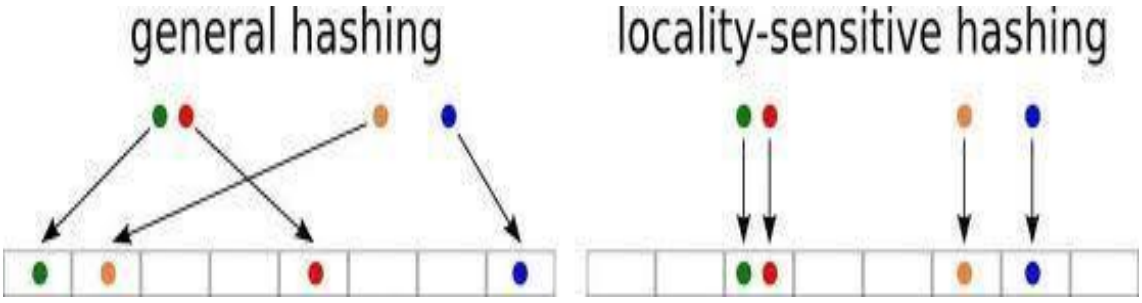


Figure 4.2: Difference between general hashing and locality sensitive hashing [35]

In LSH, a technique which divides the entire matrix into equal bands, known as banding technique, is used to find similar elements. Once the MinHash signature matrix is generated, matrix is divided into b bands where each band contains equal number of

rows. Further every band is divided into fixed number of buckets and documents are added to different buckets using universal hashing technique [36]. The documents which are added in the same buckets are called candidate pairs. Candidate pairs are those which hash at least one time to the similar bucket. A threshold value t (t<1) is picked to compare identity of two sets of points. If at least t fraction of rows contains same signatures, a pair of documents considered to be similar. Once the candidate pairs are generated, comparison is performed between them only instead of comparing every pair as in other similarity search techniques shown in figure 4.3. One has to inspect first in main memory that candidate pairs have similar signatures [37] or not and based on that investigation, closest record of pairs is found and the pairs that do not have same signatures are discarded. The observations and correctness of similarity is given by a threshold value. Majority of approximate similarity search techniques use techniques like LSH [38] depends on the domain of the similarity search.

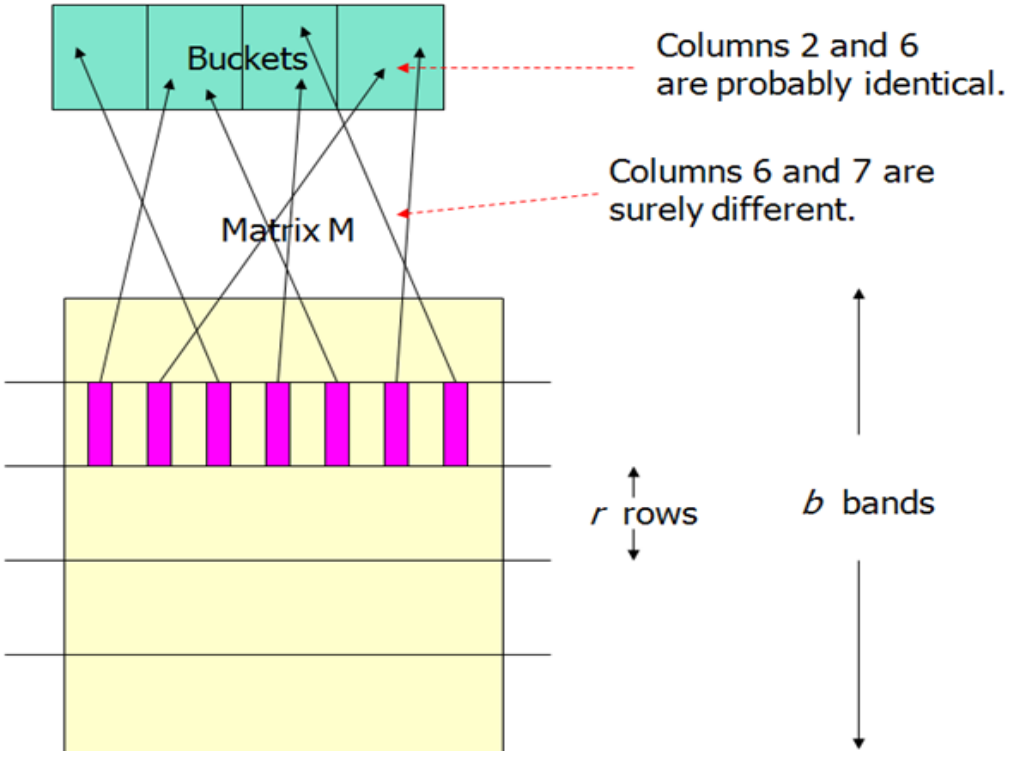


Figure 4.3: Candidate pairs in same buckets in LS

4.6.1 Bucket Hashing

Methodology followed for Bucket hashing is:

- Initially buckets which are empty are stored in figure 4.4
- The buckets $gi(v)$=$x1,\ldots,xk$ includes h2($x1,\ldots,xk$)
- The count of buckets should be same for all bands.
- Bucket size can vary according to the output

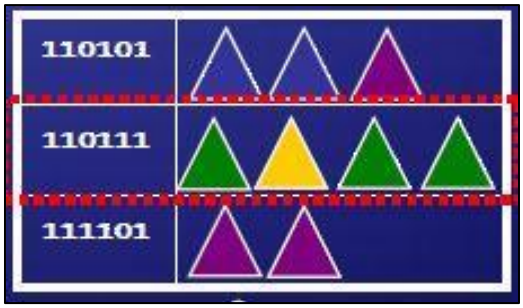


Figure 4.4 Hashing of buckets

One of the shortcomings of LSH technique is that there is a possibility of generation of false positive and false negative. In LSH, false positives[39] are dissimilar pairs which are hashed to the same bucket although they are dissimilar and similar pairs which are not dispatched to the same bucket despite being similar are false negatives. It means that the false positives are pairs which are mistakenly considered as a candidate pair and the pairs which are mistakenly not considered as a candidate pair are false negatives

Suppose d1 and d2 are parameters of distance where (d1< d2) according to one of the distance measurements d. F is called (p1, p2, d1, d2)-sensitive if for all f belongs to F;

Probability of f(y) =f(x) is

$$\begin{array}{ll} at\ least\ p1 & if\ d(x,y) < d1 \\ at\ most\ p2 & if\ d(x,y) > d2 \end{array}$$

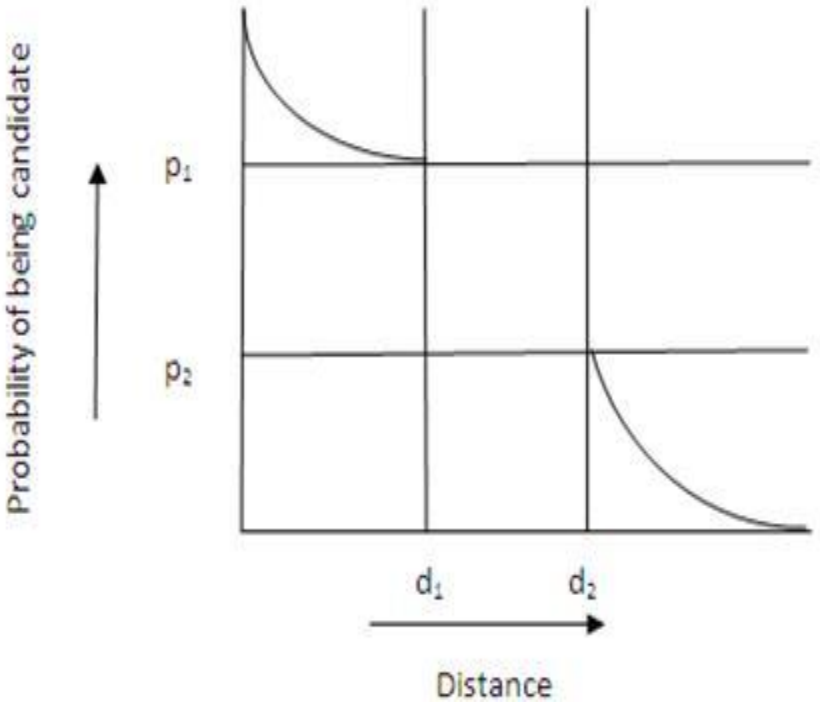


Figure 4.5: Graph showing higher probability of similarity when distance is small

Figure 4.5 presents a graph where x-axis shows the distance between two pairs and y-axis shows the probability of two candidate pairs.

Chapter 5

Design Techniques & Implementation

5.1 Sets

A set is a finite or infinite (unordered) collection of objects where order of objects has no significance and multiplicity is usually ignored like in multisets or list. The members of sets are called elements and studying their properties is known as set theory.

Set is presented in two forms:

- **Roaster form or tabular method :** elements are represented within the pair of brackets and are separated by comma
- **Set Builder form or Rules:** Set is defined within the rules, statement and formulas.

5.2 Operations

Union of two sets A and B is presented by A ∪ B. The union of elements defines when inserting all the elements in one set without repetition. It follows symmetry property.

Intersection of two sets is denoted A ∩ B. The result comes out as the grouping of all common elements from two or more sets. Intersection satisfies associative property as well as symmetry property.

The *difference* of two sets A and B is denoted by A – B. It results all the elements of A eliminating the elements of B. The difference of sets can be abelian, non-abelian and cyclic.

Distance conventions between two objects:

- It is very less if two objects are very close,
- It is large when they are far from each other
- It is negligible when are at the same position

5.3 Jaccard Similarity

The quotient between the intersection and the union of the pairwise compared variables among two objects is called Jaccard similarity [4]; it is used to calculate similarity score between two sets is a common index for binary variables as shown in figure 5.1

$$\text{J-SIM}\ (A, B) = \frac{|A \cap B|}{|A \cup B|}$$

Figure 5.1 shows two sets A and B where set A contains 5 elements and set B contains 3 elements so in total there are 8 elements appear in both A and B

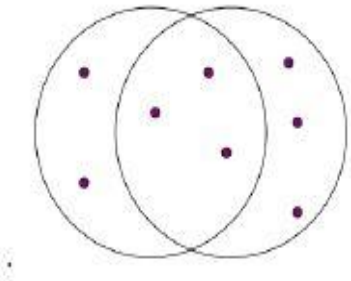


Figure 5.1: Jaccard similarity of two sets

J - SIM (A, B) = 3/8 = 0.37

5.4 Stop words

Stop Words are the set of words which do not contain crucial or necessary significance when used in search queries. Normally these words are eliminated or filtered out from the documents or any search queries because they give large amount of waste information. Example of some of the stop words are the, in, for, all, to, of, and, *etc*.

5.5 Documents Sets

The conversion of documents into measurable form is done by two commonly known techniques: **bag of words** model and **shingling**. In bag of words every record is considered in the form of unordered set of words. The technique of hashing substrings is referred to as "shingling", and each unique string is called a "shingle". In shingling technique, consecutive words of fixed size of a document are represented in the form of single object. One way to represent it would be to check document for all of its words, and present the document as it contains the clone of all distinctive word.. By regarding the documents as sets of words, we can estimate the overlapping of documents by using Jaccard similarity as shown in figure 5.2

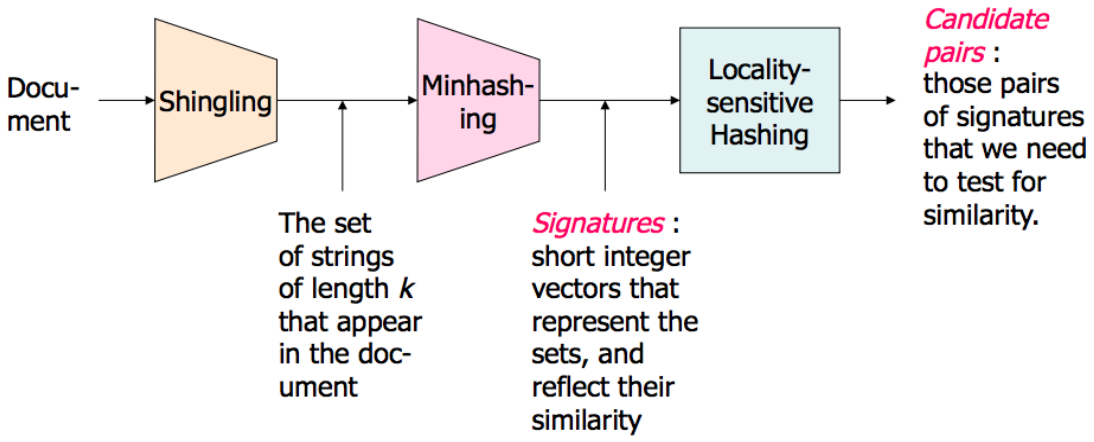


Figure 5.2 block diagram of design techniques

5.5.1 Shingling

To represent documents as sets, shingling is the most efficient way of finding lexically similar documents from the set of documents that contain short strings. If we do so, the documents which contain text information as short as phrases or sentences contain some common elements in their documents and some sentences presents in variable order in documents

- In shingling, k- shingles are used. Suppose we have a document which is a string of characters, than k-shingles make the substring of the length k

- For example, if the document D1 contains string {abcdab} and if k=2 than set of 2-shingles is {ab , bc, cd, da, ab}. Here substring ‘ab’ appears twice within the set, but considered only as a one shingle.

5.5.2 Properties

- The size of k varies according to documents; k=5 for emails and k=9 for research papers and for blogs ,news , articles, fingerprints data, *etc.* it varies from k=5 to 9
- In shingling, sets can also be depicted in the form of character level, where value of k depends on the characters not on the words
- It does not count the repetitions.
- It also removes the stop words
- It is also represented in different models like white space model used to check white spaces, capitalization used to identify proper nouns, punctuation for checking that the symbols are included in shingles or not

5.5.2 Bag of words

Bag of words is another technique used to present documents in the form of sets but in unordered collection of sets.

- It can be performed on image classification
- It count the replicas
- In BOW, weights are assigned to the words in the documents and according to the weights words are presented in the form of vectors
- It is used in many applications like e-mail filtering, image classification and natural language processing.

5.6 Characteristic Matrix

The main focus of this matrix is the hash functions which takes a 32-bit integer and make it to a different integer, with no collisions. If shingles are generated for large data set say for around a million of documents, restoring all the shingles in the main memory is not plausible. To reduce the amount of data to be stored *i.e.* for small representations of big set characteristic matrix [40], a matrix formed by checking the corresponding entries of shingles in the documents is formed. Characteristic matrix is not an efficient way of storing the shingles and presenting the data but it is very helpful for visualizing the data. These calculation can be considered as taking the union of the two sets, rearrange them in a random form, and choosing the first value in the new sorted sequence. To build a signature, one has to present collections of shingles in the form of a matrix which is termed as 'characteristic matrix' In the row r and column c of this matrix, the value '1' indicates that the item of the row r is present in the dataset of column c, else, a '0' value is marked if that element is not present in that particular row.

If four documents S1, S2, S3, S4 are considered, characteristic matrix for the shingle sets S1 = [a , f , g]; S2 = [a , b , c]; S3 = [a ,e ,f ,g] and S4 = [a, b, c, d, e]
and S5 = [c, d, e, f, g] the universal set [a, b, c, d, e, f, g] is given in Table 5.1 after hashing each shingle set acquires a bucket number, where {0, 1, 2, 3, 4, 5, 6} are bucket numbers respectively.

Table 5.1: Characteristic Matrix showing representation of document sets

Shingles	**S1**	**S2**	**S3**	**S4**	**S5**
0	1	1	1	1	0
1	0	1	0	1	0
2	0	1	0	1	1
3	0	0	0	1	1
4	0	0	1	1	1
5	1	0	1	0	1
6	1	0	1	0	1

After making characteristic matrix, hashing of shingles is done where hash values of different shingles is stored in matrix. The three hash functions are shown in Table 5.2, Hash1 ((sn+1) mod 7), Hash2 ((sn+2) mod 7) and Hash3 ((sn+3) mod 7), where sn denotes the corresponding number to the shingles.

Table 5.2: Characteristic matrix with hash functions

Shingles	S1	S2	S3	S4	S5	Hash1	Hash2	Hash3
0	1	1	1	1	0	1	2	3
1	0	1	0	1	0	2	3	4
2	0	1	0	1	1	3	4	5
3	0	0	0	1	1	4	5	6
4	0	0	1	1	1	5	6	0
5	1	0	1	0	1	6	0	1
6	1	0	1	0	1	0	1	2

5.7 Minhashing

Once characteristic matrix is generated, various permutations of rows are done and after permuting the rows first entry as ‘1’ of every column is stored and MinHash Signature matrix is generated as shown in Fig 5.3.

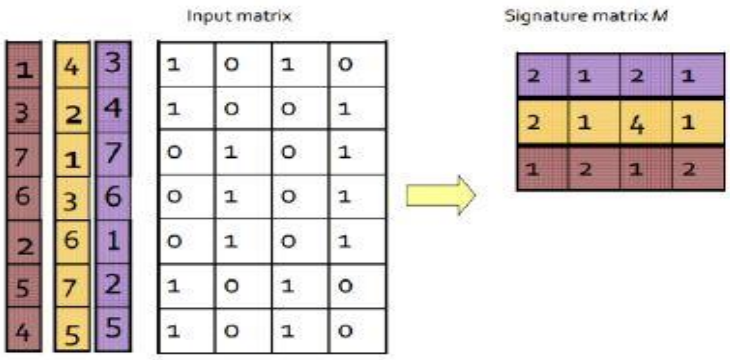


Figure 5.3 MinHash Matrix

5.8 Signature Matrix

Signature matrix (SM)[11] contain rows generated according to the hash functions and numbers of columns remain same like that in Characteristic matrix. Initially a large number value is stored in each column. The whole matrix is scrutinized row by row. The row which contains the entry as "1", for that equivalent or corresponding value of hashes is compared with already present hash functions in the signature matrix. If the value is less it is updated in SM, else no action is performed. Initial Signature matrix is shown in Table 5.3.

Table 5.3: Signature matrix

	S1	**S2**	**S3**	**S4**	**S5**
Hash1	∞	∞	∞	∞	∞
Hash2	∞	∞	∞	∞	∞
Hash3	∞	∞	∞	∞	∞

Table 5.4: Updated values after applying MinHash algorithm

	S1	**S2**	**S3**	**S4**	**S5**
Hash1	1	1	1	1	∞
Hash2	2	2	2	2	∞
Hash3	3	3	3	3	∞

In first row of characteristic matrix document S5 does not have value "1" and all other documents have "1".The hash values of all other four documents are matched with the corresponding hash values in the matrix. Same procedure is applied on all remaining rows (Table 5.5, 5.6, 5.7).

Table 5.5: Updates values of row2 and row 3 of CM

	S1	**S2**	**S3**	**S4**	**S5**
Hash1	1	1	1	1	3
Hash2	2	2	2	2	4
Hash3	3	3	3	3	5

Table 5.6: Updated values of row 4 in Characteristic Matrix

	S1	**S2**	**S3**	**S4**	**S5**
Hash1	1	1	1	1	3
Hash2	2	2	2	2	4
Hash3	3	3	0	0	0

Table 5.7: After scan of row 5 of characteristic matrix

	S1	**S2**	**S3**	**S4**	**S5**
Hash1	1	1	1	1	3
Hash2	0	2	0	2	0
Hash3	1	3	1	0	1

The final Signature matrix after scanning all the rows is shown in figure 5.8

Table 5.8: Signature matrix after scanning all rows of characteristic matrix

	S1	**S2**	**S3**	**S4**	**S5**
Hash1	0	1	0	1	0
Hash2	0	2	0	2	0
Hash3	1	3	1	0	1

Table 5.8 that signature of document S1 ,S2, and S5 are similar

5.9 LSH on Signature Matrix

Once signature matrix is generated LSH is applied on this matrix. As per LSH, initially rows of the signature matrix are divided into b bands of r rows such that if n is number of row n = b* r. For each band there is a hash function and this hash function may or may not be same for all the bands. Column vectors of the rows are hashed to the bucket according to the applied hash. Any pair of column vector that hash to the same bucket is called as candidate pairs as shown in Fig 5.4. The best results are achieved when unlike pairs do not come in the same bucket and since they are not part of same bucket such unlike pairs are never examined. For efficient results, pairs which are not same and are hashed to the same bucket (considered as false positives) should be less. The similar pairs must be assigned to the same bucket for at least one hash function

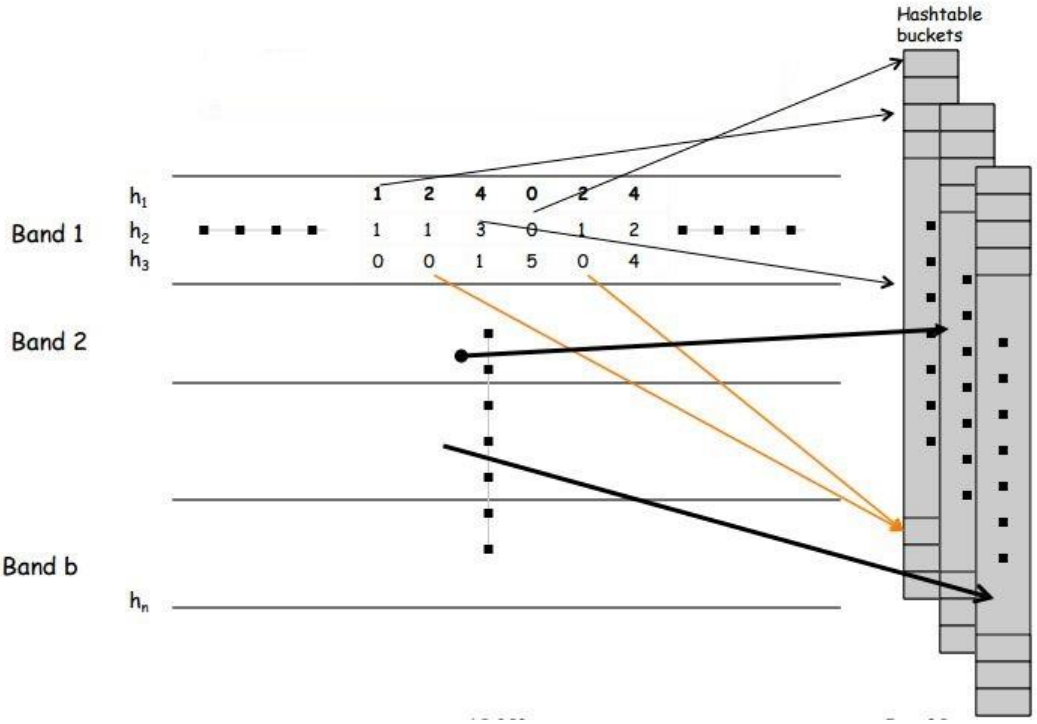


Figure 5.4 LSH on Signature Matrix

5.11 Analysis of the Banding Technique

Different possibilities of the items coming in the bucket are:

- The probability that the two entries are equal is **s.**
- The probability that the bands are equal *i.e.* each and every signature is matched in the band is $\mathbf{s^r}$.
- The probability the signatures or column vectors will not match in at least one row is $\mathbf{(1–s^r)}$.
- The probability that the none of the signature will match in any band is $\mathbf{(1 – s^r)^b}$.
- The probability that at least in one band signatures are equal in all the rows is $\mathbf{1 – (1 – s^r)^b}$.

With the selected values of b and r the similarity function generated in the form of S curve shown in the Figure 5.5 The value of threshold, denoted by similarity s where the probability of a pair becomes a candidate is 0.5.The jump at sharp point presents rough value of threshold for large values of b and r. At that point one observes that pairs which are likely to become candidates have similarity above the threshold while those which are not coming to same bucket are placed below the threshold.

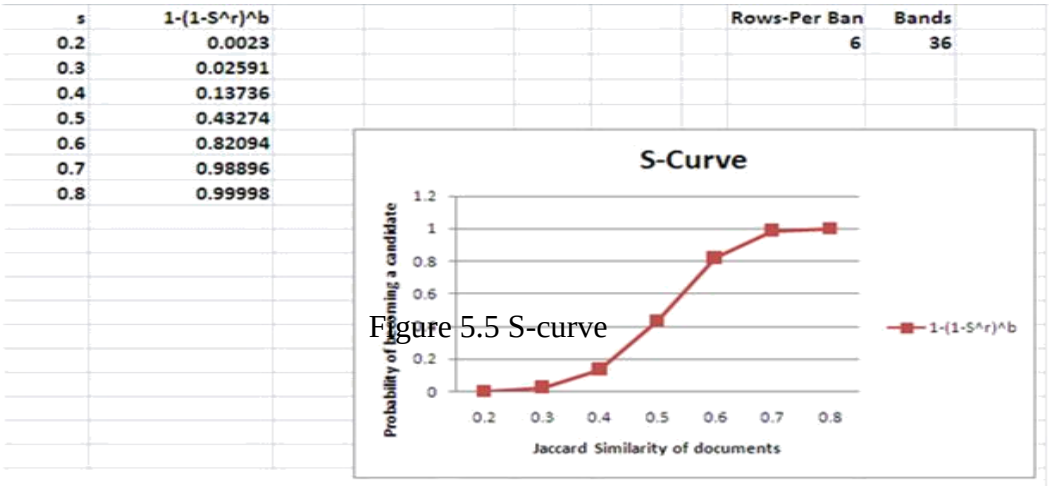


Figure 5.5 S-curve

Chapter 6

Implementation and Results

6.1 Proposed Scheme

To examine the output of the proposed scheme 300 research papers from different domain were downloaded, and assembled as files consisting of information about articles from 5 categories which include machine learning, data mining, network security, cloud computing and big data. These files contained information about the abstract, authors, journal, and keywords of all the papers. To evaluate the performance of proposed approach, three key points were considered:-abstract, keyword and conclusion. Since abstract provides the maximum information about the content of research paper it has been given highest priority as compared to other two categories. Here we find similarity score by altering various parameters (bucket, File size, Hash functions) and collecting the results and analyze it..

6.2 Proposed algorithm of finding similar documents

Input: Research papers

Output: Clusters of papers

Step1: Select all files (.txt)

Step2: Convert each set of files into shingles

Step3: After storing shingles, create characteristic matrix

Step4: Calculate the MinHash signature for each document. The MinHash algorithm is implemented using the random hash function which prevents computation of random permutations of all of the shingle IDs signature matrix and creation of signature matrix.

Step5: Compare all MinHash signatures to one another.

Step6: Calculate Jaccard similarities from above Signature matrix.

Step7: Dividing the signature matrix of n rows into b bands of r rows such that (n=b*r) .

Step8: After dividing columns of signature matrix into equal bands column vectors of the

matrix are hashed in a large bucket by using different hash function for different bands. Consider SIG (i,c) an element of the signature matrix for the hashing function i and column c. Firstly, put SIG (i, c) equal to infinity for all the 'i' values and 'c' values. To calculate the signature matrix, the rows of characteristic matrix row by row and column by column. Calculate each row by going through the following steps:

i) Calculate hash values for each row.

ii) For each c column, do the following steps:

a) If the column c in the row r has a value of '0', no action

b) If the column c in the row r has a value of 1, then for each i = 1, 2,...,n if the value of hi(r) is smaller than SIG (i,c), then consider the value of SIG (i,c) equal to hi(r).

6.3 Implementation Details

Step 1: All the text files containing abstract, conclusion and keywords are selected in as shown in Fig 6.1 and kept as three separate files because according to the proposed scheme LSH is applied on different categories separately.

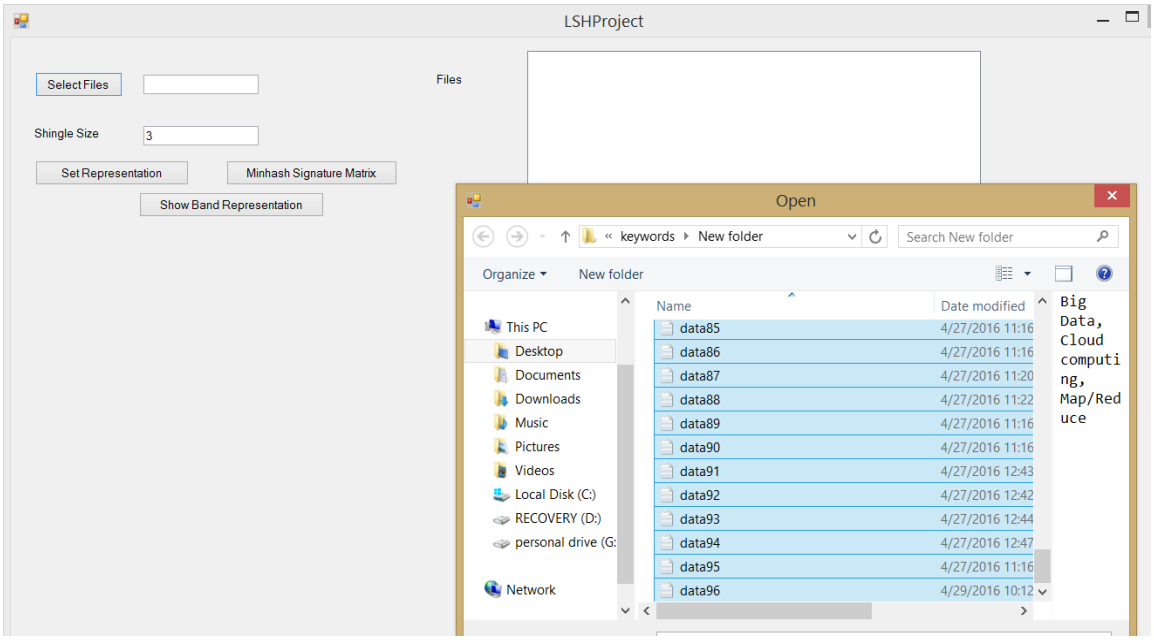


Figure 6.1: Selection of text Files

The selected files are displayed in the list box. The maximum time required to select the data of large files is shown in the first box and the size of shingle is selected in figure 6.2 single size *i.e.* k=9

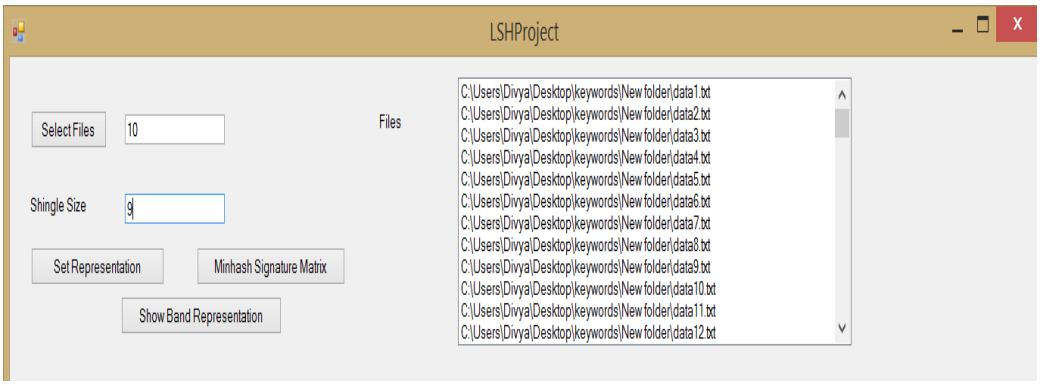


Figure 6.2: Display of files and shingle size

Step 2: Characteristic matrix is calculate. In Figure 6.3, first column in the list box shows the index number of shingles and the next corresponding seven columns presents the hash values of the shingles and the rest of the columns display the characteristic matrix where the corresponding entry of the shingles shows in the matrix.

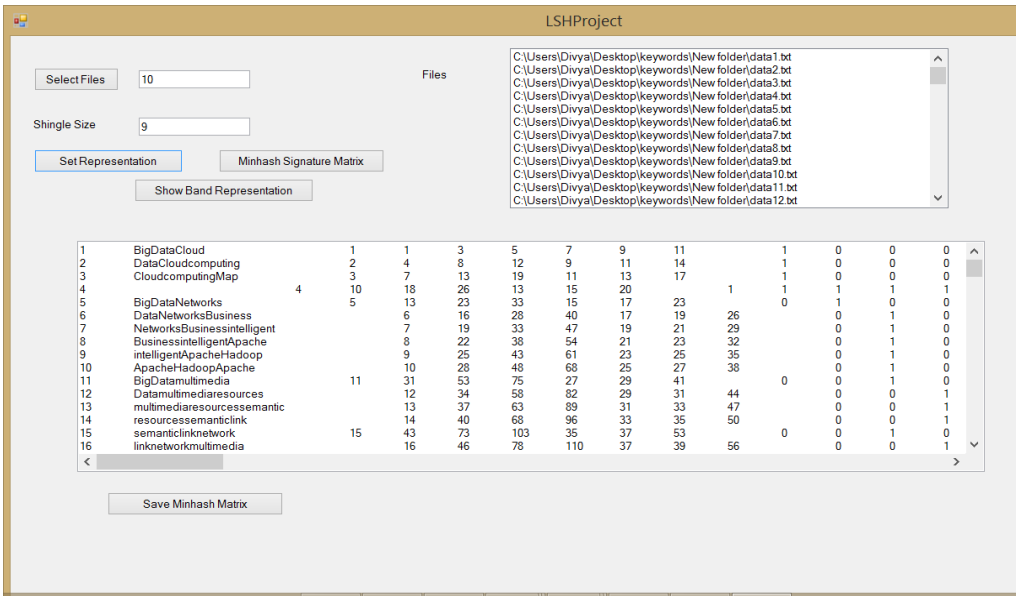


Figure 6.3: Hash values of shingles

In Figure 6.4, the entry “1” in the matrix shows that the corresponding shingle is present in the file and “0” indicates that it is not present. Here we take the text files which contain keywords.

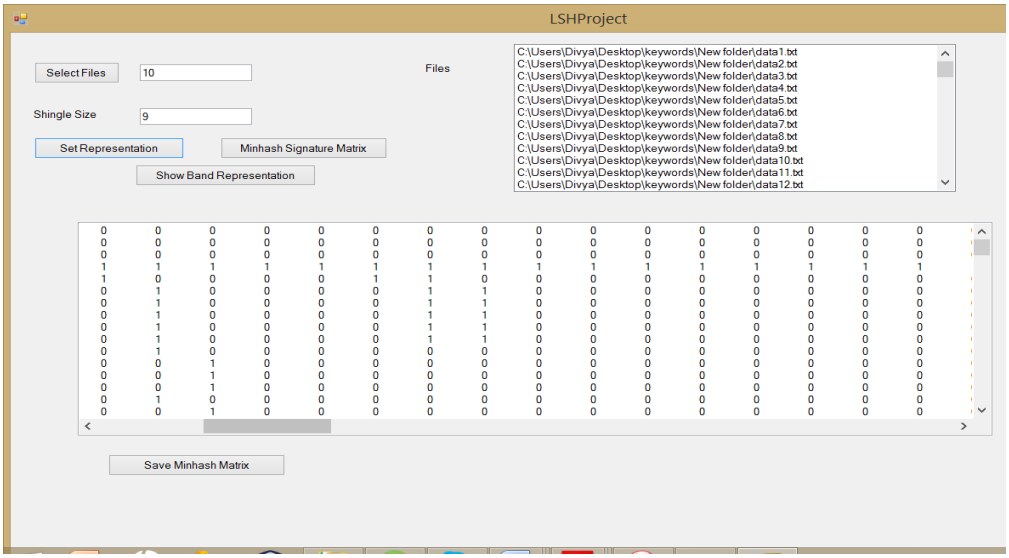


Figure6.4: Characteristic Matrix

Once the characteristic matrix is generated, Minhashing algorithm is applied and minhash signature matrix is formed figure 6.5

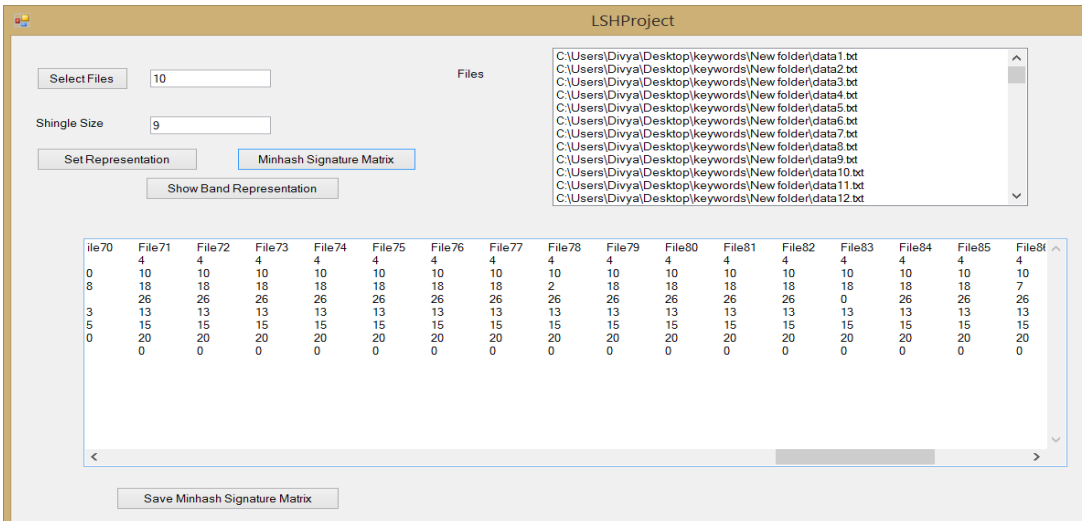


Figure 6.5 Minhash signature matrix

6.4 Results

The above algorithm has been used for 300 text files containing abstract, keywords and conclusion on same context for different domains.

- Shingles size is taken as 9 since large amount of data is considered.
- Total number of files which are coming in the same bucket from different combination is clearly presented in the graphs.
- Various parameters like size of bucket, count of hash functions, size of files are used to show the difference in similarity score in files
- It is clear that files having conclusion exceeds the percentage of similarity of files of distinct domains as compared to only abstract and keywords

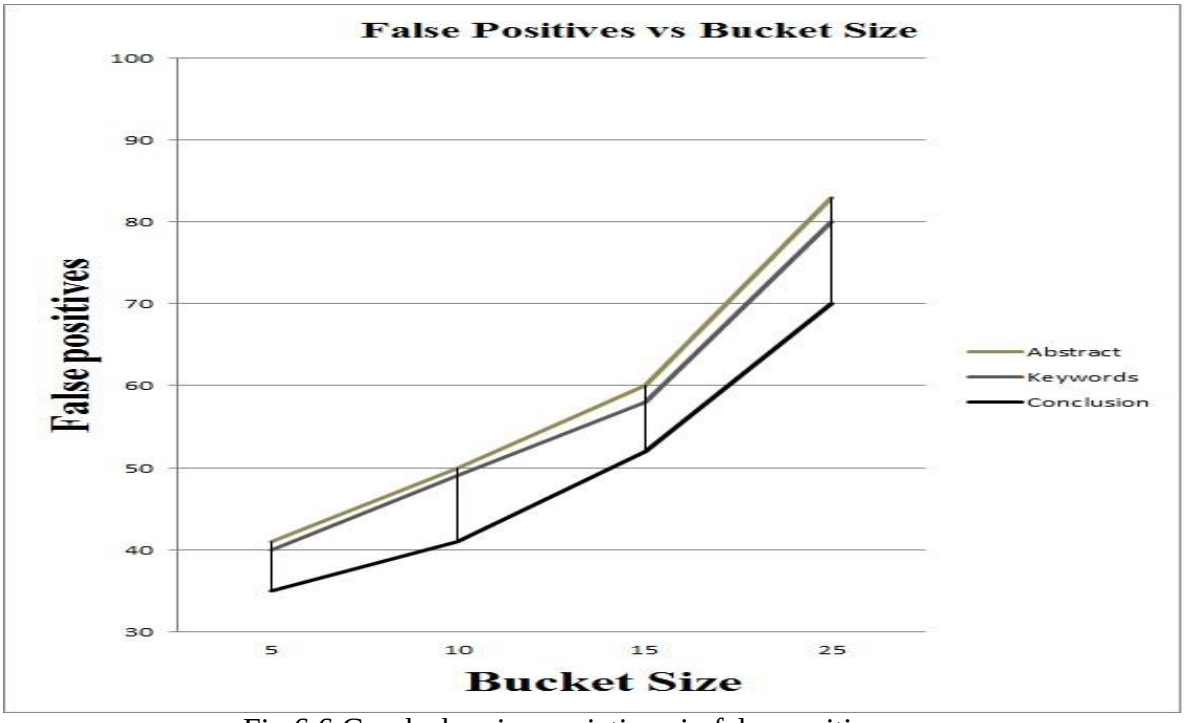


Fig 6.6 Graph showing variations in false positives

Figure 6.7 shows how variation in the hash functions changes the result when other parameters(bucket size, file size and bands) are kept constant. In order to achieve better accuracy in searching more and more hash functions should be used. It is evident from the Fig 6.15 that abstract shows more similarity than keywords.

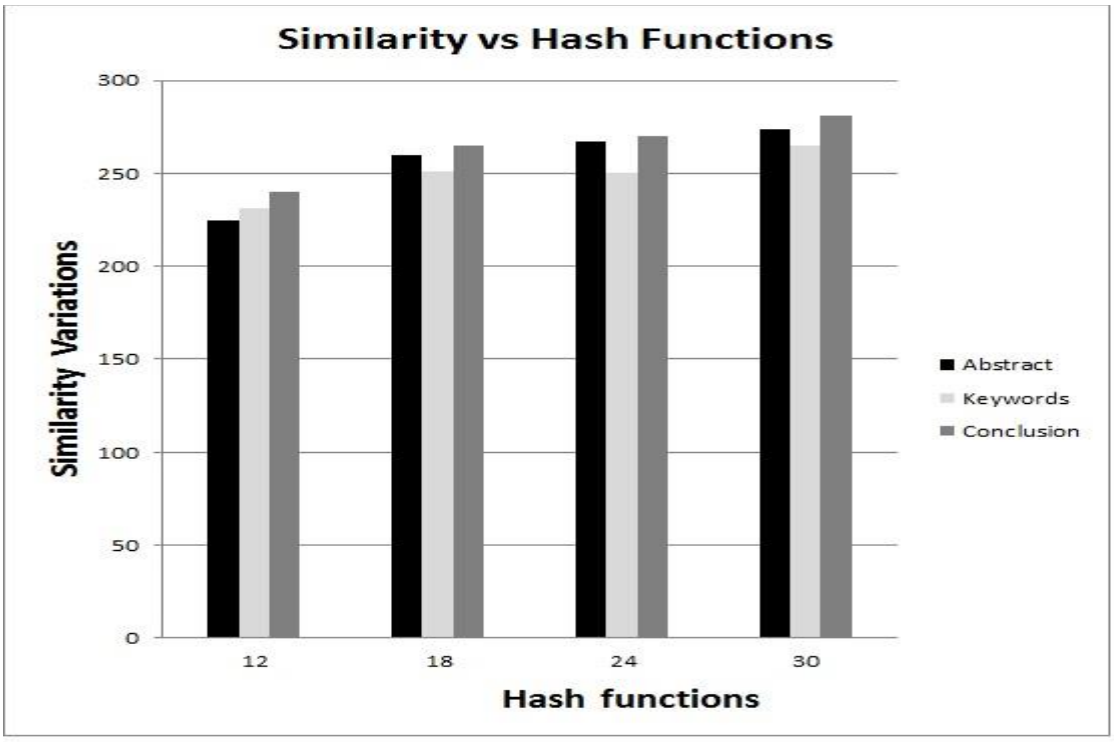


Fig 6.7 Graph showing variations in similarity by changing hash function

Size of data is a crucial factor while performing searching on any item. As we increase the size of data more and more similar files are coming in same bucket but false positives are also increasing with the increase in size. Fig 6.8 depicts how similarity varies with the data size.

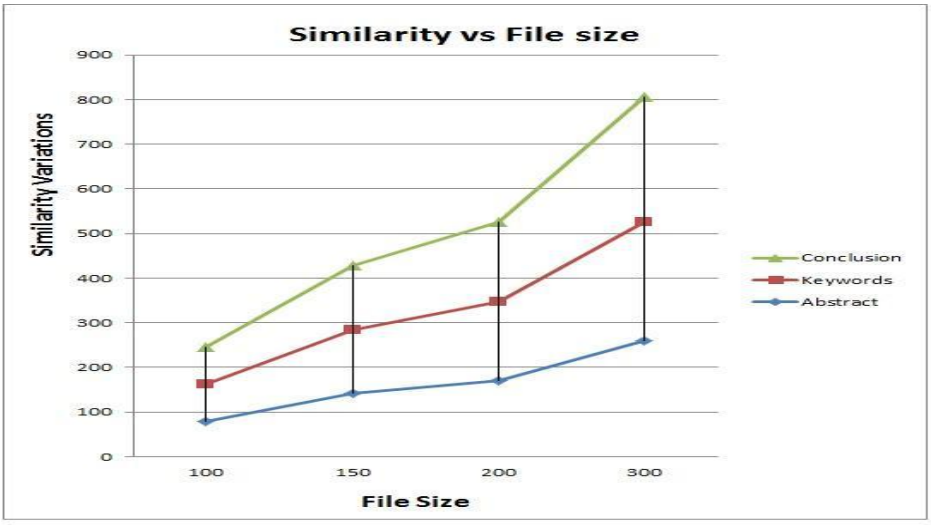


Figure 6.8 Graph showing variations in similarity by changing file size

CHAPTER 7

Conclusion & Future Scope

7.1 Conclusion

As observed in the thesis, identifying similarity between research papers is an important issue which need consideration and calculating Jaccard similarity of the shingles takes large amount of time. Locality Sensitive Hashing technique, based on Approximate Nearest Neighbor Search has been applied to find similarity among various research papers. By using banding technique research papers of same domain comes into the same bucket and one can easily identify the similarity between the papers. Characteristic matrix is generated by using shingles of files and hash functions are used to produce random permutations. Searching of a shingle linearly depends on file size, i.e. O(n), which is significantly high time when processing large files. So using Minhashing and LSH saves time and provide efficient results.

7.2 Future work

In this work, simple hash functions are used in Locality Sensitive Hashing.

- Some technique can be applied which can decrease false positives rate.
- For better performance, we can change the shingling technique by extending shingles into super-shingles.
- By creating Min hash matrix space complexity is increasing so in future we can modify algorithm of minhash matrix to resolve this issue and extend this algorithm in parallel processing environment

References

[1] Joeran Beel, Bela Gipp, Stefan Langer, and Corinna Breitinger.Research Paper Recommender Systems: A Literature Survey.",International Journal on Digital Libraries (2015)

[2] Zezula, P., Amato, G., Dohnal, V., Batko, M. "Similarity Search The Metric Space Approach" Published by Springer 2006, XVIII, 220 p., Hardcover ISBN: 0-387-29146-6 January 2006

[3] Roger Weber, "Similarity Search in High-Dimensional Vector Spaces" , 3rd International Conference of Zurich citizen of Oberuzwil (SG), Switzerland,2012

[4] Wilbert Heeringa , Jelena Golubovic, Charlotte Gooskens, Anja Schüppert, Femke Swarte, "Lexical and orthographic distances between Germanic, Romance and Slavic languages and their relationship to geographic distance" 6th international conference on computing technologies ,2010

[5] Thanh Ngoc Dao, Troy Simpson. "Measuring Similarity between sentences" Published by Springer 2004, XVII, 200 p., Hardcover ISBN: 0-385-29146-6 January 2004

[6] Yihua Chen, Eric K. Garcia, Maya R. Gupta, "Similarity-based Classification: Concepts and Algorithms" Journal of Machine Learning Research" 747-776 Submitted 9/08; Revised 2/09; Published,2009

[7] Francesco Ricci and Lior Rokach and Bracha Shapira, "Introduction to Recommender Systems Handbook", Recommender Systems Handbook, Springer, , pp. 1-35 ,2011`

[8] Edwards, Charlotte, "Online Intermediaries and Liability for Copyright Infringement" Keynote paper at WIPO Workshop on Online Intermediaries and Liability for Copyright, Geneva. World Intellectual Property Organisation (WIPO). p. 9. Retrieved September 2010.

[9] A. Rajaraman, J. Ullman, "Mining of Massive Datasets", Cambridge University Press, December 30, 2011.

[10] P. Indyk and R. Motwani, "Approximate nearest neighbor: towards removing the curse of dimensionality", Proc. Symposium on Theory of Computing, 1998.

[11] Jose Ayala, "Integrated Circuit and System Design.",Springer. p. 62. ISBN 978-3-642-36156-2 ,2012

[12] Sidorov, Grigori, Velasquez, Francisco, Stamatatos, Efstathios, Gelbukh , Alexander; Chanona-Hernández, Liliana. "Syntactic Dependency-based N-grams as Classification Features" ,LNAI 7630. pp. 1–11. ISBN 978-3-642-37798-3. Retrieved 7 October 2014.

[13] Francesco Ricci and Lior Rokach and Bracha Shapira, "Introduction to Recommender Systems Handbook", Recommender Systems Handbook, Springer, pp. 1-35 ,2011

[14] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation", Journal of Machine Learning Research,2003.

[15] Broder, A.Z. "Identifying and Filtering Near-Duplicate Document", In proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching, pp. 1-10, 2000.

[16] Chris cornelisi, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation", Journal of Machine Learning Research,2003

[17] Kalaivendhan.k,Sumathi,"An Effecient Clustering Method To Find Similarity Between The Documents", International Journal of Innovative Research in Computer and Communication Engineering, March 2014

[18] Edwin Simpson," HAC algorithm used for finding similarity",Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Istanbul, Turkey, May 4–8, 2015

[19] Wael H. Gomaa."A Survey of Text Similarity Approaches, "International Journal of Computer Applications", April 2013.

[20] Mikhail Bilenko, "Adaptive Blocking: Learning to Scale Up Record Linkage" ,In Proceedings of the 6th IEEE International Conference on Data Mining'Hong Kong, December 2006

[21] J.romes, "A Review on Pre-Clustering Approach to K-Means Clustering", International Journal of Innovations Advancement in Computer,2012

[22] Aleksander Kołcz, "Adaptive Near-Duplicate Detection via Similarity Learning", SIGIR'10, July 19–23, 2010, Geneva, Switzerland. ACM 978-1-60558-896-4/10/07,2010

[23] Broder, Andrei Z.*; Min-wise independent permutations"*, Proc. 30th ACM Symposium on Theory of Computing (STOC '98), *New York, N pp. 327–336*,doi*:*10.1145/276698.276781,1998

[24] Dominik Schnitzer , "A fast audio similarity retrieval methods for millions of music tracks", Springer Science Business Media, LLC 2010

[25] Josephson,A., "Modeling LSH for Performance Tuning",CIKM'08 October 26–30 2008 Napa Valley California USA, ACM 978-1-59593-991-3/08/10,2010

[26] A. Torralba, R. Fergus, W. T. Freeman, "80 million tiny images: A large dataset for non-parametric object and scene recognition," ,Trans. PAMI, 2008.

[27] Zixiang Yang , "hierarchical, non-uniform locality sensitive hashing and its application to video identification", IEEE,0-7803-8603-5/04/2004

[28] L. Qin et al., "Multi-probe LSH: Efficient indexing for high-dimensional similarity search", Proc. VLDB, 2007

[29] N. Katayama, and S. Satoh, "The sr-tree: an index structure for high-dimensional nearest neighbor queries", SIGMOD, pp. 369-380, 1997.

[30] M. Datar el al., "Locality-sensitive hashing scheme based on p-stable distributions", Proc. ACM Symposium on Computational Geometry, 2004.

[31] G. Junhao et al., "Locality-Sensitive Hashing Scheme Based on Dynamic Collision Counting", ACM, 2012.

[32] A. Broder et al., "Min-wise independent permutations", Proc. Theory of computing, ACM Symposium, New York, USA, pp. 327-336, 1998.

[33] A. Broder, "On the Resemblance and Containment of Documents", Proc. Compression and Complexity of Sequences, Washington, DC, USA, pp. 21-29, 1997.

[34] Chauhan Sachendra Singh ,Shalini Batra. "Finding Similar Items using LSH and Bloom Filter", IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCCT),May 2014

[35] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing", In VLDB, 1999.

[36] R. Panigrahy, "Entropy-based nearest neighbor algorithm in high dimensions", Proc. ACM-SIAM Symposium on Discrete Algorithms, ACM, New York, USA, pp. 1186-1195, 2006

[37] P. Indyk and R. Motwani. "Approximate nearest neighbor: towards removing the curse of dimensionality", Proceedings of the Symposium on Theory of Computing,

1998.

[38] T. Elsayed, J. Lin, and D. Metzler, “When close enough is good enough: Approximate positional indexes for e cient ranked retrieval”, In CIKM, 2011.

[39] Venu Satuluri, Srinivasan Parthasarathy, “Bayesian Locality Sensitive Hashing for Fast Similarity Search”,38th International Conference on Very Large Data Bases, August 27th 31st 2012.

[40] Andoni, A., Indyk, “Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions”, In Foundations of Computer Science, 2006, FOCS’06, 47th Annual IEEE Symposium,2006.

Reflective Diary

December, 2015

Day after the final exams of EST I when the placement scenario is there I had the presentation on my thesis topic as a MID Semester evaluation. It was the first time when I prepared and understand my problem statement of thesis properly and clearly. Here we have to submit a report on our thesis topic and I was in bad mood little bit because next day was my birthday and as usual I have to study. But at last presentation went pretty well. After that winter vacations started and we all went to our homes .

January, 2016

After coming back I felt lazy but at last work is work. In starting I realized that how difficult it was for me to implement my problem. I have started my work on .net.But after trying again and again I came closer to results than I got stuck don't know where. After a gap of two or three days I started exploring further about my research topic i.e. (LSH) and hence came across various other observations and techniques for my topic. I observe that I have to do little changes in my work so I started examining more papers and techniques.

February, 2016

In the first week of the month I completed collecting more data sets as it requires various sources and time to convert it into text form. Again I started my implementation where I stuck in previous month. At the end of month I got my results yipppeee .My guide also approved my work. I felt a sigh of relief. I was very happy also.

March,2016

I started preparation for writing my research paper as it needs lot of good writing skills and collection of good sources of data. But my mom dad 25th anniversary was coming so

I have to go back to home and it took around one week. I got a week late according to my schedule. After mid of march I started writing my paper as I have to send it in April. I found it difficult in the starting but later I got the paper writing skill. I have read many research papers on "LSH" and hence I manage to write my research paper at the end of march .I was feeling very good as my work was going smooth. So it is a complete chain of processes that are needed to be carried out for any proper work on time.

April, 2016

On 7 April, I send my paper and then just started waiting for the acceptance notification. Till then I started collecting data for my thesis writing so that I should not face any difficulties when I write my thesis. In middle of all this I kept trying to make my results more accurate and proper, observe various others parameters to refine my results.

May,2015

On 1 may I get the acceptance notification Hurrah! This is the Thesis summation month, we have to sum up all the research on implementation tasks for final thesis submission. So I fulfilled the following objectives for my thesis and hence started with my thesis writing.

- To study various techniques and tools available for data analytics and visualization
- To develop a research based application using various algorithms and .NET
- Execution of LSH algorithms on extracted data from research papers of all domains.
- Preparing video presentation ,posters and peer reviews.

List of Publications

- Divya Gupta and S. Batra, “Research Paper similarity using Locality sensitive Hashing ”, 2016 IEEE 3rd International Conference on Microelectronics systems and circuits , .[Accepted]

- VIDEO LINK https://youtu.be/ai2fDMWhQ7Y