Evaluating Algebraic Model Based Information Retrieval Algorithms for Small and Large Data set

Muzafar Rasool Bhat

Assistant Professor
Depart of Computer Science,
Islamic University of Science and Technology, Kashmir, India

Abstract: Different algebraic model-based information retrieval algorithms produce different retrieval results and have different efficiencies. For extending studies in information retrieval, a researcher often requires detailed comparison of results produced by different algorithms. Using Precision and Recall tests, this paper evaluates efficiency of three main algebraic model-based information retrieval algorithms namely TFIDF, VSM and LSI. Suitability of these algorithms for small and large data collections is analyzed. We have used text collections MED and CACM in this research. A simple algorithm for extraction of terms and identifying stop words is proposed. This work uses term frequencies procedure for weighing extracted terms. It further studies impact on efficiency of LSI by using varying number of singular values from 50 to 250.

Index Terms **- information retrieval, TFIDF, VSM, LSI.**

I. INTRODUCTION

Information retrieval models that have been developed so far can be broadly classified into three models: i) Set-theoretic Models ii) Algebraic Models iii) Probabilistic models.

Set-theoritic Models represent documents as set of words. Retrieval is performed usually using set-theoretic operations. Common models include Standard Boolean Model, Extended Boolean Model and Fuzzy retrieval.

Algebraic Models represent documents and queries usually as vectors or matrices. Similarity is determined by measuring angle between query vector and document vector. Ranked retrieval is performed using common matrix operations on document vector and query vector. Common models include Vector Space model, Generalized Vector Space Model and Latent Sematic Indexing etc.

Unlike Set-theoretic model and Algebraic model, Probabilistic model treat the process of document retrieval as a probabilistic inference. Ranked retrieval is performed by computing probability that a given document is relevant to a given query. Bayes theorem is generally used in probabilistic models. Common models include Binary Independence Retrieval Model, Probabilistic Relevance Model and Latent Dirichlet Allocation (LDA).

II. PROBLEM STATEMENT

For a given collection D of documents, suppose user enters a query Q, where Q = $w_1, w_2,\ldots,w_n$, a sequence of words w_i, we wish to determine a subset D* of D such that for all d ϵ D*, d is the relevant document to users query Q.

We need to formulate an information retrieval system with an objective to maximize probability of finding a relevant document d from collection of documents D with respect to a given query Q i.e.

$$MaximizeP(d|q,D) \quad (1)$$ (Berger & Lafferty, 1999)

In Algebraic Model based Algorithms, both components (i.e. documents and queries) are represented as vectors. This system can be formulated using a 4-tupple (Ho & Funakoshi, 1998) I = (D, J, Q, f), where D is a finite collection of documents, J is a set of indexing terms extracted from collection D. Q is the set of queries and function f is a ranking function. Function f: Q x D→ R` assigns a numeric value to each pair (q, d), where q ϵ Q, d ϵ D and R` is the set of non-negative real numbers.

Function f: Q x D→ R` is defined as f(q, d) = qT x d. , Where qT x d = r ϵ R` is a non –negative real number. q, d are query vector and document vector respectively. Transpose of query vector q is taken for multiplication compatibility.

Ranked retrieval can be performed using function f that measures the angle between the document vector d and query vector q. This angle measures degree of relevance of a document d to a query q. Efficiency of an Algorithm is evaluated by computing Precision and Recall values from the set of retrieved documents.

Rest of the paper is organized as follows. In Section III we formally review the available literature on algorithms based on algebraic models after giving problem statement in section II. A detailed description of TFIDF, VSM and LSI algorithms is given in section IV. Section V deal with experimental evaluations. Our discussion on the obtained results starts in section VI. Section VII concludes, discussing open issues that need further research.

III. LITERATURE REVIEW

Standardized evaluation of IR began as early as 1992 with the initiation of the annual text retrieval conference (TREC) sponsored by Defense Advanced Research Projects agency (DARPA) and National Institute of Standards and Technology [1]. TREC participants Index a large text collection and are provided search statements and relevance judgments in order to judge the success of their approaches. In this paper, algebraic model-based algorithms TFIDF, VSM and LSI are evaluated on MED and CACM datasets. These algorithms vary in terms of their complexity in implementation and efficiency of retrieval. Relevant retrieval of an algorithm is measured in terms of Precision and Recall.

For detailed overview as well as understanding supporting mathematical theory of algebraic model-based algorithms, one can refer to [2] where author has in detail explained use of linear algebra for information retrieval. Use of matrices and vector spaces in an information retrieval system is explained in [3]. While explaining use of TFIDF algorithm, [4] has determined which words in a corpus of

documents may be favorable to use in a query. Reference [5] studies TFIDF, LSI and multiword algorithms for text classification. Author while describing text classification, explains information retrieval as a major part of text classification. Author analyses TFIDF, LSI and multiword based on two kinds of properties of indexing terms i.e. statistical and semantic property.

Despite its strength, TFIDF has few limitations as argued by [4] and [5]. Reference [4] finds that in terms of synonyms, TFIDF does not check relationships between words extracted from corpus or that of a query. Furthermore, TFIDF does not check plural of a word. TFIDF therefore treats each distinct word a separate indexing term thus reduces weight Wd (weight of a term in a document d) of an individual word. This limitation could present an escalating problem for larger collections as studied by [4].

Existing methods for text-retrieval tasks can be primarily divided into two categories i) keyword oriented and ii) matrix-oriented category. Keyword oriented category manipulates key words directly using certain data structures and retrieval algorithms. However, matrix-oriented methods change keyword representation of documents into a term-by-document matrix and few decomposition techniques like Q.R factorization and SVD for improving resulting term-by-document matrix of a given collection of documents. Matrix methods generally show better performance than literal matching as claimed by [3]. Reference [3] further illustrates representation of a document using vectors besides comparing matrix methods in text-based information retrieval system using VSM.

Reference [3] explains matrix formation from a given document collection as well as use of Vector Space. Reference [3] further explains process of retrieving information using Vector Space Model (VSM). Studying essential dimensions of Latent Semantic Indexing (LSI), [6] starts his work with detailed explanation of VSM. Mechanism of ranking documents with respect to their relevance with a given query is also explained by the author. In a survey, [7] elucidates in detail Information Retrieval (IR), use of VSM and state of the art, both research and commercial, in this field besides explaining probabilistic methods of analyzing and retrieving documents.

In a typical IR scenario, while users formulate queries, a specific sequence of words, they are generally interested in the concepts or topics implied by these keywords. They generally expect that documents and queries could be matched using higher level features than words. For this purpose, [8] has proposed latent semantic analysis to convert high dimensionality word-space representation of a document to a low dimensionality vector of topics. Reference [8] discussed initially the algebraic foundation of LSI. Work carried out by [8] was further discussed by [2, 3]. Available Literature on LSI describes singular value decomposition (SVD) as a decomposing process that after finding Eigen values and Eigen vectors of a given term-document matrix, calculate singular values of term-document matrix. Based on those singular values, SVD approximates original term document matrix by a rank reduced matrix. Proper interpretation of LSI in geometric context is available in [3]. References [3, 9] argue that real power of extracting the hidden thematic structure or latency of LSI comes from SVD.

Although researchers have advanced the use of LSI and have also suggested theoretical understanding of it, however to our understanding, [10] was the first to study the values produced by LSI. Besides other advancements in LSI, like PLSI, [11] describe LSI in terms of a subspace model and propose a statistical test for choosing the optimal number of dimensions for a given collection. Reference [6] explores the appropriate k dimensions to which SVD can be truncated to. For practical purposes, optimal k can be chosen by running a set of queries with known relevance to documents in a collection and the value of K for which retrieval performance is best, can be chosen as optimal K [6]. References [6, 12] claim that optimal value of K lies in the range of 100 – 300 dimensions.

Given the available literature on TFIDF, VSM and LSI and their mathematical understanding as well as theoretical approximation, different experimental results drawn from different datasets motivate that besides the simplicity of implementation of TFIDF, it shows better results than other algorithms for few collections of smaller sizes. For having poor semantic quality, growing size of a collection can be a serious issue for TFIDF. VSM behaves a similar way as its variant LSI. LSI as claimed to have better semantic quality has however lot of computational effort involved in singular value decomposition. For collections that are dynamic in nature, SVD updating is also serious issue.

IV. Description of Algebraic model based algorithms

Among various information retrieval systems that have been developed in the recent past, systems that work on algebraic model-based algorithms, model the data using matrices. User's query is modeled as a vector (a column vector or a row vector). Relevant information which user wants to collect from data is extracted by simple vector operations. Collections (datasets) which are larger in size result in larger matrices. Familiar algebraic operations like orthogonal factorizations, singular value decomposition can be used to approximate large matrices by rank reduced matrix of smaller size. These basic algebraic operations have led to few information retrieval algorithms which include I) TFIDF, II) VSM and III) LSI.

Before we proceed to next section, we will give a formal description of these algorithms. The purpose of this description is to show how fundamental mathematical concepts from linear algebra can be used to manage and index large text collections

TF-IDF

A collection D of n documents can be reduced to a finite list of distinct indexing terms by identifying words that occur in multiple documents after removing prepositions and articles etc. Number of times term occurs in a document is called local frequency of that term while as global frequency refers to number of times this term occurs in the whole collection. Semantic content of each document d of the collection D can be generated from these distinct terms. TF-IDF works by determining weight Wd of each distinct term W using local and global frequency of W in a specific document using following equation.

$$Wd = fw,d * log(|D|/(fw,D)) \quad (2)$$

Where fw, d is the local frequency of w, |D| is the size of the corpus, and fw, D is the global frequency of w (Salton & Buckley, 1988, Berger, et al, 2000).

This calculation helps to determine how relevant a given word is for a particular document. Words that are common in a single or a small group of documents tend to have higher TFIDF values than common words such as articles and prepositions. The formal procedure for implementing TF-IDF has some minor differences over all its applications, but the overall approach works as follows.

1) For a collection D of documents, form a list words that exist in at-least two documents after removing commonly occurring words like prepositions and articles etc. from the list.
2) Prepare term-document matrix of this collection.

3) Find local frequency fw,d and global frequency fw,D of each word w with respect to each document d and collection D. (This can be simply done by row total and column total of term-document matrix).
4) Weight each word w with respect to each document d of collection D using the equation (2).
5) Input users query q, parse it into constituent words. Extract distinct relevant words that are not commonly occurring words such as prepositions and articles. Translate this query into corresponding query vector i.e. matrix with a single row and no. of columns equal to total number of distinct words as extracted from Step 1. (This can be simply done by putting 1 for the term(s) that exist(s) in a query for its respective index position. All other entries will be zero).
6) Add Wd of each word as extracted from Step 4 with respect to each document using the following equation.

$$\text{Total} = \text{Total} + \text{Wd} \qquad (3)$$

(This can be simply done by multiplying each document vector with query vector. Sum = result of multiplication).
7) Return those documents that have higher values for Total after deciding some threshold value.

Vector Space Model (VSM)

VSM explores the geometric relationship between the document vector d and a query vector q by measuring the angle between them. A document vector d that makes a minimum angle with query vector q is treated most relevant to query q. Angle between document vector d and query vector q is computed using following equation.

$$Cos(\theta) = ((d^t * q)/(\| d \|_2 * \| q \|_2)) \quad (4)$$

Where d^t is transpose of document vector, q is the query vector. ||d|| and ||q|| represent Euclidean norm of document vector and query vector respectively. After computing angle between document vector d and query vector q, results are sorted. Minimum value for $Cos(\theta)$means document is most relevant to the query q. This equation can be used to rank documents with respect to each query. Retrieval system using VSM algorithm ranks the documents in-order of their relevance to a given query q.
This algorithm works as follows.
1) Repeat Step 1 to Step 5 as mentioned in TFIDF Algorithm.
2) Find angle between query vector and each document vector using equation (4).
3) Return those documents that have smaller values for $Cos(\theta)$after deciding some thresh hold value.

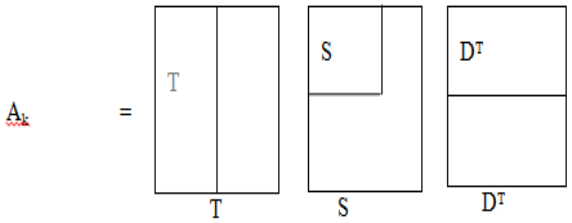

Figure 1: Pictorial Representation of SVD

Latent Semantic Indexing (LSI)

Term document matrix A, resulted from any collection of documents can be decomposed using Singular Value Decomposition (SVD). This decomposition process decomposes term-document matrix A into three matrices: a term by dimension matrix T, a singular matrix S and document by dimension matrix D (Figure 1). Where number of dimensions "r" is the rank of term-document matrix A. Matrix A can be re-computed using following equation.

$$A = T * S * D' \qquad (5)$$

The objective of singular value decomposition is not to re-compute A using T, S and D'. However, A can be approximated by reducing dimensions of T, S and D' respectively. This rank reduced matrix becomes basis for LSI. This dimension reduction of term-document matrix is accomplished by removing k+1 to r columns of T, k+1 to r columns and rows of S and k+1 to r rows of D'. This process is pictorially explained in fig. 1. This dimension reduction process is thought to reduce the noise in term-document matrix. Further researchers claim that this process reveals the latent structure present in the collection [6]. Queries are converted into vectors. Choosing optimal dimensionality reduction parameter for a collection needs to be chosen carefully. Research carried out by [6, 12] provides base for choosing k. As a standard practice, optimal value of k for a given collection is chosen by running a set of queries with known relevance. The value of k that results in a better performance is treated as optimal value of k for that collection. Value of k generally lies in the range of 100-300[6, 12].
Relevance of Query q with a document d is computed by measuring angle between reduced document vectors and query vector q using equation (5). This equation provides a similarity score for each document with respect to a given query. Like VSM retrieval, scores are sorted. Document vector(s) that make(s) minimum angle with the query vector is treated most relevant document to a given query.
Stepwise implementation of LSI is under:

1. Repeat Step 1 to Step 5 as mentioned in TFIDF Algorithm.
2. Use SVD to decompose term-document matrix resulted from Step 1 to find constituent matrices T, S and D'.
3. Find value of dimensionality reduction parameter k.
4. (Value of k can be calculated by running set of queries with known relevance. The value of k that results in better performance is treated as optimal value of k).
5. Remove k+1 to r columns of T, k + 1 to r rows and columns of S and k+ 1 to r rows of D' to compute dimensionally rank reduced term-document matrix Ak. Compute Ak using following equation. –

$$A_k = T_{mxk} * S_{kxk} * D'_{nxk} \qquad (6)$$

6. Find angle between query vector and each document vector using equation (4).
7. Return those documents that have smaller values of $Cos(\theta)$

V. Experimental Evaluation

Two Datasets MED and CACM pertaining to diverse fields and of different sizes (i.e. small and large sized datasets) are chosen to evaluate TFIDF, VSM and LSI. These datasets contain 1400 and 3204 text documents respectively. These datasets were obtained from [13]. Compressed files obtained from [13] contain text files i.e. documents, a separate file of queries for each collection and a relevance report that lists queries and documents of collection that are relevant to a given query.

Before algorithms TFIDF, VSM and LSI were tested on datasets, some preprocessing was needed. This preprocessing included i) extraction of each word from the dataset ii) Identification of all those words (commonly called as terms or indexing terms) from list of extracted words that describe the dataset iii) Identification of commonly occurring words(commonly known as stop words) like articles, prepositions etc.
For identifying distinct words that occur in at least two documents, following proposed procedure was implemented.

Input:
Li, List of Words extracted from document i
Lj, List of Words extracted from document j
Output:
Words that are occur in both document i and document j.
Procedure:
$L_i = \{w_{i1}, w_{i2},\ldots,w_{in}\}$, $L_j = \{w_{j1}, w_{j2},\ldots,w_{jm}\}$, k = 0
For each word w_{in} in L_i
  For each word w_{jm} in L_j
    If (w_{in} equal to w_{jm})
      k++
    End if
    If k > 1
      Add w_{in} to the list of words that are common to document i and document j.
      k = 0
    End if
  End for
End for

Algorithm 1: Procedure for extracting common terms from two documents.

For extracting stop words from both the data sets MED and CACM used in our study, a file containing list of 536 stop words obtained from [13] was used. A procedure that compared each extracted distinct word from a collection with the list of 536 stop words was implemented. Words that were found in the list of 536 stop words were discarded for further processing. Result of the preprocessing is contained in TABLE 1.

VI.RESULTS AND DISCUSSION

For evaluating efficiency of information retrieval algorithms, two metrics Precision and Recall have been used in this study. Precision precisely refers to how successfully an algorithm retrieves the documents that possibly are relevant to a given search query. Recall Refers to how a given algorithm separates irrelevant documents from the relevant ones. These two metrics are defined as:

$$Precision = m1/n1 \quad (7)$$

Where m1 = (Relevant documents) Observed ∩ (Retrieved Documents) and n1 = (Relevant Documents) Expected

$$Recall = m2/n2 (8)$$

Where m2 = (Relevant documents) Observed ∩ (Retrieved Documents) and n2 = Retrieved Documents.
Here (Relevant documents) Observed refers to number of relevant documents retrieved by the algorithm while as (Relevant Documents) Expected refers to how many documents are relevant to the query as per relevance report of a given dataset with respect to a given query.

These two measures Precision and Recall are inversely related proportional.

TABLE 1:`1`Results of Pre-processing on MED and CACM.

Dataset.	Total No. Documents.	Tokens Extracted.	Stop Words Identified.	Non embedded symbols	No. of Index Terms
MED	1400	164976	352	220	4534
CACM	3204	182034	366	241	4128

To estimate efficiency of an algorithm, values of both best and worst possible Precision and Recall measures are calculated. An average value of these measures is considered as efficiency of an algorithm. Average value of Precision and Recall is generally called as Average Precision and Average Recall.

Here we present results of each algorithm (TFIDF, VSM and LSI) on both data sets MED and CACM. This section also presents results of LSI algorithm on both data sets MED and CACM using varying number of singular values

Performance of TF-IDF, VSM and LSI on MED and CACM Data sets

In our experiment, we evaluated efficiency of an algorithm by running standard set of queries. Retrieval results were noted and compared with the relevance report of the dataset. Two counts were maintained i) Total number of documents retrieved and ii) Total number of relevant documents retrieved. Based on these two counts, average Precision and Recall values have been calculated. The observations from the results as mentioned in TABLE 1.1 are enumerated below.

1. In our studies (Table 2.1), we have found that TF-IDF shows better performance in terms of average precision and average recall values for MED collection, a small sized collection which contains 1400 documents.
2. We have also observedthat if we remain specific to precision only then TF-IDF returns documents that are more relevant to the query than VSM and LSI for MED collection (Table 2.1 and Figure 2.1).
3. Since encoding TF-IDF is simple and straight forward, this makes it more appropriate for forming the basis for more complicated algorithms on information retrieval system.

TABLE2.1: Average Precision, Recall values of TFIDF, VSM and LSI Algorithms

TFIDF		VSM		LSI	
Average Precision	Average Recall	Average Precision	Average Recall	Average Precision	Average Recall
0.527226	0.167088	0.303333	0.125194	0.333333	0.107603
0.537972	0.223308	0.291667	0.146207	0.34	0.139985
0.51641	0.275011	0.30348	0.171353	0.325	0.155886
0.507292	0.335354	0.293028	0.213797	0.335417	0.210976
0.483291	0.364634	0.290351	0.235238	0.317974	0.226297
0.468957	0.374231	0.265026	0.279742	0.316053	0.250525
0.43741	0.449107	0.258016	0.307087	0.296135	0.281316
0.380108	0.488202	0.243689	0.326105	0.292308	0.319102
0.346436	0.539346	0.225245	0.366801	0.272569	0.338841
0.314133	0.567911	0.230175	0.367789	0.270037	0.35572
0.288271	0.583597	0.213316	0.417662	0.275654	0.385826
0.276279	0.615158	0.207648	0.437123	0.268772	0.424039
0.266852	0.638895	0.195198	0.46084	0.259085	0.441063
0.241279	0.657789	0.184061	0.477975	0.250307	0.466782
0.228252	0.686123	0.175713	0.509411	0.236241	0.478185
0.214621	0.697009	0.178404	0.5208	0.222092	0.497525
0.218103	0.713466			0.209577	0.519251
0.195201	0.726713			0.200528	0.544359
0.181645	0.753754			0.191243	0.562805
0.150021	0.757783			0.183437	0.569946
0.139112	0.767355				

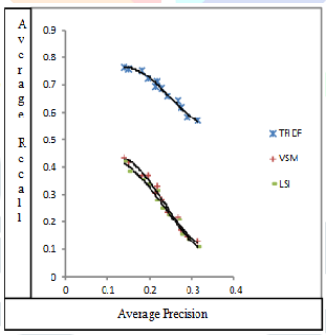

Figure 2.1:Average Precision Recall curves of TFIDF, VSM AND LSI

4. Despite strenghts, it has been found that TF-IDF has few limitations as well. i) it has been observed that TF-IDF does not take synonyms in consideration. In its implementation, before treating a distinct term as an indexing term, TF-IDF does not check synonyms of that indexing term. ii) In terms of plurals, it was noticed that TF-IDF treated word and it's plural as two different indexing terms.
5. VSM and LSI have shown significantly better performance for a large sized collection CACM (Table 2.2 and Figure 2.2). In our study, it has been observed that as compared to TF-IDF, on an average VSM and LSI better distinguishes relevant documents and irrelevant documents while retrieving a document from a collection. This is possibly due to better semantic quality of VSM and LSI. This conclusion is substantiated by [6]. Exploration of Semantic property of indexing terms seems independent of size of collection.
6. As per IR literature, LSI outperforms VSM significantly. In our studies, we have found that performance improvement of LSI over VSM is not statistically significant (Table 2.2 and Figure 2.2). However this study has revealed that unlike VSM, its variant LSI has the ability to handle higher dimensionality.

TABLE 2.2: Average Precision, Recall values of TFIDF, VSM and LSI Algorithms for CACM.

TFIDF		VSM		LSI	
Average Precision	Average Recall	Average Precision	Average Recall	Average Precision	Average Recall
0.206751	0.215420	0.104794	0.161805	0.130964	0.175346
0.188077	0.238855	0.104021	0.172291	0.123795	0.183024
0.181204	0.250866	0.102045	0.183831	0.120050	0.191586
0.164916	0.255349	0.098598	0.192105	0.116907	0.200577
0.161483	0.270431	0.097205	0.201939	0.115117	0.211462
0.157762	0.276764	0.091071	0.208023	0.108167	0.215368
0.154119	0.288680	0.090285	0.214417	0.106647	0.228852
0.149098	0.292676	0.088612	0.224768	0.107414	0.246235
0.145227	0.294508	0.087425	0.234118	0.104727	0.256239

0.143831	0.306223	0.085799 0.083295	0.244313 0.255828	0.105273 0.101972 0.097772 0.097171	0.272939 0.279169 0.279169 0.291610

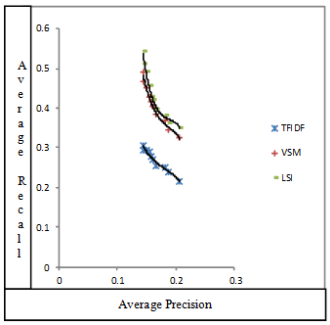

Figure 2.2: Average Precision, Recall curves of TFIDF, VSM and LSI Algorithms

7. Limitations of TF-IDF to handle synonyms and plurals as mentioned at 4 probably became the reason for showing poor results in retrieving documents from CACM dataset which is larger in size than MED (Table 2.2).

8. In our studies we have further noticed that in Rank reduced matrix, semantic content remains preserved if optimal rank reduction parameter k is chosen carefully since we have found that use of Rank Reduced Term Document Matrix in LSI instead of complete term document matrix resulted from collection has least effect on performance of LSI unlike VSM that needs original Term Document Matrix (Table 2.1, Table 2.2 and Figure 2.1, Figure 2.2).

Performance of LSI using varying number optimal dimensionality reduction parameter k for small dataset MED and large dataset CACM.

1. In our study, we have observed that arbitrarily chosen rank reduction parameter k can severely affect efficiency of LSI algorithm. Using range of singular values from 50 to 200, average performance of LSI varies significantly with minimum at k = 50 for MED and k = 150 for CACM (Table 3.1, Table 3.2 and Figure 3.1, Figure 3.2).

Table 3.1: Average Precision, Recall values of LSI Algorithms with Singular Values from 50-150

LSI(with singular values from 50 - 150)		
No. of Singular Values	**Average Precision**	**Average Recall**
50	0.284781	0.423925
55	0.283947	0.434321
60	0.288991	0.434289
65	0.290702	0.441648
70	0.294167	0.44661
75	0.290702	0.452671
80	0.293421	0.446539
85	0.297675	0.453452
90	0.301184	0.460156
95	0.299474	0.468037
100	0.2925	0.466065
105	0.292412	0.453934
110	0.304474	0.450542
120	0.300088	0.472482
130	0.298377	0.465755
135	0.303553	0.462131
140	0.299254	0.469372
145	0.298465	0.461543
150	0.299370	0.461791

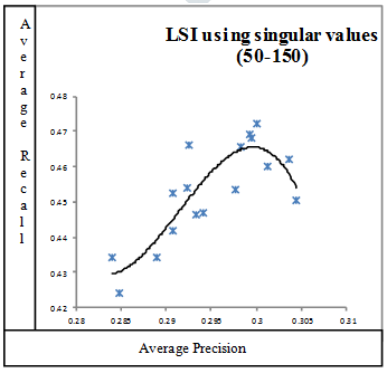

Fig 3.1: Average Precision Recall curve using LSI algorithm

2. Value of k for maximum efficiency remained elusive for both collections MED and CACM (Figure 3.1, 3.2 and Table 3.1, 3.2). For efficient performance of LSI, we took value of k by repeated runs of the algorithm. As a standard practice in IR, the value which resulted in better performance was taken optimal value of k. We tested LSI over a range of values between 100- 300 as [6, 12] claim that value of k typically range in between 100 – 300.

3. In our study, we have found that decomposition of a term document matrix to find its constituent matrices left unitary matrix, singular matrix and right unitary matrix takes lot of time. This unavoidable decomposition process to use LSI renders it almost

unusable for collections that are dynamic in nature as lot of mathematical computations are involved. To update the newly entered entries as indexing terms, constituent matrices of a term document matric needs to be recomputed. Recomputed term document matrix needs to be decomposed again.

4. Our study has revealed that theoretical interpretation of decomposition process and apropos usage of dimensionality reduction parameter k needs more research.

Table 3.2: Average Precision, Recall values of LSI Algorithm using singular values from 150- 185

LSI(with singular values from 150 - 185)		
No. of Singular Values	**Average Precision**	**Average Recall**
150	0.122014	0.158382
155	0.124309	0.163812
160	0.123936	0.159837
165	0.126220	0.169847
170	0.134362	0.183537
175	0.126070	0.170634
180	0.133418	0.179395
185	0.132406	0.177570
190	0.140136	0.185942
200	0.130964	0.175346

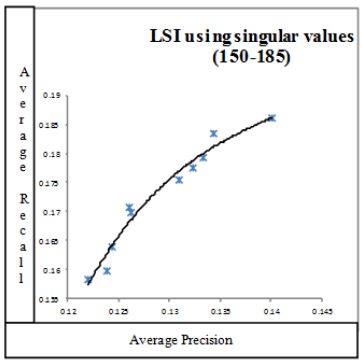

Figure 3.2:- Average Precision, Recall curve of LSI Algorithm.

VI. CONCLUSION AND FURTHER RESEARCH

In this paper, experiments have been conducted to examine the efficiency / performance of three methods: TF-IDF, VSM and LSI in information retrieval systems. Basically two kinds of properties of indexing terms are considered while formulating any algebraic model based IR system . These properties are:

i) Statistical property (i.e. discriminative power of the indexing term to identify a document in which a term occurs).

ii) Semantic property (i.e. to how much extent the indexing term can describe the text)

Lacking the standard measure to gauge statistical and semantic properties mathematically, these qualities are merely considered by intuition. In IR literature, VSM and its variant LSI are considered to better explore semantic quality of indexing terms than TF-IDF and TF-IDF explores the statistical properties of indexing terms.

Our study has revealed that for collections that are smaller in size, implementation of an algorithm that explores statistical property (TFIDF in our case) of indexing terms shows considerably better performance. However for collections where semantic content is vastly spread out and are larger in size, an algorithm that explores semantic property (LSI and VSM in this study) of indexing terms shows better performance.

Optimal reduction parameter k needs to be chosen carefully. In our studies, we choose parameter k by repeatedly running LSI algorithm using range of values from 50-200. We selected optimal dimensionality reduction parameter k by selecting a value which resulted in better performance.

It has been noticed that we need a mathematical method that addresses polysemy and synonymy of English. The basis for such a method can be representation of a document in a hyper-dimensional space so that a best variant of a given context is used as an approximation to that context. Furthermore new models like Fuzzy model and Probabilistic models may be used to further the research vis-à-vis information retrieval as these two models work completely in a different way than set-theoretic models and algebraic models.

REFERENCES

[1] Singhal, A., 2001. Modern information retrieval: A brief overview. IEEE Data Eng. Bull., 24(4), pp.35-43.

[2] Berry, M.W., Dumais, S.T. and O'Brien, G.W., 1995. Using linear algebra for intelligent information retrieval. SIAM review, 37(4), pp.573-595.

[3] Berry, M.W., Drmac, Z. and Jessup, E.R., 1999. Matrices, vector spaces, and information retrieval. SIAM review, 41(2), pp.335-362.

[4] Ramos, J., 2003, December. Using tf-idf to determine word relevance in document queries. In Proceedings of the first instructional conference on machine learning (Vol. 242, No. 1, pp. 29-48).

[5] Zhang, W., Yoshida, T. and Tang, X., 2008, October. TFIDF, LSI and multi-word in information retrieval and text categorization. In 2008 IEEE International Conference on Systems, Man and Cybernetics (pp. 108-113). IEEE.

[6] Kontostathis, A., 2007, January. Essential dimensions of latent semantic indexing (LSI). In 2007 40th Annual Hawaii International Conference on System Sciences (HICSS'07) (pp. 73-73). IEEE.

[7] Greengrass, E., 2002. Information retrieval: a survey by ed greengrass. Information Retrieval, 141, p.163.

[8] Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K. and Harshman, R., 1990. Indexing by latent semantic analysis. Journal of the American society for information science, 41(6), pp.391-407.

[9] Wu, H. and Gunopulos, D., 2002, December. Evaluating the utility of statistical phrases and latent semantic indexing for text classification. In 2002 IEEE International Conference on Data Mining, 2002. Proceedings. (pp. 713-716). IEEE.

[10]Schütze, H., 1992. Dimensions of Meaning. SC, 92, pp.787-796.

[11]Zha, H., Marques, O. and Simon, H., 1998. A subspace-based model for information retrieval with applications in latent semantic indexing (pp. 29-42). Pennsylvania State University, Department of Computer Science and Engineering, College of Engineering.

[12]Kontostathis, A. and Pottenger, W.M., 2006. A framework for understanding Latent Semantic Indexing (LSI) performance. Information Processing & Management, 42(1), pp.56-73.

[13]Letsche, T.A. and Berry, M.W., 1997. Large-scale information retrieval with latent semantic indexing. Information sciences, 100(1-4), pp.105-137.

[14]General Resources Resources @ Glasgow Information Retrieval Group. Available at: http://ir.dcs.gla.ac.uk/resources.html (Accessed: January, 2019).