Information Retrieval by Modified Term Weighting Method using Random Walk Model with Query Term Position Ranking

Md Masudur Rahman
Roll No: 040201
Shamima Yeasmin Mukta
Roll No: 040202

Computer Science and Engineering Discipline
Khulna University
Khulna-9208, Bangladesh
February, 2009

Information Retrieval by Modified Term Weighting Method using Random Walk Model with Query Term Position Ranking

A thesis submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering (CSE) at Khulna University.

Abu Shamim Mohammad Arif — Supervisor
Assistant Professor
Computer Science & Engineering Discipline
Khulna University
Khulna-9208

Dr. Md. Rafiqul Islam — Second Examiner
Professor
Computer Science & Engineering Discipline
Khulna University
Khulna-9208

Dr. Md. Rafiqul Islam — Head of the Discipline
Professor
Computer Science & Engineering Discipline
Khulna University
Khulna-9208

Acknowledgement

First of all we would like to thank to Almighty Allah the merciful, who has given us the ability, intelligence and energy to complete our thesis.

We wish to convey our heartiest gratitude to our honorable supervisor Abu Shamim Mohammad Arif, Assistant Professor, Khulna University for his kind cooperation, suggestion, guidance and continuous encouragement throughout the course of study. We would like to mention that his broad-minded suggestion, inspiration, careful observation have helped us to continue our work in friendly environment and actually he made us done the work.

We also want to express our sincere regard and heartiest gratitude to our second examiner Prof. Dr. Md. Rafiqul Islam, Head, Computer Science & Engineering Discipline, Khulna University, for his valuable advices, suggestions and encouragement.

Moreover, we would like to thank our honorable teachers of CSE discipline for their goodwill support and important advices to complete our thesis. We also like to thank all of our seniors and friends for their suggestion, encouragement, wise and valuable information often and on. Without those helps and co-operation of all of them mentioned above, it was really difficult for us to get through such a work.

Abstract

Information retrieval is one of the most demanding research areas that are gaining importance day by day. Term weighting is a core idea behind any information retrieval technique. Nowadays, graph based ranking algorithms are playing vital roles in term weighting. Study about recent works shows that in calculating term weight, structural information associated with term frequency is used and term dependency is considered using graph based approach. In our thesis, we have introduced a method of information retrieval using graph based random walk model considering positional values in computing inverse document frequency of a term with a view to provide exact scores to terms and assigning suitable weights to terms in user provided query to emphasize user's interest. Experiments have shown that our approach provides improvement in precision & recall of information retrieval system.

Table of Contents

List of Figures

List of Tables

CHAPTER I

Background

1.1 Introduction

Information retrieval is a demanding sector in modern time. The straight-forward definition of Information retrieval is given by Lancaster in [9] – 'Information retrieval system does not inform (*i.e*. change the knowledge of) the user on the subject of his inquiry. It merely informs on the existence (or the non-existence) and whereabouts of documents relating to his request'. Day by day information retrieval is becoming more difficult due to increased amount of electronic information available. People are surrounded by information but they can't use them effectively due to overabundance and lack of efficient methods to retrieve that. Thus the necessity of efficient methods arises, which deal with providing relevant information in response to user queries maintaining trade-off in precision and recall. There are already several algorithms existing to fulfill the needs for information.

Generally, a graph based ranking algorithm is used to decide the importance of a vertex within a graph. It recursively calculates global information which is computed from the entire graph, instead of depending only on local vertex-related information. Recently graph-based ranking algorithms have been used with graphs extracted from text and successfully applied to text classification [2], key phrase extraction [3], and word sense disambiguation as in [3]. Brin and Page proposed the random walk term weighting algorithm applying on graph, has been used in citation Analysis, social network and analysis of the link structure of the web as in [1]. The basic idea implemented by a random walk algorithm is to provide importance to a vertex (term) within the graph. When one vertex links to another one, it is basically casting a vote for other vertex [2]. The higher number of votes that are cast for a vertex indicates the higher importance of that vertex. Also, the importance of the vertex casting the vote determines how important the vote itself is. Therefore, the score of a vertex is determined based on the votes that are cast for it and the score of the vertices casting these votes.

Term weighting is one of the most important issues in any information retrieval system which is intended to classify the indexing terms by assigning them weights. Actually the main purpose of term weighting is to improve both recall and precision of the retrieval. A graph based ranking algorithm was first time proposed by Blanco *et al.* in order to calculate terms weight in the field of information retrieval as in [1]. They use random walk graph-based ranking algorithm and its Text-Rank adaptation as in [3] to derive term weights from textual graphs. Their experiments show that random walk weight (*rw*) can perform at least comparably to traditional term frequency (*tf*) weights in information retrieval.

One of the most important objectives of an Information Retrieval System is to retrieve relevant documents with respect to user queries. Many information Retrieval systems are based on Vector Space Mode (VSM), where documents and queries are represented as vectors of terms and their similarity scores are computed to use an inner product. Yue-Heng *et al.* propose an improved term weighting scheme, where they consider term positions and information gain of term for term weighting as in [4]. This paper provides an improved and optimized term weighting scheme based on the detail analysis to the nature of VSM model. The same indexing term in different position in a document, for example, title, abstract, text itself, etc, has different expression competence to the document content.

The main drawback associated with Yue-Heng *et al.* as in [4] model is that it relies on a bag-of-words approach. It implies features independence, and disregards any dependencies that may exist between terms in the texts. It defines a 'random choice'. This procedure can be effective to capture the relevance of a term in a local context, but not in case of the global context. So, an argument was made by Hassan *et al.* as in [2] that the bag-of-words may not be the best technique to capture term importance. Term dependencies have been considered by Hassan *et al*. in their proposal of a random walk model for term weighting as in [2]. Here, term weighting integrates both the locality of a term and its relation to the surrounding context. They model this local contribution using a co-occurrence relation in which terms that co-occur in a certain context are likely to

share between them some of their importance. However, they do not consider term position to calculate the weight of term.

The combination of the concept of term position approach of Yue-Heng *et al*. [4] to random walk model proposed by Hassan *et al*. in [2], a new approach is proposed by Rafiqul *et al.* in [5] and here a bit improved performance is experienced than the previous algorithms. They weigh a term and exploit the relationship of local information of a vertex (term position) as well as global information (information gain) and term dependency. Taking into account these three factors their result show the improvement of random walk model and providing better term weighting for information retrieval.

In our thesis, we are going to propose an information retrieval system that considers structural information both for calculating term frequency and inverse document frequency which are estimation of localized contribution and globalized contribution of a term respectively [11]. We are going to use random walk model graph based ranking algorithm for considering term dependency among terms within a document [1]. Here, we are also proposing a new approach for query processing by assigning suitable weight to individual query terms to emphasize user's interest for better precision and recall of the system.

1.2 Motivation

Information retrieval has become quite tough nowadays as the amount of information is becoming huge and it needs very efficient algorithms which will retrieve document without delay, maintaining precision & recall to a satisfactory level. So, the future prospect of this research field is quite bright and it demands for more efficient & brilliant ideas for developing exact solution to human needs for information. So we are motivated to study in this field. Several searching algorithms have been developed so far with a view to address different level of problems in information retrieval which are becoming more complex day by day. However, we have observed that there have been few approaches so far towards information retrieval using random walk model. As term-weighting is a core idea behind any searching technique, we are introducing a method of information retrieval using random walk model considering positional value for

computing inverse document frequency, *idf* for a term within the document along with assigning trained weight to terms in the user provided query. The main purpose here is to apply exact importance (score) to term rather than increment scores that really deserves to have. We hope this approach will improve the precision & recall satisfactorily.

1.3 Problem Formulation

Problem formulation is the most important part of any research work. In an information retrieval system, there are three types of problems embedded-document representation, query representation and document ranking. Firstly, documents should be accessed for term weighting which can be compared to a machine learning process. Secondly, user query should be provided in a form suitable for machine execution and thirdly, document ranking should be represented according to their relevance to user query. There are many methods to implement all those features but often there raises a question of satisfaction and perfection.

1.4 Objectives

We have planned and also have proposed some solution to fulfill the following objectives.

- To develop a method that uses term-dependency in document ranking.
- To develop a method which exploits the structural information computing of TF, IDF and ranking of query terms.
- To modify the existing method used by Rafiqul *et al.* as in [5], so that number of iterations through corpus is reduced.
- To process query assigning positional trained value to the query term for better information retrieval.

The overall objective is of this thesis is to develop an improved method for term-weighting, document-ranking using efficient version of Random Walk Model.

1.5 Organization of the Thesis

Our thesis has been organized into six chapters. Each chapter contains distinct type of information regarding our work.

Chapter I (*Background*): This chapter describes the motivation, problem formulation, objectives of our thesis.

Chapter II (*Literature Survey*): This chapter represents all the previous works in our sector. It contains individual details, features implemented by each method.

Chapter III (*Information Retrieval by Modified Term weighting using Query Term Position Ranking*): This chapter represents our proposed solution towards the fulfillment of our objectives.

Chapter IV (*Experimental Results & Analysis*): This chapter shows the evaluation results of our thesis works.

Chapter V (*Concluding Remarks*): This chapter contains conclusion, limitation of our work. Here we have also notified about our future works.

CHAPTER II

Literature Survey

2.1 Introduction

The brief discussion of different methods of Information Retrieval is given in this chapter. There are three basic Information Retrieval model: Boolean Model, Vector Space Model (VSM), and the Probabilistic Model. A lot of extensions have proposed by researchers on the basic methods of information retrieval and some of them will be related to our research. So, after describing the basic information retrieval models, we will discuss some of the extension of these methods proposed by researchers.

2.1.1 Boolean Model

The Boolean Retrieval Model is a model for information retrieval in which we can pose any query which is in the form of Boolean expression of terms. The terms are combined with the operators AND, OR and NOT. The model views each document as just a set of words. This model considers that index terms are present or absent in a document. As a result the index term weights are assumed to all be binary. A query q is a composed of index terms linked by above three operators. The Boolean Model predicts that each document is either relevant or non-relevant [6]. The Boolean model is a simple retrieval model based on set theory and Boolean algebra. Because the concept of a set is quite intuitive, this model provides a framework that is a very easy for users of an information retrieval system to grasp. The queries are specified as Boolean expression, which have precise semantics [6].

2.1.2 Vector Space Model

The representation of a set of documents as vectors in a common vector space is known as the vector space model. It is fundamental to a host of information retrieval operations ranging from scoring documents in a query, document classification and document clustering. The vector model recognizes that the use of binary weights is too limiting and proposes a framework in which partial matching is possible as in [6]. The vector model

assigns non-binary weights to index terms in quires and in documents. These term weights are ultimately used to compute the degree of similarity between each document stored in the system and user query. By sorting the retrieved document in decreasing order of this degree of similarity, the vector model takes into consideration documents which match the query terms only partially. The effect is that the ranked document answer set is a lot more precise than the document answer set retrieved by the Boolean model as in [6].

The vector space model can be described in the following way. Assume, *N* is the total number of documents in the system and n_i is the number of documents in which the index term k_i appears.$freq_{i,j}$ is the raw frequency of term k_i in the document d_i **.** Then, the normalized frequency $f_{i,j}$ of term k_i, in document d_i is given by

$$f_{i,j} = \frac{freq_{i,j}}{max_i \times freq_{i,j}}$$

Where, the maximum is computed over all terms which are mentioned in the text of the document. The inverse document frequency for k_i, which is the inverse of the frequency of a term k_i among the documents in the collection, is given by

$$idf_j = log \frac{N}{n_i}$$

Where *N* is the number of total documents in the corpus and n_i is the number of documents containing the term *j*.

The best known term-weighting schemes use weights which are given by

$$W_{i,j} = f_{i,j} \times log \frac{N}{n_i}$$

2.1.3 Probabilistic Model

Probabilistic model was first introduced in 1976 by Robertson and Sparck Jones. Later became known as the binary independence and retrieval (BIR) model as in [6]. This model attempts to capture the information retrieval problem within in probabilistic framework. Given a user query, there is a set of documents that contains exactly the relevant documents and no other. This is the ideal answer set. The querying process can

be thought of as the process of specifying the properties of an ideal answer set. The problem is that we don't know exactly what these properties are. Therefore, a method has been made to initially guess what these properties are. This initial guess allows us to generate a preliminary probabilistic description of the ideal answer set which is used to retrieve a first set of documents. An interaction with the user is then initiated with the purpose of improving the probabilistic description of the ideal answer set. This can happen by the user looking at the retrieved documents and deciding which ones are relevant and which ones are not. The system can then use this information to refine the description of the ideal answer set.

The probabilistic model assumes that the probability that user will find a document to be relevant depends on the query and the documents representations only.

2.2 Extension of the Basic Methods

The above models discussed could not satisfy the demands for Information retrieval as complexity arises. So, the basic models are modified to fulfill the requirement. This section will describe some modified methods of the basic methods for Information retrieval.

2.2.1 An Improved Term Weighting Scheme for Vector Space Model

Term weighting has been explained by controlling the exhaustiveness and specificity of the retrieval, where the exhaustiveness is related to recall and specificity to precision [7]. The term weighting for the vector space model has entirely been based on single term statistics. There three main elements:

i. Frequency of term j within document i, $tf_{i,j}$

ii. Inversed document frequency of term i, $idf_j = log(\frac{N}{n_j} + .01)$

Here, N is the total number of documents in the corpus and n_j is the number of documents holding term j.

iii. Length normalization factor. Normalizing the document vectors is used to ensure that the documents are retrieved independent to their lengths.

These three factors are multiplied together to make the resulting term weight. The weight W_j of a term j is then given by:

$$W_j = f_{i,j} \times idf_j = \frac{tf_{i,j} \times log(\frac{N}{n_j} + .01)}{\sqrt{\sum_{t \in d} \left[tf_{i,j} \times log(\frac{N}{n_j} + .01) \right]^2}} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.1)$$

Equation (2.1) assumes that the most important indexing terms should be those terms that occur certain times in one document and as few as possible in other documents. So, the foundation of VSM is the term frequency within document and inversed document frequency.

The distribution proportion of a term is an important factor to measure its weight in that document. So, Yue-Heng Sun *et al.* as in [4] proposed a method that considers structural information and information gain for term weighting. They introduced information gain (IG_j) for term j, to give more importance to a term. Here, information gain has discriminating power to identify a document. To calculate $(tf.idf)$ the equation given in [4] is as follows.

$$W_j = f_{i,j} \times idf_j = \frac{tf_{i,j} \times log(\frac{N}{n_j} + .01) \times IG_j}{\sqrt{\sum_{t \in d} \left[tf_{i,j} \times log(\frac{N}{n_j} + .01) \times IG_j \right]^2}} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.2)$$

Yue-Heng Sun *et al.* as in [4] divides the documents into several sections such as title, abstract and text, etc. The same term appears in the different positions in a document, should be considered to have different priorities (weights). So, the terms in the title should have higher weights than those in the abstract and the text and so on. To reflect the importance of term position they added some adjustable factors, shown as below:

$$tf_{i,j} = \alpha \times tf_{ij1} + \beta \times tf_{ij2} + \gamma \times tf_{ij3} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.3)$$

Where, tf_{ijk} is the frequency of a term in the k_{th} area and α, β, γ are the factors that can be adjustable according to pre-experiments where $\alpha > \beta > \gamma >= 1$.

2.2.2 Random Walk Term Weighting for Information Retrieval

At first, Blanco *et al.* as in [1] used the basic random walk graph-based ranking algorithm to weight terms of the document which calculates score of each vertex (term) using its TextRank adaptation from textual graphs. The score of a vertex is determined based on the votes that are cast for it and the score of the vertices casting these votes. Assume, $G = (V, E)$ be a directed graph with the set of vertices V and set of edges E, where E is subset of $(V \times V)$. For a vertex V_i, let $In(V_i)$ be the set of vertices that point to it (predecessors), and let $Out(V_i)$ be the set of vertices that vertex V_i points to (successors). The score of a vertex V_i is defined as follows as in [8]:

$$S(V_i) = (1 - d) + d \times \sum_{j \in ln(V_i)} \frac{S(V_j)}{|out(V_j)|} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.4)$$

Where d is a damping constant and its value can be taken between 0 and 1. It has the role of integrating into the model the probability of jumping from a given vertex to another random vertex in the graph.

2.2.3 Random Walk Term Weighting for Improved Text Classification

Hassan *et al*. proposed a system that models the weighting problem as a "random-walk" rather than "random-choice". They assumed that an imaginary reader or 'walker' who steps through the text on a term by term basis. During the walk, the importance of the term is determined by the probability of the random-walker to encounter the target term in the text. They modeled their system by using a co-occurrence relation in which terms that co-occur in a certain context are likely to share some of there importance between them. A given term has a relation to a context, and the context, of course, relates to a collection of terms. In order to model this recursive relation, they used a graph based

ranking algorithm, namely the Page-Rank random-walk algorithm as in [8], and its Text-Rank adaption to text processing as in [3].

In this paper, Hassan *et al.* summarized the best Page-Rank-based term ranking models as follows:

$\overleftrightarrow{rw0}$: It represents the basic or original model, as described in (2.4) in which it uses an undirected graph with a constant damping factor that adheres strictly to the traditional formula of Page-Rank.

$\overleftrightarrow{rwe.idf}$: This model represents an undirected graph approach that uses the weighted edge version of Page-Rank with a variable damping factor. The edge weight is calculated by the following formula:

$$E_{v1,v2} = tf.idf_{v1} \times tf.idf_{v2} \quad \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \quad (2.5)$$

Where $E_{V1,V2}$ is the edge connecting *V1* to *V2*, and $tf.idf$ represents the term frequency multiplied by the inverse document frequency.
The damping factor is expressed as a function of the incoming edges' weight, calculated as follows:

$$d_{E_{v1,v2}} = \frac{E_{v1,v2}}{E_{max}} \quad \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \quad (2.6)$$

Where, $d_{E_{V1,V2}}$ is the damping factor and E_{max} represents the highest weight for an edge in the graph. The resulting node ranking formula is:

$$S(V_a) = \frac{(1-d)}{|N|} + \sum_{V_b \in In(V_a)} C \times \frac{d_{E_{V_a,V_b}} \times S(V_b)}{|Out(V_b)|} \quad \dots\dots\dots\dots\dots\dots\dots \quad (2.7)$$

Where *N* represents the total number of nodes in the graph and *d* is the damping constant, *C* is a scaling constant.

2.2.4 An Effective Term Weighting Method Using Random Walk Model for Information Retrieval

Rafiqul *et al.* [5] proposed an approach that combines the concept of term positions approach of Yue Heng *et al.* [4] to the random walk model proposal by Hassan *et al.* as in [2]. The random walk models described in Hassan *et al.* as in [2] do not consider term position within a document and information gain of term to compute the weight of that term. So, the weighting of terms calculated using that method does not reflect the actual weight of term.

This approach calculates $tf.idf$ and random walk weights (rw) for each term in the selected documents. To calculate $tf.idf$ of each vertex equation (2.2) is used. Using the value of $(tf.idf)$, weight of an edge is calculated by equation (2.5). To determine the damping factor $d_{E_{Va,Vb}}$, weight of each edge is required in equation (2.6). The value of damping factor $d_{E_{Va,Vb}}$ varies in equation (2.7) to reflect importance of term. The more the value of damping factor, the term is more important to retrieve the related information. All the terms in the document are added as vertices in a graph to represent the document. All the terms that fall in the vicinity of a given term are considered as dependent terms. This is represented by a set of edges that connect terms to all other terms in a fixed window. To implement the edge connection this approach construct graph using hash map data structure. After constructing the graph they iterate the random walk algorithm until convergence. The threshold value of convergence is 0.0001. At the last step vertices are sorted based on their final score.

2.3 Drawbacks of Previous Works

The Boolean model has some major drawbacks. One of them is that its retrieval strategy is based on a binary decision criterion (*i.e.* a document is either relevant or non-relevant) without any notation of grading scale, which prevents good retrieval performance.

Another one is that while Boolean expressions have precise semantics, frequently it is not simple to translate an information need into a Boolean expression. Yet despite these drawbacks, the Boolean model is still the dominant model with commercial document database system and provides a good starting point for those new to the field [6].

Theoretically, the vector space model has the disadvantage that the index terms are assumed to be mutually independent. However, in practice, consideration of term dependencies might be a disadvantage. Due to the locality of many term dependencies, their indiscriminate application to all the documents in the collection in fact hurt the overall performance [6].

The Probabilistic model assumes that there is a subset of all documents which the user prefers as the answer set for the query q [6]. The problem with this assumption is that it does not state explicitly how to compute the probabilities of relevance. The need to guess the initial separation of documents into relevant and non-relevant sets is the main disadvantage of this model. Another fact is that the method does not take into account the frequency with which an index term occurs inside a document (*i.e.* all weights are binary) and the adoption of the independence assumption for index terms.

Yue-Heng *et al.* [4] proposed information gain (IG_j) and term positions in their improved term weighting scheme for vector space model. This scheme considered *'bag of words'* approach. So, the main drawback of this approach is that it did not consider term dependency.

Term dependency was first considered by random walk model. Blanco *et al.* [1] first time used graph-based random walk ranking algorithm to weight terms in information retrieval which calculates score of each vertex. The drawbacks of this model are they did not consider edge weighting, term frequency tf, inverse document frequency idf and term position.

Hassan *et al*. [2] proposed a random walk model for term weighting which removed the *'bag of words'* approach. They introduced term dependencies to calculate edge weight in their approach. Hassan *et al.* [2] has drawbacks that they did not consider term position and information gain to calculate the weight of term.

Rafiqul *et al.* [5] introduced an effective method which combines term positions in [4] and term dependencies in [2]. Here, term position is used to calculate term frequency. They considered term dependency using random walk model. The main drawback of this model is that they did not consider term position to calculate inverse term frequency.

CHAPTER III

Information Retrieval by Modified Term Weighting Method using Random Walk Model with Query Term Position Ranking

3.1 Introduction

This section deals with the main part of our thesis. Here we have described our proposed solution and we have tried to visualize the modifications we have introduced. We have shown the modification in calculating inverse document frequency and also explained how to assign weight to query terms.

3.2 A Modified Random Walk Term Weighting Method with Query Term Ranking

So far we have studied we have observed that there are huge works in term-weighting with potential number of proposed solutions. In case of Random Walk Model not too many papers were published and in this field of term-weighting the following issues have been considered so far.

- The dependency of the words within document using graph-based approach eliminating the bag of words approach of vector space model [2]
- The variable damping factor & structural information in calculating the score of a vertex within the graph of terms within document [5]
- The random-walk approach in case of selecting the vertex within the graph [2]
- Multiple no of window size in case of calculating the score of a vertex within the graph [5]
- All the terms except the stop words are considered which gives exact assumption about document structures & dependency.

However, those are the so far discovered features added to the Random Walk Model term-weighting method. But there are always something left for perfection. We observed that following issues have not been considered yet.

- In case of term weighting structural information has been used so far in different methods all methods have considered structural information in calculating term-frequency, *tf* only, not in case of calculating Inverse document frequency, *idf* yet which is an important thing to consider.

- Structural information of terms in query provided by the user is not yet introduced in information retrieval with Random Walk Model though it is not a new concept.
- The existing methods of Random Walk Model for term-weighting used the algorithms are effective but not efficient and will perform low in case of huge sized corpus as well as greater window-size.[5] and this can be improved through dynamic IDF calculation.

In our thesis proposal we were committed to implement those above features which showed themselves as promising factors towards efficient information retrieval system. However, we have developed that proposed system and steps followed in our proposed system are as follows.

Firstly, each document's content is tokenized and stemmed after removing the stop words from the content. We used the list of stop words provided by SMART system as in [10].

Secondly, term frequency tf, inverse document frequency idf (using equation (3.3)) and information gain IG are calculated individually to compute $tf.idf$ for each term which is used to calculate damping factor $d_{E_{v1\,v2}}$. Here a text document is represented as graph containing distinct term as nodes and co-occurred words are denoted as connected nodes by edges. Then we apply random walk term weighting on that graph assigning some value to each node initially. Every time a node is selected randomly, score is calculated using equation (2.7) and next node selection is determined by the damping factor which predicts the probability of jumping to that node. This random walk continues until the score of each node converges that means the error rate falls below the threshold value (usually .0001).

Lastly, documents are provided score associated with query term's positional value as asserted previously in this chapter and term's score obtained from random walk term weighting model. Then, documents are ranked according to their score and represented as output of the search.

3.3 Formal Approach of Implementation

In our proposal submitted previously, we proposed solutions of three problems those exist in the current random walk model term weighting methods. We have successfully removed those problems in our implementation and we have observed significant positive changes in performance which we have shown in section 5.1. Here, the steps we have followed in our implementation are given below.

- Stemming
- Calculating TF & IDF.

- Calculating Term Weight
- Document Graph
- Random Walk on Graph
- Query Processing
- Document Ranking

3.3.1 Stemming

The first phase of our implementation is stemming which involves removing all punctuations, stop-words, numbers etc from the document [5]. This is a very important task to start term weighting of the terms of a corpus. To perform this stemming we have used the famous algorithm of stemming called Porter's Algorithm. This can be visualized with an example. Such as a term in ADI corpus '*subscription*' is stemmed to '*subscrip*' which is the root of the word. Stemming also performs another job, i.e. it extracts the distinct words from the corpus. For example, our implementation has found 966 & 5198 distinct words from ADI and CRAN corpus respectively.

3.3.2 Calculating TF and IDF

In term weighting for calculating term frequency tf, structural information of terms has been considered so far but for inverse document frequency computing, this structural information has not been considered yet. We proposed an algorithm to consider structural information for obtaining inverse document frequency. This algorithm was able to capture contribution of a term locally (within a single document) as well as globally (within the corpus). We have successfully implemented a way to calculate IDF considering structural information. We can see that the traditional formula for calculating IDF in [4] is stated as below:

$$idf_j = log\left(\frac{N}{n_j} + .01\right) \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.1)$$

Here, N denotes total number of document in the corpus, n_j denotes document frequency for term j and idf_j denotes inverse document frequency. We can see that no structural feature has been used to calculate idf_j. We have computed idf_j for a term j considering structural information which can be visualized through our algorithms.

3.3.2.1 Algorithm for Calculating IDF

To implement this idea we have considered a structured document containing three sections: title, abstract and body having different weights like $\alpha, \beta, \gamma, (\alpha > B > Y \geq 1)$. Let, N is the total number of documents in the corpus and initially a term j has document frequency zero. We have iterated through all documents in the corpus one document at a time where all items in a document have to be traversed. If a term j is found first time, then we have enlisted it as a new individual term together with putting its document frequency equal to 1 multiplied by positional value of the term j. The document containing term j is also saved along with setting term frequency equal to 1 multiplied by positional value. Here, the inverse document frequency is also computed each time a term is found. This algorithm ensures the removal of erroneous enlisting for a term and provides an estimation of global contribution of a term within the entire corpus. Our algorithm works as follows.

Step 1: Look for new term in the document.

Step 2: If term j is eof go to step 5, otherwise go to step 3.

Step 3: Consider the term position of term j within the document into three sections: title, abstract and body having weights $\alpha, \beta, \gamma, (\alpha > \beta > Y \geq 1)$. We have used $\alpha = 4, \beta = 2, \gamma = 1$. Now, calculate the document frequency using this formula.

$$d_j = d_j + 1 \times (\alpha_i + \beta_i + \gamma_i) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.2)$$

Here, if the term j resides in one of three sections then corresponding weight will be considered during multiplication otherwise $(\alpha_i + \beta_i + \gamma_i) = 1$ as traditional formula [4].

Step 4: Calculate IDF using the following modified formula as stated in [4].

$$idf_j = log\left(\frac{N \times (\alpha_i + \beta_i + \gamma_i)}{\sum_{i=1}^{K} 1 \times (\alpha_i + \beta_i + \gamma_i)} + .01\right) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.3)$$

Here, $1 \leq (\alpha_i + \beta_i + \gamma_i) \leq 7$, where in case of calculating document frequency either α or β or γ will have only nonzero value at a time and thus will produce a value $d_j \leq N * (\alpha_i + \beta_i + \gamma_i)$.

Step 5: Exit

Here in our algorithm we have used some probabilistic features such as estimation of global contribution of a term within the corpus. Suppose lets consider a term '*catastrophe'* has been found in title and abstract of the 1st document, in abstract and body of the 2nd document, in abstract and body of the 3rd document. Now according to the traditional method the document frequency will be simply 3 but we can see that it is a very important word for 1st document than the other two documents, but these facts are totally ignored in calculating document frequency and all documents are paid the same importance. We have visualized this necessity of marking 1st document as important document assigning the positional value as in calculating $d_j + 1 \times (\alpha_i + \beta_i + \gamma_i)$. We also have ensured that if a document can have a term both in title and abstract it will have a score of $(\alpha_i + \beta_i + \gamma_i) = (4 + 2) = 6$ and this works well as a way to prioritize a document having a term with greater positional values. We have calculated using this estimation and have got improved weighting for terms.

3.3.3 Calculating Term Weight

We proposed a solution to reduce an extra iteration from the existing algorithm proposed by Rafiqul *et.al* [5] in term weighting method using random walk model. Rafiqul *et.al* [5] used to compute idf and IG at the last of indexing of all terms which requires an extra iteration through the database of all stemmed words. This is an overhead to the performance in case of huge number of words in the database. So, to reduce the iteration we are considering the following two steps.

a. Calculate idf dynamically each time when a term is processed in a document. Here, we are adding a document-counter (df) which keeps the track of the document being enlisted together with structural information. We are using this information, each time when a term is processed, to update the value of inverse document frequency.
b. Calculate term-weight for each term dynamically using equation (2.2). As all necessary parameters for calculating W_j are at hand during the iteration so we can calculate them easily and we have done that.

3.3.4 Document Graph

After extracting the distinct words from the single document we have represented the document as a graph. As, a graph can represent only distinct nodes so document graph will also only contain the distinct terms connected to each other using edges, where edges represent the interconnection between the terms of the document [3]. During graph representation a window size, S is selected for denoting co-occurrence between words within a distance of S from the current word. This is necessary for considering the structural information among the terms of the document. Such as *'air-force'* is a phrase of two words. Each of them represents different meaning individually, means it is not like *'air'* or *'force'* . So we have to consider the relative position of the terms in the document which reveals the semantic importance of terms within a document [1]. We have considered different sized window during the generating the document graph. The higher the window size the greater the connectivity which helps to capture the semantic features of the document. Now, we are giving an example of how to build graph from a document of number .I509 of CRAN corpus [10].

3.3.4.1 Contents of Document

.I 509

.T

a graphical approximation for temperatures and sublimation rates at surfaces subjected to small net and large gross heat transfer rates .

.A

Adams , e. w.

.W

A graphical approximation for temperatures and sublimation rates at surfaces subjected to small net and large gross heat transfer rates. considers a material, acted upon by heat of conduction, which changes its state by sublimation at the heated surface. the derived method is most suitable under conditions of severe heating such as space vehicle re-entry .

3.3.4.2 Graph of the Document

From the above text document we have made a graph after stemming, where co-occurred words are represented as the edges among the nodes in the graph [2]. Here we have used window size as 2 that means only each term is connected to the next term only. The following figure represents the documents-graph.

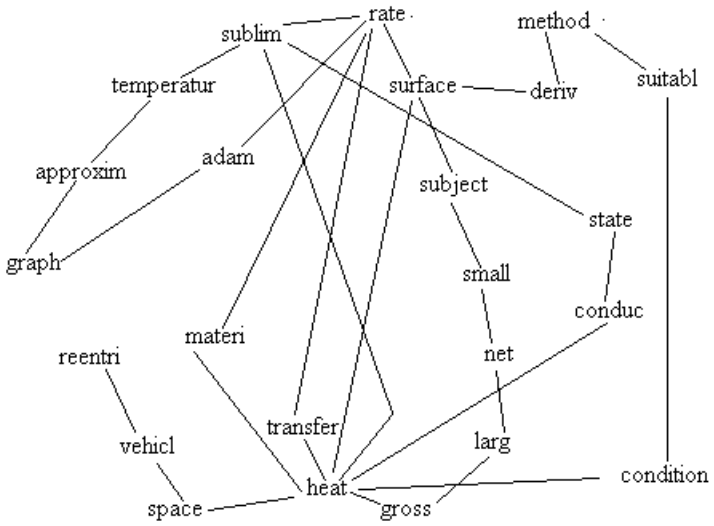

Figure 3.1: Document Graph

3.3.5 Random Walk on Graph

Random walk on graph is a newly used approach towards assigning scores to the terms (node) in the graph. It is something like the voting system. The node which has the most connectivity will have the highest score, because semantically many nodes are dependent on it [1]. Random walk model is a probabilistic approach which expresses the probability of a node of the graph currently being selected. Here as we are using the graph of weighted edge different edges carry different level of semantic values [2]. The edge weight influences the selection of the next node to move from the current node (term). The edge-weight is measured in term of a normalized factor called damping factor containing the value between 0 and 1 [2].

When the document graph is constructed, initially all nodes are assigned with some initial value on which final score is not dependent. During traversal of graph, if one node is selected randomly, then next node is selected based on the current score of neighbor nodes. The highest score node not yet

traversed is selected as the next node for scoring. If there is a deadlock, then this run of traversal is terminated and randomly another node is selected for starting traversal. This process continues until the score of the nodes converges [2]. For convergence, a threshold value (0.0001) is used to stop the random walk and the current score of each node of the graph is saved for document ranking [2].

For example, we run our algorithm on a document graph of document called .I509 from CRAN corpus [10] and have found the following scores. We know that scoring of term is mostly dependent on TF-IDF measurement and connectivity of the node (term) within the graph and we also found that evidence in the following table containing scores of several terms. From the table we can observe that term *'heat'* and *'rate'* have got higher scores than others as expected from the document graph.

Table 1: Scores of Terms

Term	Score
Heat	0.09667
Rate	0.08715
Surface	0.09097
Temperature	0.02866
Sublimate	0.04290
Transfer	0.02946
Large	0.03512

3.3.6 Query Processing

Query processing is another addition in our information retrieval system. Query representation has a great impact on the retrieval. So far, there are different approaches towards information retrieval like information retrieval using feedback, long query etc. For query processing we are performing two intermediate steps before query is submitted to the system for document retrieval as follows.

- Stemming on query
- Positional value of query term

3.3.6.1 Stemming on Query

As all terms of the documents of the corpus stemmed in the database, so, before submitting the query to the system we are stemming the query words. This step is helpful for matching query term to the document term rather than submitting the raw form of query to the search engine [13]. We have observed that this helps better retrieval of information. For example, consider the query of id .I001 for CRAN corpus [10] contains many terms unnecessary for information retrieval.

.I 001

What similarity laws must be obeyed when constructing aero-elastic models of heated high speed aircraft?

This query is stemmed to the following form better for processing by the search engine. Here we stop words are removed.

Similar law obei construct aeroelast model heat high speed aircraft

We also added a feature called partial matching of term that means if a term is not exactly in the dictionary then we considered the similarity of terms by maximum substring matching. We found this helps better performance in precision and recall of the system.

3.3.6.2 Query Term Positional Values

We proposed a solution to prioritize the query terms. Sometimes the position of a query term has an important effect that reflects human mind's intention of search. For example let us consider the query *'Random walk on graph'*. Here the query is specifies the random walk algorithm on graph where if the query is represented as *'Graph Random walk'*, here the main focus is on graph. So due to the positional difference of the query term the internal meaning (appeal) is changed. This is a heuristic approach towards searching which ensures better performance in retrieving relevant documents. We have considered some relative values for query positions which will be considered during retrieval of document. Such as when the 1st query will be given to search engine it will add extra score for *'Random walk'* term during search and document containing this term will get extra chance to be retrieved. For example, for the above query .I001 contains 10 distinct stemmed terms.

Table 2: Query term positional values

Term	**Value**	**Term**	**Value**
Similar	1.0	Model	0.4
Law	0.9	Heat	0.3
obei	0.8	High	0.2
construct	0.7	Speed	0.1
aeroelast	0.6	Aircraft	0.0

We have implemented this feature and we have found significant changes in the retrieval due to different permutation of query terms. So our system can respond to the semantic changes of the query structure.

3.3.7 Document Ranking

Document ranking is another important thing to pay attention finally and here we have used an approach called flexible matching of query terms which increases our recall. This idea is implemented in the way that whenever a query term does not match exactly with any term in the database then it selects the nearly matched term and captures the weight and adds to the document score. Thus the average score for the documents is increased and it is helpful to the retrieval of relevant document also. For example, the above query of CRAN corpus [10] produces such kind of document ranking using our implemented algorithm.

Table 3: Document ranking & scores

Ranking	Document	Score
1	486.txt	3.62495
2	51.txt	3.13811
3	329.txt	2.98487
4	573.txt	2.87873
5	13.txt	2.75043
6	663.txt	2.66954
7	332.txt	2.25792
8	435.txt	2.25050
9	305.txt	2.23183
10	359.txt	2.19349

Here documents are arranged according to their rank. Here ranking reflects relevance of document. During the development of our system, we observed that performance is based on maintaining tradeoff between precision & recall of the retrieval.

CHAPTER IV

Experimental Results & Analysis

4.1 Introduction

Development of an information retrieval system is successful when it can give better performance than the previously developed systems. To evaluate the performance of our system we have used a bench-mark evaluation system called Terrier Information Retrieval System [12]. Experiments have shown interesting and improved results both in terms of precision and recall.

4.2 Performance Factors

Performance of an IR system is measured in terms of either efficiency of effectiveness, where efficiency denotes the time complexity of the retrieval and effectiveness denotes how effectively search result is provided. We have observed recent works and found that effectiveness is considered in evaluation process there. So, we have chosen it in the evaluation of our system. This evaluation of effectiveness is based on two factors.

- Precision
- Recall

Generally, if we look at the types of document we can categorize the documents into 4 categories in the form of matrix as shown in Figure 4.1.

Irrelevant	Irrelevant & retrieved	Irrelevant & not retrieved
Relevant	Retrieved & relevant	Relevant but not retrieved
	Retrieved	Not retrieved

Figure 4.1: Matrix representation of relevant & irrelevant documents

If we look at the following figure the above matrix representation of relevant and retrieved documents will be clearer where their relationships are expressed using Venn diagram in Figure 4.2.

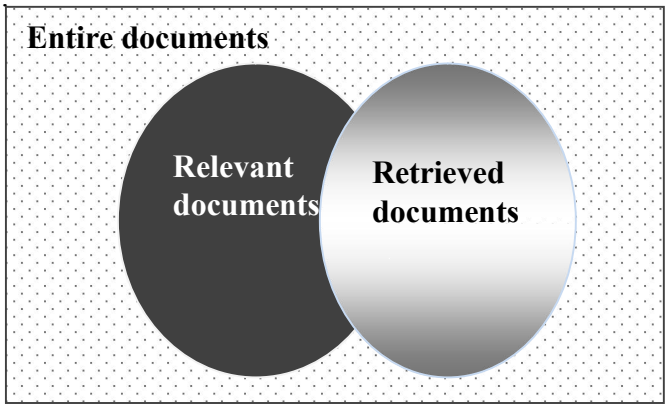

Figure 4.2: Relevant & retrieved documents

Precision can be defined as the ratio between no. of relevant documents retrieved and no of total documents retrieved. Precision is a major concern for any information retrieval system. In our experiment we have got different types of precision for different corpuses and it is clear that for better precision we have to give penalty in recall.

Recall is another major concern on which system performance depends. It can be defined as the ratio between no of relevant documents retrieved and total no of relevant documents in the corpus. We have observed that if we allow retrieve large no of documents then recall increases and reciprocally precision decreases.

$$recall = \frac{No\ of\ relevant\ documents\ retrieved}{No\ of\ total\ relevant\ documents}$$

$$precision = \frac{No\ of\ relevant\ documents\ retrieved}{No\ of\ total\ documents\ retrived}$$

4.3 Performance Analysis using Experimental Results

We have used five corpuses CRAN, CACM, CISI, ADI & MED in our experiments which have given hopeful results and we believe that if we can tune the system more accurately it will give more excellent results. Here we are analyzing the results into different sections such as term weights, precision & recall etc.

In our term weighting, we paid extra attention to the calculation of IDF of a term considering the structural information globally which was supposed to give more accurate weight for the terms of the corpus and this is a completely new idea so far we know. We have used corpuses and the following table shows some basic information about our corpuses.

Table 4: Corpuses used for experiment

Corpus	No of Distinct terms	No of Documents	Average no of terms per document	No of Queries	Average no of terms per Query	Average relevant documents per query
CRAN	5198	1400	14.2	225	5.3	15
CACM	7176	3204	46.6	64	12.7	13
CISI	7160	1460	35.5	66	12.5	17.5
ADI	966	82	25.5	35	4.5	4.85
MED	8548	1033	48.9	30	12.6	23.8

4.3.1 Term Score Performance

As we have said earlier that we have used different metrics for measuring term (node in document graph) score along with random walk model. We have used different measures for calculating IDF, so we have different scale of scores but with the same property of all previous methods and with better response towards window size. As we know, window size reserves the semantic importance for term within a document. Here in our experiment, with the increment of window size significant change in the score for the terms has been observed and it is a different approach towards exact evaluation of the content of a document. Previous works have been done on different corpuses like CRAN, CACM etc. We have done our experiments, which reveal a better score with some regular attributes in change. We believe that this is due to the modified approach of calculating IDF considering the global positional information of a term in the document. Our term score has a different metrics of measurement, so here we have represented their normal values in table.

For example, we have taken document number .I509 from CRAN corpus and .I3203 from CACM corpus respectively, where we observed significant changes in score

from the scores calculated by the existing methods. Here Table 5 represents the score of document .I509 from CRAN corpus.

Table 5: Term weighting over CRAN corpus using our proposed method

Term	Weight of Term					
	Buddha [14] (Window size N)			Proposed Method (Window size N)		
	N=2	N=3	N=4	N=2	N=3	N=4
Heat	0.01001	0.03530	0.03642	0.04420	0.04176	0.08715
Surface	0.01152	0.04419	0.04428	0.03728	0.03500	0.09097
Temperature	0.04326	0.27950	0.26667	0.03632	0.02765	0.02866
Sublimate	0.26754	0.06596	0.06441	0.038229	0.03396	0.04290
Rate	0.02453	0.02850	0.02557	0.05954	0.04803	0.08715
Transfer	0.03150	0.01951	0.01340	0.02459	0.02804	0.02964
Act	0.01001	0.01924	0.01816	0.03042	0.03199	0.04335
Large	0.01845	0.27950	0.26667	0.03158	0.03012	0.03512
Condition	0.03542	0.06596	0.06441	0.02106	0.02101	0.02101
State	0.01532	0.02850	0.02557	0.02116	0.02273	0.02150
Material	0.01142	0.01951	0.01340	0.02115	0.02204	0.02243
Vehicle	0.06543	0.01924	0.01816	0.02116	0.02128	0.02150

We have also taken document numbered as .I3203 from CACM corpus and have found the scores of some stemmed terms stated in Table 7. Here we also have observed significant changes in scores from the previous methods of term weighting for different kind of window sizes. Sometimes the score may seem too different but it is nothing but level shifted and relative scores among the terms should be considered important rather than individual scores. These scores may not be fixed all the time because they come across a randomized process called random walk on document graph.

Table 6: Term weighting over CACM corpus using our proposed method

Term	Weight of Term					
	Buddha. [14] (Window size N)			Proposed Method (Window size N)		
	N=2	N=3	N=4	N=2	N=3	N=4
select	0.03542	0.0353	0.03642	0.078804	0.081681	0.091914
definition	0.04326	0.04419	0.04428	0.076637	0.053829	0.067865
prepare	0.26754	0.2795	0.26667	0.100000	0.100000	0.100000
standard	0.06543	0.06596	0.06441	0.048879	0.060459	0.070823
committee	0.02453	0.0285	0.02557	0.058984	0.069230	0.085720
program	0.01001	0.01951	0.01340	0.038858	0.048105	0.055642
review	0.01845	0.01924	0.01816	0.038244	0.048249	0.055758

4.3.2 Precision & Recall Analysis

We have run our experiments on CRAN, CACM, CISI, MED and ADI corpus with different no of queries and for each query we get two approximation- precision & recall. Now we are representing the information in terms of graph. Basically this is not a regular graph and can't be deduced with any well-known equation. We have used MATLAB software to generate the graph from different precision & recall measurement.

For CRAN corpus we have selected the following queries .I11, .I192, .I41, .I47, .I94, .I107, .I140, .I160 and have measured average precision & recall which is represented in the following graph. Here we see significant change in the curve Figure 4.3.

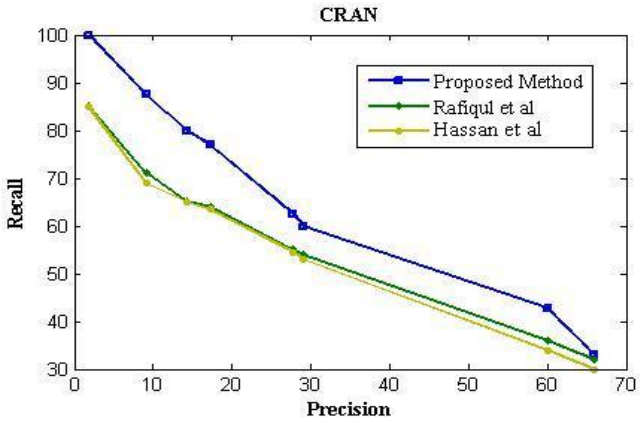

Figure 4.3: Precision-Recall for CRAN

For CACM corpus, we have selected the query number .I2, .I12, .I36, .I38, .I49, .I27, .I39, .I61 and we have computed average precision and recall and plotted those points in the curve. This curve also shows significant change in the curve. As we know, the higher the curve towards right, the higher the recall. Our curve has also showed this positive change which is demonstrated in the Figure 4.4.

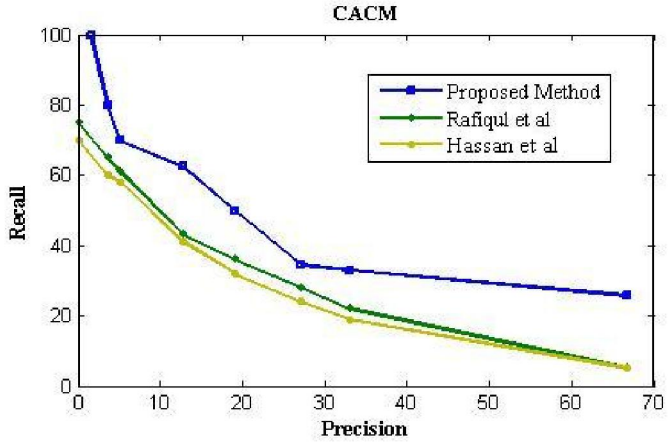

Figure 4.4: Precision-Recall for CACM

For CISI corpus we have got a bit different type of performance graph using query no .I6, .I31, .I23, .I28, .I35, .I27, .I20, .I52, .I24, .I66 which also shows higher recall & precision as stated in Figure 4.5.

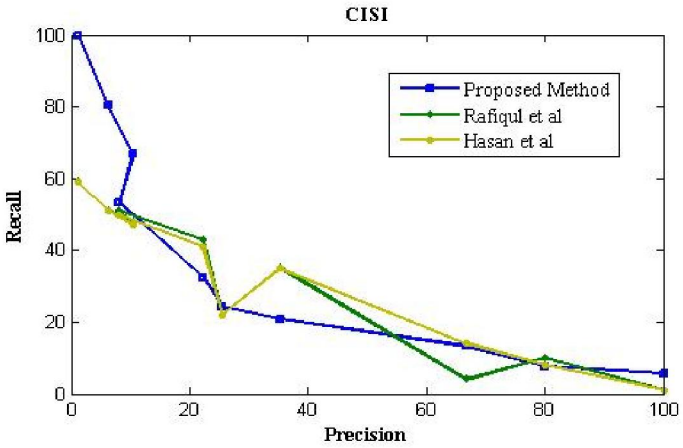

Figure 4.5: Precision-Recall for CISI

We have also experimented over small sized ADI corpus and also found a precision-recall curves which is nearly straight line, means it has a significant ratio between precision and recall. This curve is represented in the following graph Figure 4.6.

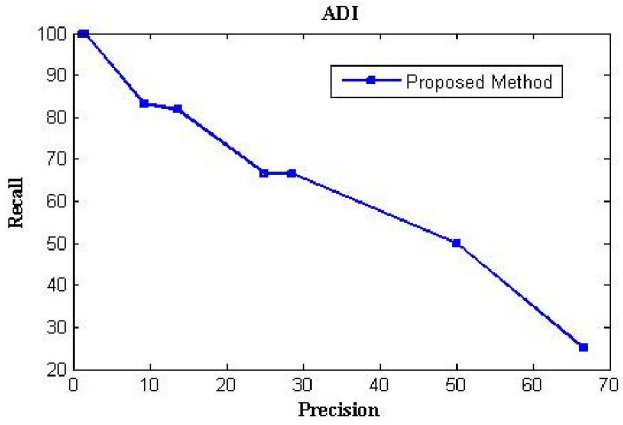

Figure 4.6: Precision-Recall for ADI

For MED corpus which contains at 1033 documents and 30 queries, we also get another performance curve as stated in the following Figure 4.7

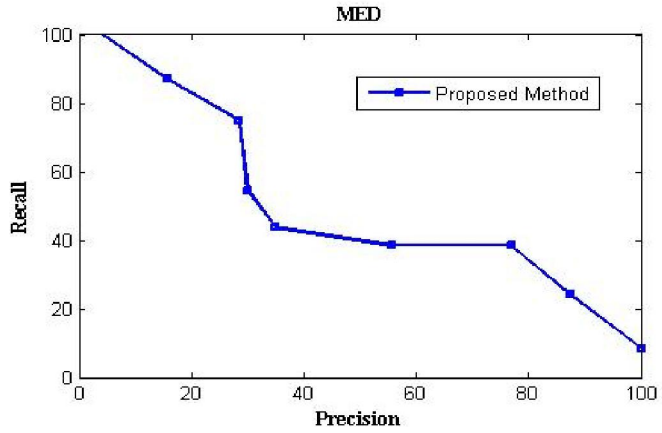

Figure 4.7: Precision-Recall for MED

From the experiments of the above corpuses we can say that our system has average higher recall rate than the existing methods but has a lower precision rate. This is often useful when the system will be allowed higher retrieval rate.

We have tested our system against those well-known corpuses and found average precision for them for different window sizes. We observed that average precision for those corpuses remain almost fixed that means it is a bit less responsive to different window sizes especially in case of CRAN corpus.

Table 7: Average precision of corpuses

Window Size	**Mean Average Precision**							
	CRAN			**MED**		**CISI**		
	Blanco *et al.* [1]	**Rafiqul *et al.* [5]**	**Our Method**	**Chuang *et al.*[13]**	**Our Method**	**Blanco *et al.* [1]**	**Rafiqul *et al.* [5]**	**Our Method**
2	0.1495	0.1511	0.1738	0.5055	0.5524	0.2634	0.2731	0.2577
4	0.1417	0.1528	0.1702	0.5259	0.5524	0.2698	0.2147	0.2231
5	0.1235	0.1498	0.1654	0.4892	0.5129	0.2761	0.1935	0.1955
9	0.1358	0.1397	0.1654	0.4989	0.5106	0.2751	0.1942	0.2257
12	0.1237	0.1369	0.1634	0.4562	0.4985	0.2521	0.1911	0.2232
14	0.1526	0.1595	0.1654	0.4283	0.4876	0.2654	0.2003	0.1955
15	0.1236	0.1461	0.1652	0.4020	0.4563	0.2745	0.1971	0.2259

CHAPTER V

Concluding Remarks

5.1 Conclusion

As current information retrieval systems are emphasizing on effectiveness of the system rather than time complexity because of the availability of powerful machines. Information retrieval itself is a probabilistic system and it is not guaranteed to have same performance level all the time. It depends on various factors like term weighting algorithm, query representation by the user, size and type of corpus, machine speed, memory etc. Our system has provided positive changes performance evaluation in case of corpus like CRAN, MED, CISI etc according to data available to us. During performance evaluation we faced the problems of data missing very much. Most of the authors didn't represent their results in suitable forms those can be used for the evaluation of our system. So often there is a chance of misinterpretation of data, erroneous data, wrong results and we had to work those non-pleasant environment.

5.2 Limitation

The main problem of our system is that the trained weight assigned to our query term is questionable. We didn't verify whether the query term weight really reflects the term's importance or not. We proposed the modified equation to calculate idf, it still reflects the global contribution of a term within the corpus in an estimation basis. We found our system a bit non-responsive towards different window sizes of document graph.

5.3 Recommendation

In future works, we recommend to apply some heuristic approach towards query term weighting as well as query term matching rather than simple formal approach and we think this will lead to more perfection in term weighting and information retrieval. We also recommend to apply these ideas to other sectors of information retrieval like keyword extraction, sentence extraction, word-sense disambiguation etc.

References

1. Roi Blanco, Christina Lioma "Random Walk Term Weighting for Information Retrieval", SIGIR, ACM, Amsterdam, Netherlands' 07 July 23-27, 2007.

2. Samer Hassan, Rada Mihalcea and Carmen Banea "Random Walk Term Weighting for Improved Text Classification". In Proceedings of TextGraphs: 2nd Workshop on Graph Based Method for Natural Language Processing, ACL 53-60, 2006.

3. Rada Mihalcea and Paul Tarau. "TextRank: Bringing Order into Texts", In Proceedings of Empirical Methods in Natural Language Processing. ACL 401-411, 2006.

4. Yue-Heng Sun, Pi-Lian He, Zhi-Gang Chen "An Improved Term Weighting Scheme for Vector Space Model", Proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai, August 26-29, 2004.

5. Md. Rafiqul Islam, Buddha Dev Sarker, Md. Rakibul Islam. "An Effective Term Weighting Method Using Random Walk Model for Information Retrieval". International Conference on Computer & Communication Engineering (ICCCE08), Istana Hotel, Kuala Lumpur, Malaysia, (13-15) May, 2008, Paper ID: 709.
6. Sarah Friedman "Information Retrieval Using Long Queries". 2001 CUN USA.

7. Van Rijsbergen C.J, "Information Retrieval", Butterworths, 1979.

8. L. Page, S. and Brin, R. "The Anatomy of a Large-scale Hyper textual Web Search Engine". Technical report, Stanford Digital Library Technologies Project, 1998.

9. Lancaster F.W., "Information Retrieval System: Characteristics, Testing & Evaluation", Wiley, New York (1968)

10. ftp://ftp.cs.cornell.edu/pub/smart

11. Christopher D. Manning, "An Introduction to Information Retrieval", Cambridge University, England.

12. http://ir.dcs.gla.ac.uk/terrier

13. Huei Chuang, Dik. L. Lee, Kent Seamons "Document Ranking and the vector-space Model", IEEE conference, March/ April, 1997.

14. Buddha Dev Sarker, Md. Rakibul Islam, "An Effective Term Weighting Method Using Random Walk Model for Information Retrieval", undergraduate thesis, 2008. Computer Science & Engineering Discipline, Khulna University, Khulna.