Dedication

In the Name of Almighty Allah
The Most Beneficent and Most Merciful

To my;
Caring Mother, Considerate Teachers
&
All those who contributed towards my future

Acknowledgements

All praise to the Almighty Allah, the most Merciful and the most gracious one. Without whose help and blessings, I would not have been able to complete this research. Many thanks to my project supervisor, Dr. Aasia Khanam, whose constant motivation, unflagging efforts and uninvolved words of wisdom ever proved a lighthouse for me; it was earnestly felt whenever we swayed. Despite his never ending assignments of university management, student counselling, project supervision and teaching, he did never mind whenever we went for an advice, within or without the time slot allocated for us.

Acknowledgement is also due to my teachers for dedicatedly instilling and imparting enlightenment to me during the course of studies and afterwards for our project. I am also very thankful to my parents for their tacit and avowed support, patience and understanding.

I would like to thank my friends who gave me confidence to face the difficulties of life. They all gave me good company and everlasting memories. Especial thanks to my friends Mr. Asif Sohail, Malik Khurram, Farhan Khan and Ayesha Tahir.

Abstract

A novel approach to informational retrieval from document collections is presented. The approach strikes a balance between Boolean models and vector models of information by employing fuzzy for determining similarity between user query and the searched document. The fuzzy approach is ideally suited for the ranking of documents and enhancing exact match based search with synonym based search. Experimental results show that the fuzzy inference system for in-formation retrieval is the best system to enhance the retrieval performance both in terms of precision and recall rates

Table of Content

List of Figures

List of Tables

SECTION I

INTRODUCTION

1 INTRODUCTION

Information Retrieval (IR) deals with the representation and access to information items. The representation and organization of items should provide the user with comfortable access to data for his concern.

1.1 Motivation

In Information Retrieval scenario the aim of data retrieval is to find out what key words from the user query exists in documents but sufficiently not enough in satisfying the information needs of a user. Actually the end user is interested only in information rather on data of a user query present in document. Natural language text that is dealt by information retrieval, this text is always semantically ambiguous and not well structured. The document’s content in a collection must be understood by an information retrieval system and are ranked for user query according to its degree of relevance. The extraction of information from document and getting to know its relevance is a difficult process. The core of the information retrieval is its conception. The aim of information retrieval process is to retrieve all relevant documents to the user query as well as trying to minimize the retrieval of non-relevant documents to the user query.

The relevant information retrieval process influenced both by logical view of the document as well as user task implemented by retrieval system.

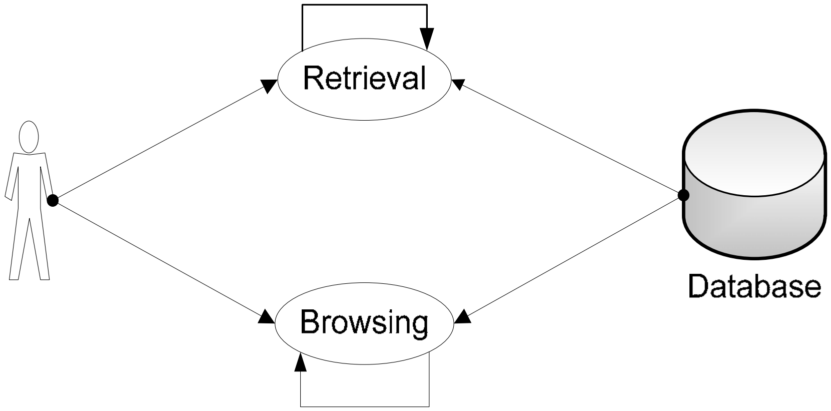


Figure 1-1 : Interaction of the user with the retrieval system through distinct tasks

1.2 User Task

The information need expressed by a user query has to be translated by the user of an information retrieval system in the language given by an IR system. In an IR system, set of keywords are specified in terms that delivers the meaning of information need [1].

1.3 Logical View of Documents

Documents in IR systems are represented by the keywords that represent logical view of document. The keywords can either be given by the text of document directly and can be provided by the specialist. In either case, whether the keywords are provided by the human or extracted from the documents, It given an illusion of document's logical view. [1]

If full text of document is indexed, the cost of using it increased and it will become more expansive to use. In many systems, the transformations or text operations are applied by using stemming, elimination of stop words and noun group identifications to improve the efficiency of the systems [1].

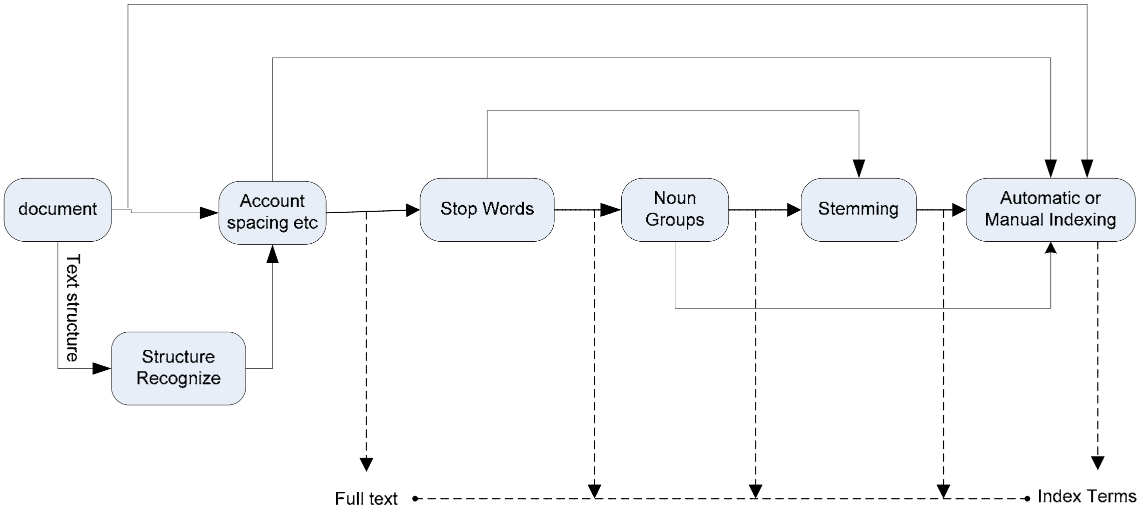


Figure 1- 1: Logical view of a document from full text to set of index terms

One of the major concepts in information retrieval is the concept of "Relevance". That can only be determined by the user by using his judgments. Many complicated factors are involved in order to judge the relevance of document. To estimate the relevance of the document, the Information Retrieval System (IRS) must be based on some model that offers a reliable illustration of both documents and user information need. Most of the existing IRS's and search engines present a very simple model of IR; a model that compromised the effectiveness at the cost of efficiency. An important aspect that influences the efficiency of IRS's depends on how the documents are represented. The logical view of the document after the extraction of keywords and term weighting can become a simpler approach. The systems made for document's representation are known as Information Storage System. [18]

1.4 **IR Challenges**

Three major issues have to be considered in coming days by IR Research[2].

1. People will face several problems for the retrieval of their relevant information, irrespective of high interactivity. What technique will permit the retrieval of quality information in fast going world?
2. The demand of faster response has become increasing with an increasing requirement to access. So, what technique will allow quicker response time of query and giving the facility for fast indexing?
3. The interaction of user with the system affects the quality of retrieving information. So how will the deployment and design of new strategies of information retrieval be affected with good knowledge of user activities?

The present thesis aims to address the first two challenges by applying soft computing technique of fuzzy sets and fuzzy logic.

1.5 **Problem Statement**

The aim of the research was "To build Fuzzy Inference System in order to score the documents in such a way that most relevant documents will get higher score against the user's information need" Relevant documents are then fetched on the basis of these scores.

1.6 Thesis Overview

Chapter 2 discusses the Information Retrieval (IR) process i.e. the text operations used to retrieve the documents. We discuss the existing information retrieval models and then see how these models are categorized.

In chapter 3 we provide an overview and description of fuzzy logic. We explained the core concepts of fuzzy logic in this chapter. At the end of this chapter we discuss why to use and why not to use fuzzy logic.

Chapter 4 discusses Design and Implementation of proposed scheme.

Chapter 5 is specifically designed for Results and Evaluation of the proposed schemes. At the start of this chapter we will explain the basics of testing metrics like precision, recall, E-measure, F-measure and Fallout. On the basis of these metrics we compare our proposed system with existing systems.

SECTION II

BACKGROUND

2 Background

2.1 The Retrieval Process

Before the retrieval process is started it is necessary to define a text database that specifies following

1. What documents will be used.
2. What operations on the text will be performed. The logical view of the documents is generated from the original documents by applying text operations.
3. The text model (The structure of the text and items to be retrieved)

The retrieval process in IRS starts with the indexing of documents database. The same text operations on documents are also applied to the information need supplied by the user. The user need is transformed for the system representation by applying the query operation on original query. Retrieved results of documents are obtained after processing of query. Indexing structure makes it possible for fast query processing [3].

The documents retrieved are ranked according to its relevance before the user gets the result. The ranked documents are then examined by the user for his required information [6].

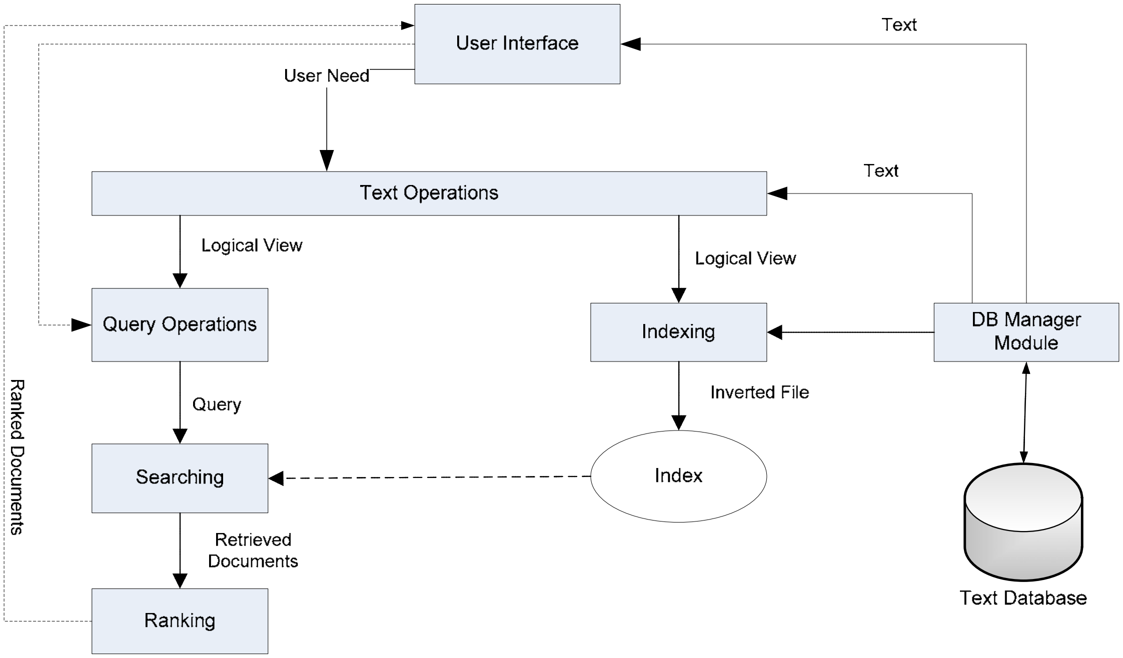


Figure 2-1: The process of retrieving information

2.2 Modelling

An index is set of words or simply a word in document. The fundamental idea of index term is that semantic of document and the The set of index terms expresses the information need of the user. Too many semantics of documents are lost from user request because of this considerable oversimplification. Further matching between each document and the user request is attempted in this very imprecise space of index terms. Thus the retrieved documents in response to a user request are frequently irrelevant [1].

In order of determine the relevant or irrelevant documents is the major issue regarding the information retrieval process, which is dependant of ranking of documents. So ranking of documents is the core of IR process.

Distinct set of documents yields distinct IR models. The IR model predicts relevance of document.

2.3 Taxonomy of Information Retrieval Models

There are three famous model in classical theory of information retrieval, named as Boolean, Vector and Probabilistic. Documents is either relevant or relevant in Boolean model. In VSM, vectors represents the documents and user queries. Therefore its an algebraic model. Query and documents representation are modelled in the framework of probabilistic model that is build on probability theory. So the model is probabilistic as its name implies.

In our discussion our scope will be limited to fuzzy set model and some of the rest will only be discussed for comparison with our proposed model.

Here it is significant to differentiation of ranking and filtering of documents. In ranking, the documents are numbered only according to its relevance with user's query term while in filtering a user profile is defined in which his interests are recorded and documents that are considered relevant will be filtered. In filtering, documents with the ranking above certain threshold will be selected while the rest will be discarded [2].

2.3.1 Formal Characterization of IR Models

To build model we first have to think the representation for the documents and for the user information need. Given these representations, we then conceive the framework in which they can model. This framework should also provide initiation for constructing a ranking function. For instance, in classic Boolean model, the framework is composed of set of documents and the standard operations on sets.

2.3.2 Classic Information Retrieval

The classic models consideration in information retrieval is described by index terms that are set of keywords. The semantics of index terms helps the IR system to remember the main theme of the document. Thus index terms are used to index and summarize the contents of the document's main features. As the nouns in English grammar have self contained meaning and so easier to understand, so nouns are normally used to represent the index terms. Although, the extracted index terms from the document collection are all the distinct words present in the collection, in which case document logical view is full text [1].

Some index terms are vague and considered important as compared to other to summarize the contents of documents. The term that appears in almost every document in a collection is considered less important while the terms that appears in very few documents tell a lot and its importance become increases, because it narrows down when used to describe document contents. Every index terms in a document collection is assigned the numerical weight in order to capture this effect [7].

Let d_j is a document, k_i is an index term, the weight assigned to the pair (k_i,d_j) is $W_{i,j} \geq 0$. The significance of index terms is quantified by these weights that better explains the semantic content of the document.

2.3.2.1 Definition:

Let the system has t index terms and k_i is specific index term [1].

$$K = \{k_1, k_2, k_3, \ldots\ldots., k_t\}$$

Each index term in document is assigned a weight $W_{i,j} > 0$. If index term is not presents in document then its weight will be zero.

$\vec{dj}$ is document vector for the weights of all the index terms.

$$\vec{dj} = (W_{1,j}, W_{2,j}, W_{3,j}, W_{4,j}, W_{5,j}, \ldots., W_{t,j})$$

The weight that is related to index term k_i is returned by the function gi in any t-dimensional vector $g_i(\vec{d}_j) = Wi, j$.

The weights of the index terms in a document are normally mutually independent. So these index terms are uncorrelated in document. Consider the term computer and networks are used to index a given document which covers the area of computer networks. The appearance of one of these two words attracts the appearance of the other.

Thus these two words are correlated and their weights could reflect this correlation. To enhance the processing speed of document's ranking computation, The computation of the weights of index terms is strongly simplified by the use of mutual independence of index terms[1].

The above discussion provides support for discussing the three classic IR models, that are Boolean retrieval, vector space and probabilistic models.

2.3.2.2 Boolean Model

The simplest model in which the precise nature queries are specified through Boolean expressions.

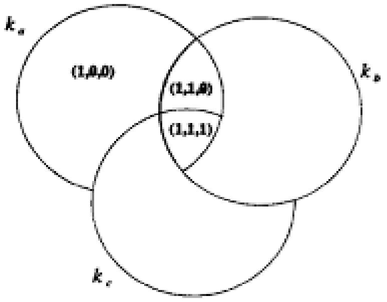


Figure 2-2: The three conjunctive components for the query

The deficiency in this model is that a document to be retrieved is either relevant or non-relevant because of its Boolean nature. Second, In general, Boolean expressions are not easier to be translated from the information need, but these expressions possesses precise semantics. So it is much more data retrieval model instead of information retrieval model [3].

The index term either exists or not exists in document when processed by boolean model, so binary weights will be considered, ie. $Wi, j \in \{0,1\}$. The index terms composes the query q, joint using three connectives OR, NOT, AND. The disjunction of conjunctive vectors is essentially represented by Boolean expression of the query (DNF Disjunctive Normal Form). For example, the query $[q = k_a \wedge (k_b \vee k_c)]$can be written in disjunctive normal form as [$\vec{q}_{ndf} = (1,1,1) \vee (1,1,0) \vee (1,0,0)$]. Where every component related to the record (k_a, k_b, k_c) is a vector of binary weights. So the conjunctive components of $\vec{q}_{ndf}$ are binary weighted vectors. Figure 2.2, illustrates that query q is represented by three conjunctive components.

Definition: Binary weights will be assigned to all index terms of a document. Boolean expression expresses the query q. Let the query q has its disjunctive normal form $\vec{q}_{dnf}$. Let the conjunctive component of $\vec{q}_{ndf}$ is $\vec{q}_{cc}$. The similarity of query q with document dj is expressed as

$$sim(d_j, q) = \begin{Bmatrix} 1 \ if \ \exists \ \overrightarrow{(q_{cc}} \in \vec{q}_{dnf}) \ \wedge (\vee_{k_i}, g_i(d_j) = g_i(\vec{q}_{cc})) \\ 0 \ otherwise \end{Bmatrix}$$

If sim(dj,q)=1 then the document dj is relevant to query q is predicted by Boolean retrieval model otherwise the document is not relevant will be predicted.

The document can either be relevant or non-relevant is the prediction of Boolean model. Partial matching of the query is not supported by Boolean model. For example, the document vector for dj is $\vec{d}_j = (0,1,0)$. The index term kb is present in the document dj but not relevant to the query $[q = k_a \wedge (k_b \vee k_c)]$. [1]

2.3.3 Vector Model

The model supports partial matching by assigning non binary values to index terms [8]. The relevance of document and query is based on these non binary weights. The ranked retrieved documents from vector model are more accurate and precise as compared to the retrieved documents from Boolean retrieval model [2].

Definition: It assumes the positive and non binary weights for index terms in the documents as well as in queries. $W_{i,j}$ and $W_{i,q}$ are all greater than zero. Let $\vec{q}$ be the query vector.

$$\vec{q} = (w_{1,q}, w_{2,q}, w_{3,q}, \ldots\ldots, w_{t,q})$$

The vector of document is represented by $d_j = (W_{1,j}, W_{2,j}, W_{3,j}, \ldots\ldots, W_{t,j})$

As a result, the user query q and a document dj can be represented in t-dimensional space as shown in Figure 2.3

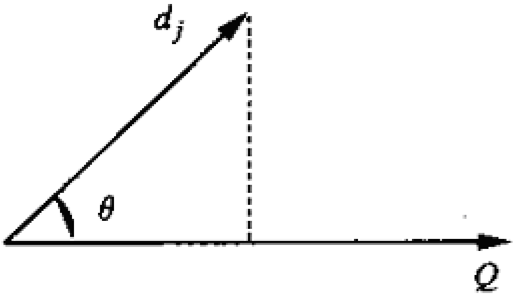


Figure 2-3 : The cosine of θ is adopted as sim(dj,q)

In vector model, the similarity of user query q with document d_j is evaluated through the correlation of vectors $\vec{q}$ and $\vec{d}_j$. The cosine of angles among these two vectors is quantified by this correlation.

$$sim(d_j, q) = \frac{d_j \cdot \vec{q}}{|\vec{d}_j| \times |\vec{q}|}$$

$$sim(d_j, q) = \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2 \times \sqrt{\sum_{j=1}^{t} w_{i,q}^2}}}$$

Where $|\vec{d}_j|$ is the norm for document vector and $|\vec{q}|$ is the norm for query vector. As $|\vec{q}|$ is same for all documents so it doesn't change the ranking. Instead of predicting whether the document is relevant or not it ranks the document according to its similarity with query q. one can establish threshold on sim(d_j ,q) and retrieve the documents with a degree of similarity above that threshold[2].

To compute ranking, we must have to identify how to obtain the weights of index terms. There are multiple ways to calculate the weights of index terms. Here we will discuss clustering technique of document indexing. Given the vague explanation of set A and collection of objects C. The aim is to separate the collection of objects C into two sets by using the clustering algorithm. One of the set contains the members that belongs to set A, while the other set contains the member that do not belong to set A. Clustering algorithms mainly disputes two main issue; The first one requires to identify the features that better describes the objects of set A. another need to determine the features that better differentiate between the elements in Set A with elements in collection C [1].

To quantify the dissimilarity of Inter-cluster, the inverse of frequency of the term k_i needs to be measured with in entire collection among all documents. This inverse of frequency for given term is said to be inverse document frequency of *idf*. The idea in using this approach is to increase the importance of those terms that are useful and decreases the importance of useless terms in order to distinguish among relevant and non-relevant documents.

Definition: Let the system contains N number of total documents and there are n_i documents in the system that contains the index term k_i. Let $freq_{i,j}$ is the frequency of document d_j of term k_i. Then, the normalized frequency $f_{i,j}$ of term k_i in document d_j is given by

$$f_{i,j} = \frac{freq_{i,j}}{\max freq_{i,j}}$$

Where the maximum is computed over all terms which are mentioned in the text of document d_j . Let idf_i is inverse document frequency for k_i , be given by

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

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

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

or by a variation of this formulae such term-weighting strategies are called tf-idf schemes.

2.3.4 Probabilistic Model

Also called Binary Independence Retrieval (BIR) model [1]. The set of relevant documents for user query q is said to be an ideal answer set. We can think of the querying process as the features of an ideal answer set that can be specified through that process. At query time, these properties are unknown; an initial guess has to be made as an effort to know these properties. The initial probabilities of an ideal answer set are generated through initial guess. The probabilities of an ideal answer set can be improved with continuous interaction of the user. The user examines for relevant documents from the set of top k ranked documents. The ideal answer set will become more refined with such repeated interaction of the user [2].

Following are the important assumptions that are to be made for probabilistic model. Given a user query q and a document d_j in the collection, the probabilistic model tries estimating probabilities that how much is the document is relevant to the user. The subset of documents that user prefer to be relevant to the query will be represented by R while the rest of documents not in R are assumed non relevant [1].

Given a query q, the probabilistic model assigns to each document d_j as a measure of its similarity to the query, the ration P(d_j relevant to q) / (P(d_j non-relevant to q) which computes the odds of the document d_j being relevant to the query q. taking the odds of document as rank minimizes the probability of an erroneous judgment.

Definition: In probabilistic model, binary weights are assigned to the weight variable of index terms. $W_{i,j}$ ε {0,1} , $W_{i,q}$ ε {0,1}. The index terms set contains the subset for query q. Let the relevant documents initially knows comprises the set R. Let $\overline{R}$ is the set of non relevant documents (The complement of R). The probability of relevant document dj to query q has the probability $P(R \mid \vec{d}_j)$ and the probability of non-relevant document dj to the query q is $P(\overline{R} \mid \vec{d}_j)$. Let sim(dj,q) is the similarity of query q with the document dj is defined by ratio [1]

$$sim(d_j, q) = \frac{P(R|\vec{d}_j)}{P(\bar{R}|\vec{d}_j)}$$

$$sim(d_j, q) = \frac{P(\vec{d}_j|R) \times P(R)}{P(\vec{d}_j|\bar{R}) \times P(\bar{R})}$$

Given the set R of relevant documents, a document dj randomly selected has the probability $P(\vec{d}_j \mid R)$. More on this, If the randomly selected document is relevant, It has the probability P(R). Since $P(R)$ and $P(\overline{R})$ are same for all documents in collection so we can write [1]

$$sim(d_j, q) \approx \frac{P(\overrightarrow{d_j}|R)}{P(d_j|\bar{R})}$$

2.3.5 Fuzzy Set Model

An alternative set theoretic model for Classic Boolean model that represents document and queries through sets of index terms that are partially related. So the matching of documents becomes approximate [4]. This can be modelled by defining a fuzzy set for each query.

2.3.5.1 Fuzzy Set theory

In fuzzy set theory, the membership value of fuzzy variable is mapped within a fuzzy set.

Definition: Let A and B are two fuzzy sets from Universe of Discourse U.

$$\mu_{\bar{A}}(u) = 1 - \mu_{A}(u)$$

$$\mu_{A\cup B} = \max\left(\mu_A(u), \mu_B(u)\right)$$

$$\mu_{A\cap B}(u) = \min\left(\mu_A(u), \mu_B(u)\right)$$

Fuzzy sets are useful for representing vagueness and imprecision and can be applied to various domains. Here our domain is restricted to information retrieval.

2.3.5.2 Fuzzy Information Retrieval

An approach to model the information retrieval process is to adopt thesaurus. The thesaurus can also be used to model information retrieval problem in terms of fuzzy sets [1]. A normalized correlation factor $c_{i,l}$ between two terms k_i and k_l an be defined by

$$c_{i,l} = \frac{n_{i,l}}{n_i + n_l - n_{i,l}}$$

The fuzzy set definition for the terms correlation matrix for index term k_i. Let $\mu_{i,j}$ be the degree of memebership for document d_j

$$\mu_{i,j} = \mathbf{1} - \prod_{k_l \in d_j} (1 - c_{i,l})$$

Which computes an algebraic sum over all terms in document d_j .

The user states his information need by providing a Boolean like query expression. This query is then converted to its DNF (Disjunctive Normal Form).

$$\vec{q}_{dnf} = cc_1 \vee cc_2 \vee cc_3 \ldots \vee cc_p$$

Where p is the number of conjunctive components of $\vec{q}_{dnf}$. The procedure to compute the documents relevant to a query is similar with the procedure adopted by classic Boolean retrieval model.

SECTION III

FUZZY LOGIC

3 Fuzzy Logic

3.1 Overview of Fuzzy Concepts

In real world lot of fuzzy knowledge exists i.e. uncertain, vague, inexact, imprecise, ambiguous or probabilistic. Reasoning and human perception usually contain fuzzy information probably began from imprecise human observation. It is hard to answer in logic based system because they do not have any exact answer. Only the humans can give the practical answers which are most likely to be true. Expert systems can give such answers with the description of their reality level. Different expert opinions, unreliable and incomplete information is easy to be handled by expert systems [19].

The fuzzy logic maps the input space with output space with the use of the rules that is English like if-then statement. All the rules are evaluated at the same time and their order is not important [19]. e.g. If height is tall, The ranges of expected heights need to be defined together with what is meant by the term tall.

3.2 Description of Fuzzy Logic

The applications of fuzzy logic are increasing drastically by numbers and varieties in recent years. There are countless applications for fuzzy logic and these applications range from predicting genetic traits, medical diagnosis, auto-focus on cameras, temperature control, decision support systems and Natural Language Processing (NLP) etc [4].

There are two different meaning of fuzzy logic. Fuzzy logic is an expansion of multi-valued logic and is a logical system in a narrow sense. Although fuzzy logic is approximately identical to the fuzzy set theory in a wider sense, it is associated with the classes of objects that has degree of memebership in it and do not have clear-cut boundaries. From this viewpoint in the narrow sense, fuzzy logic differs both in substance and concept from conventional multi-valued systems[4].

Linguistic variables are major concepts used in fuzzy set theory that will be explained in detail in upcoming sections, i.e. a variable whose values are English like words instead of crisp numbers. The ongoing tendency in visibility tells us about the use of fuzzy logic in genetic algorithms, neuro-computing. In general, genetic algorithms, fuzzy logic, neuro-computing be seen like the major component of so called soft computing.

Soft computing has multiple methodologies combinations, among them the neuro-computing and fuzzy logic has the highest visibility at this occasion. A useful approach designed by Dr. Roger Jang for this objective is called Adaptive Neuro-Fuzzy Inference System (ANFIS) [4].

Here it is important to describe some major concepts used in fuzzy logic like what is fuzziness, what are variables and fuzzy values etc.

3.2.1 Fuzziness

Fuzziness exists where the edges of pieces of information is not clear. For example, terms like tall, young, high or good are all fuzzy. There are no fixed numerical values that define the term tall when defining the fuzzy variable height. For some people, 5.5 feet is tall and for others 6 feet is the tall height. The concept tall has no clear boundary. Height 7 feet is definitely tall and height 2 feet is definitely a short height. Although height 5.5 feet is considered to be tall has some possibility and generally it is considered in context in which it is dependent. Their may be some possibility that the height is to be a tall and also have some possibility to be short at the same time. The interpretation of such kind of information depends on the concept of fuzzy set theory. Apart from the fact classical set theory where one deal with things whose membership value with in a set is clearly defined in two valued logic either true or false, but in fuzzy set theory membership value for an element can be partial in a set, i.e. an element that fits in a set with some value of membership.

$$\mu_A : U \varepsilon [0,1]$$

This number $\mu A(x)$ represents the member value of x in the fuzzy set A (with 0 means that x has no membership in a set and 1 means that x has full membership value in the set while values between 0 and 1 represents that x has partial membership within a set). For example, the fuzzy set of term young is defined in table 3.1.

Table 3-1: Membership Values for Fuzzy Terms

Fuzzy Term tall	
Height	Grade of Membership
7	1.0
6	0.8
5	0.6
4	0.4
3	0.2
2	0.0

$\mu_{tall}(7) = 1, \mu_{tall}(6) = 0.8, \ldots, \mu_{tall}(2) = 0$ [19]

Fuzzy sets, fuzzy variables and fuzzy values all represents the fuzzy concepts.

3.2.2 Fuzzy Sets

The real numbers (x_i) on membership values (u_i) is useful for mapping a fuzzy set in the range of 0 and 1. The set of pairs u_i/x_i is characterized through a fuzzy set, where the real number x_i has the membership value u_i. These set of values can be represented as $\{u_1/x_1\ u_2/x_2 \ldots u_n/x_n\}$. There is an increasing order in the set for the values of x $\{x_1 <= x_2 <= \ldots <= x_n\}$. Lowest membership value will be assigned to all values less than x_1 and the highest membership value will be assigned to all values greater than x_n. Value that exists in between consecutive number x_i and x_{i+1} will be assigned the membership value between two consecutive points that exists on the line. [19]

Normally the fuzzy set having pairs (x,y) is characterized in the world of fuzzy logic by y/x. As the ordered pair x,y is reversed so it creates some confusion in understanding, but u/x is easier to understand by giving the impression of the membership value of u at x.

Figure 3.1 is a triangular shaped fuzzy set that is the representation of set {0.0/0.3 1.0/0.5 0.0/0.7}. The real number line covers all values in the fuzzy set that is the very compact representation of a fuzzy set.

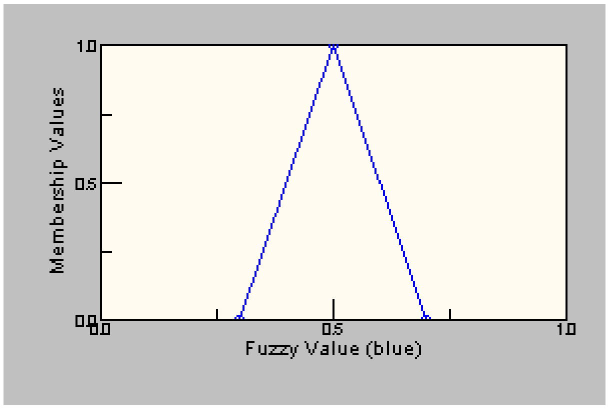


Figure 3-1 : Fuzzy Set

3.2.3 Fuzzy Variable

A fuzzy variable describes the fundamental components that are used to define a fuzzy concept. It consists of variable name (e.g. height or temperature) the units of variable (e.g. feet or Centigrade), these linguistic expressions are mapped to a fuzzy value that possesses a certain fuzzy concept such as height is very tall. [19]

Lower and upper bound are defined for each set in a universe of discourse that illustrates fuzzy variable height.

(very tall or medium) and slightly short

3.2.4 Fuzzy Value

The Fuzzy Value permits one to make a particular fuzzy concept for a known Fuzzy variable, say height. e.g. you may like to describe the concept height is very tall. suppose that we have a Fuzzy Variable for height with the term tall defined, we just form the Fuzzy Value by specifying the height Fuzzy variable and a linguistic expression (in this case very tall). The linguistic expression is parsed and a Fuzzy Set that describes the shape of this fuzzy concept is formed and saved with the Fuzzy Value. So a Fuzzy Value is a mapping of a Fuzzy Variable and a linguistic expression to define the fuzzy concept. [19]

3.2.5 Fuzzy Rules

The input values, antecedents and conclusion of a fuzzy rule is represented through three sets of fuzzy values possessed by a fuzzy rule. A rule can be written as follows: [4][19]

```
if antecedent1 and
    antecedent2 and
        ...
    antecedentn
then
    conclusion1 and
    conclusion2 and
        ...
    conclusionm
```

The buildings of the rule are antecedents; the antecedents must be true before the evaluation for the conclusion part of the rule. The actual value of the antecedents corresponds to the fuzzy rule attached with the fuzzy values input set. The execution of the rule determines the actual conclusion set by using the Fuzzy rule executor attached with the rule. Most common fuzzy inference algorithms like Mamdani are implemented by fuzzy rule executor. [4][19]

3.3 **Why Use Fuzzy Logic**

The general observations about fuzzy logic [4]:

- Its easier to master it.

- Easier mathematical concepts are required behind fuzzy logic theory. Minor complexities are involved due to its flexible approach.
- For a given system, More functionality layer can be added on existing systems. So its extendable.
- Almost everything in the world is imprecise itself in nature if looking closely at it.
- Fuzzy logic can represent non-linear functionality of any complicated system.
- You can build a fuzzy system based on training data that exactly matches the input and output data. The adaptive techniques like ANFIS (Adaptive Neuro-Fuzzy Inference Systems) make this process simpler. Fuzzy Logic Toolbox in MATLAB contains these adaptive techniques.
- Fuzzy logic makes best use of experience of people.
- In comparison with neural networks that generate opaque, impenetrable models based on training data set, fuzzy logic builds the system on top of the experience of experts who are well versed with the system.
- In most of the scenarios fuzzy logic enhance such systems to simplify their implementation.
- Fuzzy logic as the logic of natural language.
- As fuzzy logic is qualitative in nature rather than quantitative and becomes understandable and easier to use.

The last point is important and need to be discussed here. Natural language that is used by a common person on regular basis is shaping to become efficient and convenient from several years of human history. A statement of our common language represents an achievement of proficient communication.

3.4 Why not to use Fuzzy Logic

Fuzzy logic is not a solution to every problem. When should you not use fuzzy logic? The simplest argument mentioned on the start of this chapter, fuzzy logic maps the inputs to outputs. Try some other approach if not convenient with this approach. If their exists some solution already that is simpler than use it. Many applications like the controllers is doing well without the use of fuzzy logic. Although, if you have a good knowledge about fuzzy logic, you will definitely get an experience that it can be a very effective tool for dealing in proficient and quick manner with non-linearity and vagueness[4].

3.5 Fuzzy Inference Methods

Two types of methods are commonly used for fuzzy inference. Both have their own advantages. Which method should be used for given system, it depends on existing scenario. Although comparison of both the method is given below. Mamdani system is most common and widely accepted by the researchers. In contrast to mamdani, sugeno method works well with optimization, linear and adaptive techniques and are computationally efficient.

Because sugeno method is a much precise and computationally perfect model as compared to Mamdani systems, the adaptive techniques for the development of fuzzy models makes best use of Sugeno system. The customization of the membership functions is done through these adaptive techniques according to our requirements so that the fuzzy system best represents the data.

SECTION IV

DESIGN & IMPLEMENTATION

4 Design and Implementation

Recalls from chapter 2, most of the mainstream commercial systems employ the described models for Information Retrieval; there is a growing recognition of the need to improve the performance of these systems by using improved modelling techniques. In this regard, the field of Soft Computing holds great promise to exploit the peculiar characteristics of IR domain. Traditional techniques do not adequately address the inherent imprecision and vagueness in document representation, query formulation, and document-query relevance. Fuzzy logic is a soft computing paradigm that provides adequate constructs for reasoning with a tolerance for imprecision and vagueness.

We will start this chapter by discussing the problems in existing system and then discuss in detail how these problems are answered by our proposed system. This chapter covers the history of IR algorithms and what our system suggested in improving those systems.

4.1 Problems With Existing System

The existing models discussed above suffer from some problems. Boolean retrieval models are simple to implement but not very much effective. Vector Space Model is effective but too much pre-processing and disk space is required. Our model is a hybrid, using vector space model for information retrieval and logic based boolean model for document scoring. Based on fuzzy set theory and fuzzy logic, the proposed model gives simplicity of logic based models and the performance and flexibility of vector space models. The above mentioned techniques do not cater the approaches proposed by different authors defining the same concepts. Figure 2 shows the architecture of the proposed model.

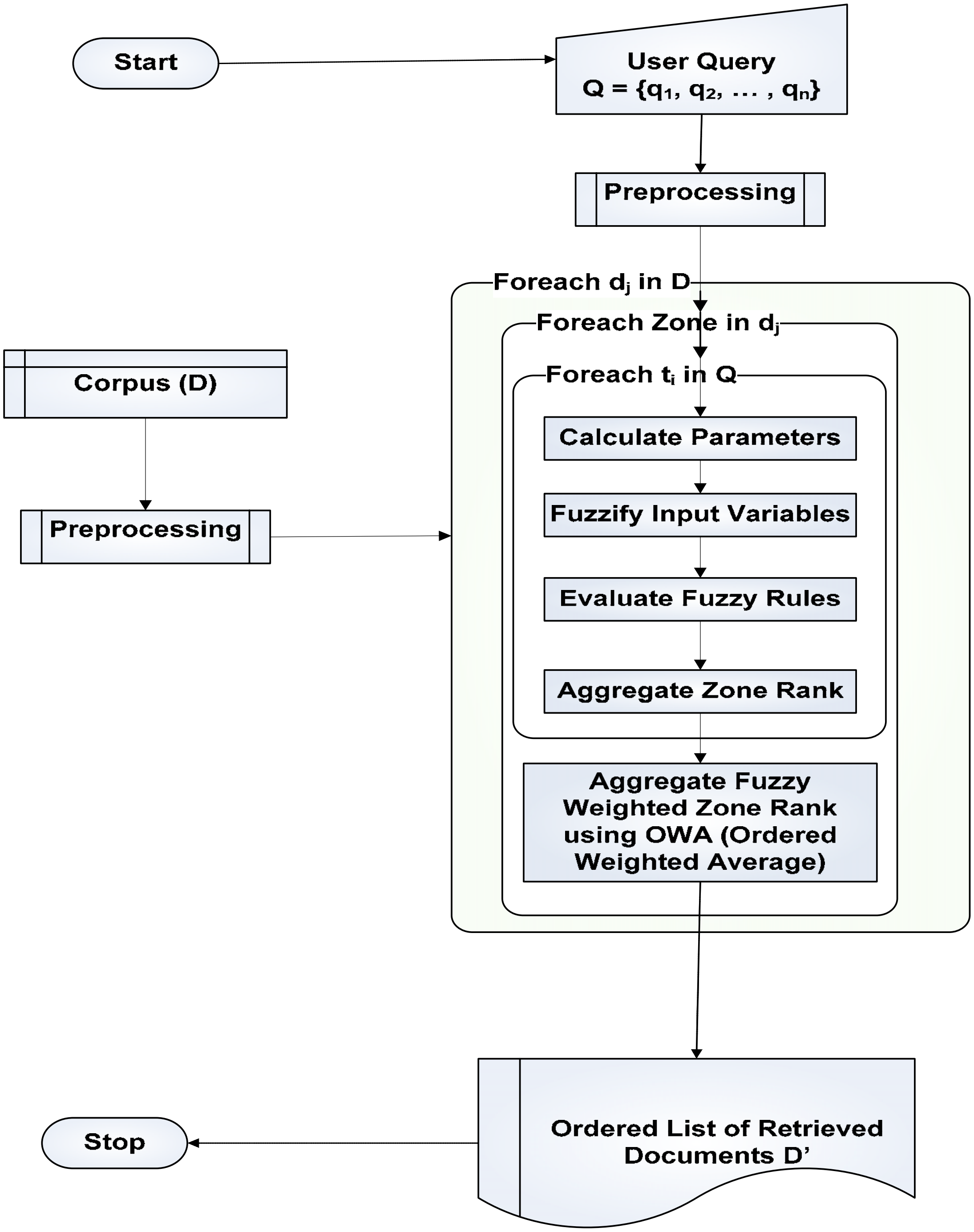


Figure 4-1: Flow Chart of Proposed Scheme

4.2 Proposed Approach

Fuzzy logic is an extension of Boolean retrieval model falls under the category of logic based model. It allows vague matching of documents with query. It can be modelled by defining a fuzzy set against each query and every document will have a degree of membership in that set [10]. Lot of work has been done in the field of information retrieval but this system includes some special features that is useful in finding most relevant documents in the system by using fuzzy inference system.

Most significant features included in this model for better relevance scoring of documents are given below that were not addressed by any existing fuzzy based inference systems for information retrieval.

4.2.1 Weighted zone scoring

Semi structured document is represented through different markup tags. These tags define the zones of document. Each zone of document has its own importance according to the relevance of document. Like the existence of query term in abstract section should have more membership value than its occurrence in Reference section.

4.2.2 Query Expansion

Semantics matching: Most of the time it is noted that most relevant documents defining the same concept as the user information needs cannot be retrieved. It is because that document is written by an author who has his own style of writing. He used different words in his document in contrast to the terms given by user in his query. To do this we form different queries by replacing the term with its multiple synonyms. In our discussion we will refer these queries as synonym queries. It is logical to give low membership values to these queries.

Phrase Searching: We are going to redefine the term proximity, we are not only interested in closeness of query terms with document but also the same order of query terms in documents as in query. As changing order of query terms lead to inverse results of what you are looking for e.g. “cat killed rat” may seems weird if its order get shuffled e.g. “rat killed cat”. But enforcing only this strict order will lead the retrieval model to Boolean retrieval model. So we suggested not only redefining the query but also redefining the term “term” in its original meaning. Previously a query word was called term, while term can be word, phrase or expression. So we’ll be using term in its original meaning. To achieve this we will make a set of queries with all possible consecutive combination of original query words. It is obvious here to give more membership value to a query having all query words in order.

4.3 Fuzzy Linguistic Variables

Various inputs and outputs are represented by fuzzy linguistic variables owing to the flexibility of this representation [13]. The input variables are tf , idf, overlap (How many terms in query occurs in document), Match (Either exact or synonym), and Zone (Title, Abstract etc). All of these inputs have fuzzy sets with Gaussian curves as membership functions.

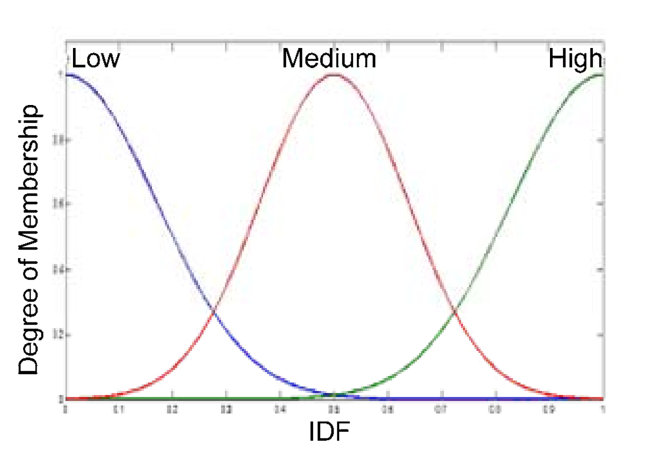


Figure 4-2 : Fuzzy Input Variable

4.4 Modeling Fuzzy Interface Systems

Sugeno and Mamdani are two types of systems that are supported. Mamdani is most widely used inference framework because of its simplicity but sugeno’s inference system gives more optimal results because the system is trained based on actual training data.

The major difference between the two methods is that the constant and linear output membership function can be used in sugeno.

We have chosen sugeno method for our FIS. The overall system diagram of our FIS is given

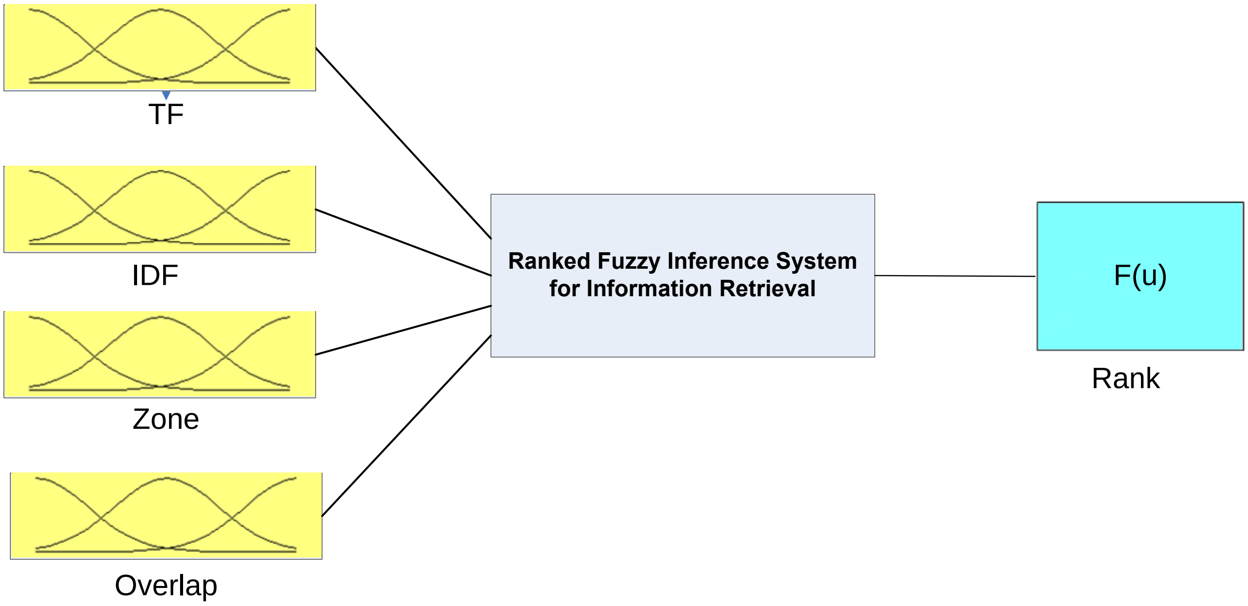


Figure 4-3 : System diagram of Proposed Scheme (ZoRFIS)

4.4.1 Fuzzy Rules and Inference

A typical rule in a Sugeno fuzzy model has the form.

If Input 1 = x and Input 2 = y, then Output is z = ax + by + c

For a zero-order Sugeno model, the output level z is a constant (a=b =0).

Following are some of the rules that are generated by ANFIS in our FIS.

- If (tf is Low) and (idf is Low) and (zone is Low) and (overlap is Low) then (output is out1mf1)

- If (tf is Low) and (idf is Medium) and (zone is Low) and (overlap is Low) then (output is out1mf7)
- If (tf is Medium) and (idf is Low) and (zone is Low) and (overlap is Low) then (output is out1mf19)
- If (tf is Medium) and (idf is High) and (zone is Low) and (overlap is Medium) then (output is out1mf32)
- If (tf is High) and (idf is Medium) and (zone is High) and (overlap is Medium) then (output is out1mf47)

4.4.2 Rules output Aggregation and Defuzzification

After all rules have been evaluated, their results are combined by the aggregation operation to obtain the final relevance rank of each document. The aggregation operation again produces a fuzzy set over a range of values. This is defuzzified to determine a crisp value from using the weighted average (wtaver) method.

4.4.3 Load Training Data

The desired input and output data for the system contained in the training data to be modelled begins this process to train the FIS by loading the training data set. The training data is represented by an array and is arranged as column vectors in which the last column represents the output data. Here we have four input columns tf, idf, zone and overlap and score is the output variable calculated as

$$\text{Score} = \text{tf} * \text{idf} * \text{zone} * \text{overlap}$$

Table 4-1 : Training Data

tf	idf	zone	overlap	Score
0.005586592	0.324119469	0.5	1.096710205	0.000992919
0.005882353	0.324119469	1	1.096710205	0.002090971
0.009009009	0.324119469	1	1.096710205	0.003202389
0.009803922	0.324119469	1	1.096710205	0.003484952
0.014388489	0.324119469	0.5	1.096710205	0.002557303
0.016304348	0.324119469	1	1.096710205	0.005795627
0.018867925	0.324119469	1	1.096710205	0.006706889

tf	idf	zone	overlap	Score
0.022222222	0.324119469	0.5	1.096710205	0.003949613
0.025641026	0.324119469	1	1.096710205	0.00911449
0.026086957	0.324119469	1	1.096710205	0.009273003
0.02739726	0.324119469	1	1.096710205	0.009738771
0.031007752	0.324119469	1	1.096710205	0.011022175
0.005555556	0.432460612	16	1.49271137	0.024815123
0.005681818	0.432460612	2	1.587962963	0.007803766
0.006451613	0.432460612	16	1.49271137	0.028817562
0.006535948	0.432460612	1	1.49271137	0.004219208
0.007407407	0.432460612	0.5	1.49271137	0.002390885
0.007407407	0.432460612	4	1.248975876	0.016003937
0.007407407	0.432460612	16	1.49271137	0.033086831
0.007692308	0.432460612	2	1.587962963	0.010565099
0.008	0.432460612	2	1.587962963	0.010987703
0.008064516	0.432460612	2	1.587962963	0.011076313
0.008695652	0.432460612	1	1.49271137	0.005613382
0.00877193	0.432460612	16	1.49271137	0.039181774
0.009345794	0.432460612	16	1.49271137	0.041745067
0.010526316	0.432460612	2	1.587962963	0.014457504
0.010582011	0.432460612	2	1.587962963	0.014533999
0.010638298	0.432460612	16	1.49271137	0.047518321
0.012048193	0.432460612	2	1.587962963	0.016547745
0.015748031	0.432460612	0.5	1.49271137	0.005082983
0.015748031	0.432460612	4	1.248975876	0.034024118
0.015748031	0.432460612	16	1.49271137	0.070342082

4.4.4 Loading initial FIS structure

After loading the training data, in next step we generated the initial structure for our fuzzy inference system using grid partitions. We have selected 3 membership functions for each of input variable and membership function type to be selected is gbellf (Generalized bell shaped function), and output membership function type selected is linear.

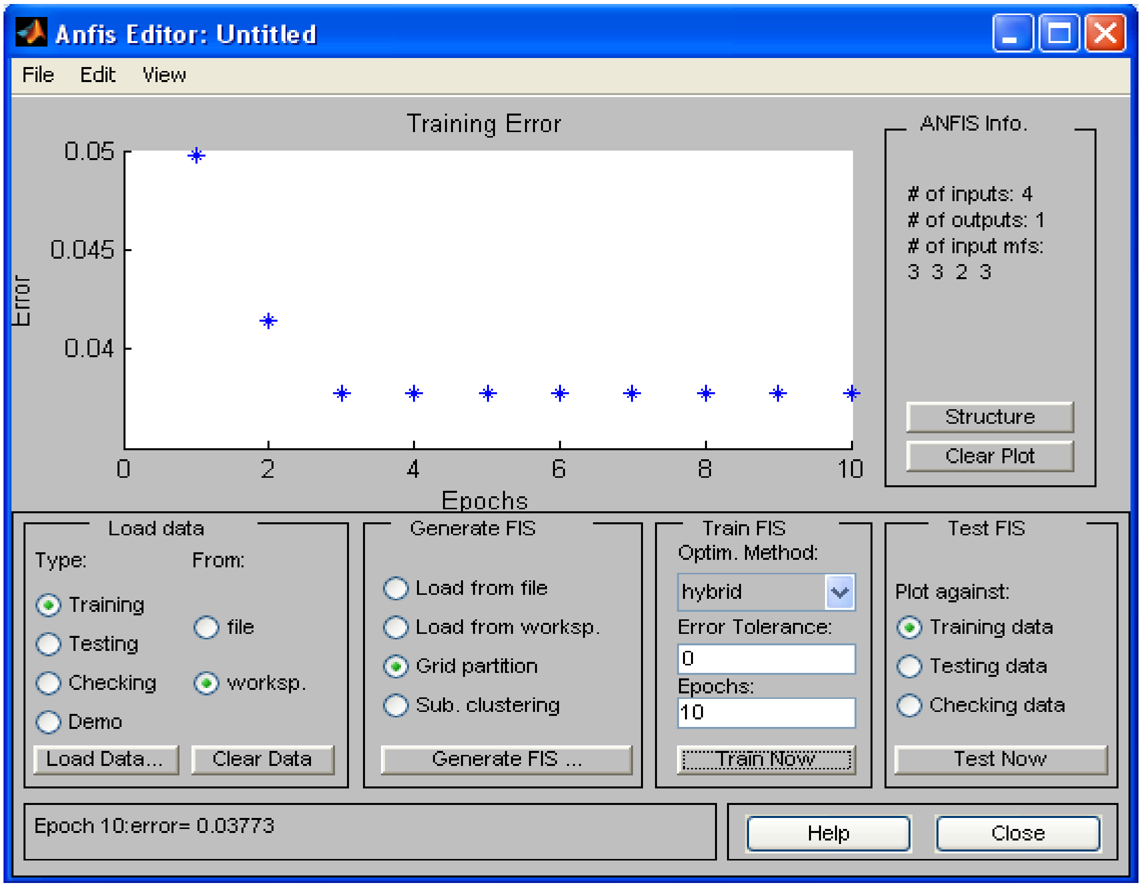
Generalized bell-shaped built-in membership function is dependent on three parameters a, b and c given by

$$f(x;a,b,c) = \frac{1}{1 + \left|\frac{x-c}{a}\right|^{2b}}$$

parameter b in above equation is generally positive. The curve centre is located through parameter c. the second argument of bell shaped function gbellmf is params that accepts the parameters in the form of vector.

4.4.5 Train FIS

When training data is loaded and the initial structure is generated for our fuzzy inference sytem, we trained our FIS by choosing the hybrid optimization method, it is the combination of least square and back propagation gradient descent method. Train the system by selecting the required no of epochs. Figure 4.4 shows training error plot after training of FIS.



ANFIS Info.

# of inputs: 4
# of outputs: 1
# of input mfs:
3 3 2 3

Load data

Type: ● Training ○ Testing ○ Checking ○ Demo

From: ○ file ● worksp.

Generate FIS

○ Load from file
○ Load from worksp.
● Grid partition
○ Sub. clustering

Train FIS

Optim. Method: hybrid

Error Tolerance: 0

Epochs: 10

Test FIS

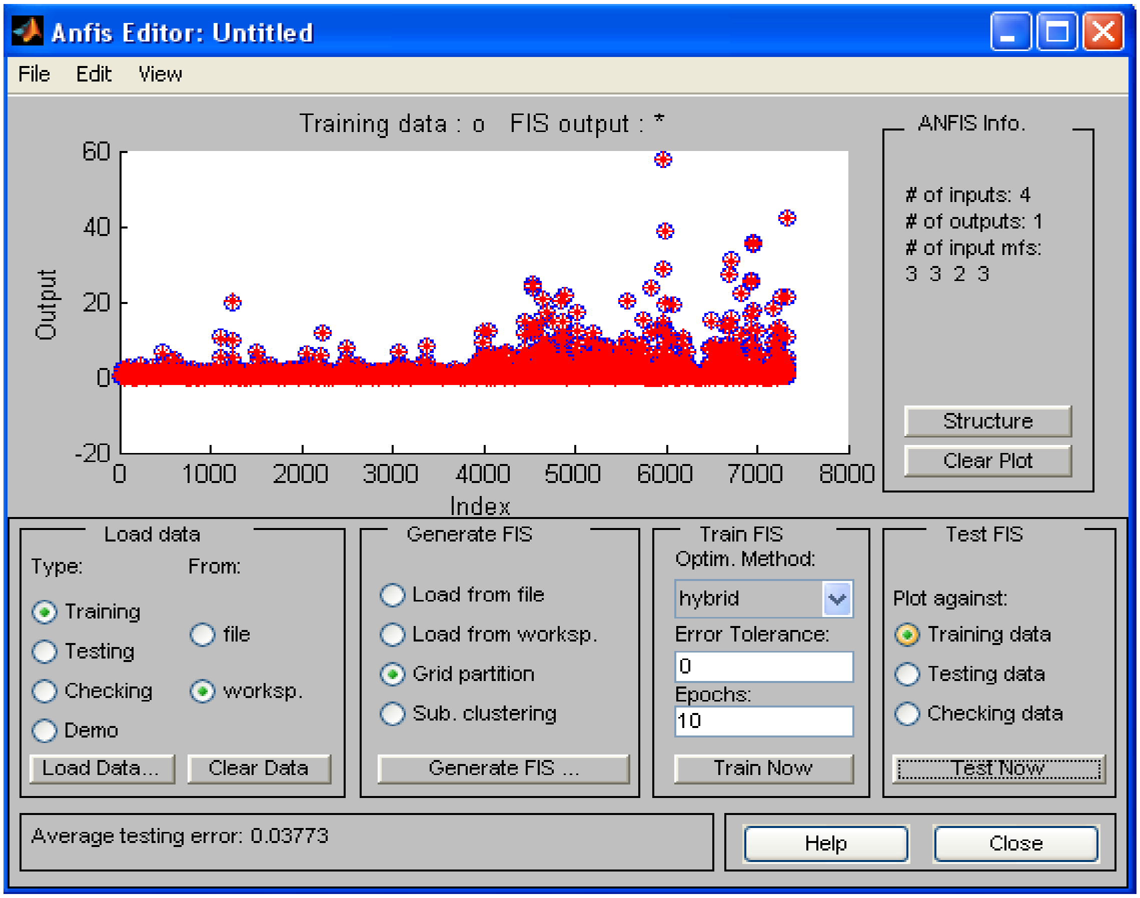
Plot against: ● Training data ○ Testing data ○ Checking data

Epoch 10:error= 0.03773

Figure 4-4 : ANFIS Editor

4.4.6 Validating the trained FIS

After the fuzzy inference system is trained, the last step of this process is to validate the resulting FIS against the training data. It plots the test data against FIS output. The validation process compares the load training data with the generated output data of auto created fuzzy inference system. Figure 4.5 showing the validate model diagram for FIS.



ANFIS Info.

# of inputs: 4
# of outputs: 1
# of input mfs:
3 3 2 3

Load data

Type: ● Training ○ Testing ○ Checking ○ Demo

From: ○ file ● worksp.

Generate FIS

○ Load from file
○ Load from worksp.
● Grid partition
○ Sub. clustering

Train FIS

Optim. Method: hybrid

Error Tolerance: 0

Epochs: 10

Test FIS

Plot against: ● Training data ○ Testing data ○ Checking data

Average testing error: 0.03773

Figure 4-5 : ANFIS Editor (Validate Model)

4.5 Fuzzy Inference Process

Based on the inputs of linguistic variables and rules defined above, the fuzzy inference system gives the final output by using inference steps given below.

4.5.1 Fuzzify inputs

The first step in fuzzy inference process is to take crisp numerical value as an input and determining the degree of membership in respective fuzzy sets and output is the fuzzy degree of membership. [4]

4.5.2 Apply fuzzy operator and implication method

In case of multiple parts of antecedents, fuzzy logic operators are applied to the antecedents to determine a single numeric value between 0 and 1. This is the degree of the support of the rule. The output of each rule is a consequent fuzzy set. rule's weight must be considered before applying the implication method, which is applied to a number given by antecedents after applying fuzzy operator. The implication method determines the shape of output fuzzy set by a number given by antecedents.

4.5.3 Output Aggregation

After all rules are evaluated, their results are combined in some fashion to make a decision. The evaluated results which are the fuzzy sets of each rule are then aggregated to produce a single fuzzy set. This process is called aggregation. Aggregation happens once for each output variable in our case it will be applied to our output variable Relevance. The input of aggregation the output of each fuzzy set and output is a fuzzy set for output variable Score.

4.5.4 Defuzzification

The aggregated fuzzy set in the input to this process and the output is a single number. The range of values are covered in aggregated fuzzy set and crisp value can be determined after its defuzzification. Weighted Sum and Weighted Average are common methods used for defuzzification in MATLAB. As Weighted Average produces more flexible results, so its used for defuzzification here.

SECTION V

RESULTS AND EVALUATIONS

5 Results and Evaluations

In this chapter the results and evaluations of the proposed scheme is discussed in detail. Evaluating the effectiveness of IR system is non-trivial process. Many different measures for evaluating the performance of information retrieval systems have been proposed. The most widely used statistical classification is precision and recall. In IR scenario, the set of retrieved documents helps to define precision and recall and set of relevant documents. Other common measures to be used are Mean Average Precision (MAP), F-measure, E-measure, Fallout etc.

The performance of proposed approach is compared with existing algorithms i.e TF-IDF and TF-IDF Length Normalized discussed in chapter 2. All preprocessing steps (i.e. stop words removal and stemming) were identical.

5.1 Performance Measure

Normally the performance of IR algorithms is measured through multiple performance metrics. The collection of queries and documents is required for these measures. Relevancy defines the ground truth for all these measures. A document can either be relevant or non relevant with respect to a query.

The returning results of a query uses following measures to measure the effectiveness of our proposed scheme.

5.2 Precision

In an Information Retrieval scenario, Precision is defined as the ratio of relevant documents retrieved to total number of documents retrieved by the search. Precision can be seen as a measure of exactness [7].

This is the fraction of the returned results that are relevant to the information need [11].

$$Precision = \frac{|\{Relevant\ Document\} \cap \{Retrieved\ Documents\}|}{|\{Retrieved\ Documents\}|}$$

$$Precision = \frac{\#\ (Relevant\ Items\ Retrieved)}{\#\ (Retrieved\ Items)}$$

Where numerator represents number of relevant documents that are retrieved and denominator represents total number of documents retrieved. A Precision score of 1.0 means that every result retrieved by a search was relevant.

If the total number of hits is 5 out of which 4 are relevant i.e total number of relevant hits is 4. Then

Precision = 4 / 5*100= 80 %

Table 5.1 shows the precision values of 100 queries against all documents in the corpus.

Table 5-1: Precision values of 100 cf queries

Precision			
Query No	**TF IDF**	**TF-Norm**	**ZoRFIS**
1	0.1704	0.1839	0.2366
2	0.0222	0.0180	0.0252
3	0.1649	0.1600	0.1509
4	0.0540	0.0675	0.0800
5	0.4590	0.5555	0.4175
6	0.0955	0.0898	0.1402
7	0.0542	0.0774	0.0761
8	0.0461	0.0454	0.0769
9	0.0784	0.0761	0.0833
10	0.1791	0.1690	0.1538
11	0.2758	0.2394	0.2000
12	0.0241	0.0201	0.0333
13	0.1466	0.1182	0.1250
14	0.1594	0.1587	0.2119

Precision			
Query No	TF IDF	TF-Norm	ZoRFIS
15	0.3600	0.5000	0.5000
16	0.2608	0.2790	0.3333
17	0.1157	0.1188	0.1818
18	0.2372	0.2272	0.2632
19	0.0588	0.0822	0.1415
20	0.2436	0.2460	0.3118
21	0.0743	0.0746	0.1029
22	0.2580	0.3714	0.4419
23	0.0638	0.0961	0.1364
24	0.0540	0.0666	0.2105
25	0.2533	0.2441	0.2316
26	0.1944	0.2173	0.1491
27	0.0300	0.0223	0.1081
28	0.3214	0.3548	0.3636
29	0.0645	0.1470	0.1455
30	0.1384	0.1733	0.1875
31	0.2388	0.3768	0.5636
32	0.0877	0.1176	0.1364
33	0.6750	0.5538	0.3846
34	0.2083	0.1506	0.2658
35	0.0534	0.0479	0.0538
36	0.0365	0.0515	0.0714
37	0.3900	0.3666	0.5667
38	0.3571	0.4838	0.3250
39	0.6216	0.6800	0.5984
40	0.2592	0.2156	0.2308
41	0.1000	0.0864	0.0667
42	0.1326	0.1110	0.2063
43	0.4404	0.4387	0.4468
44	0.3512	0.3058	0.6378
45	0.1000	0.1232	0.1563

Precision			
Query No	TF IDF	TF-Norm	ZoRFIS
46	0.0679	0.0693	0.0972
47	0.0791	0.0869	0.1408
48	0.1391	0.1666	0.1793
49	0.1839	0.1882	0.2258
50	0.1265	0.1264	0.1930
51	0.4689	0.4336	0.5660
52	0.0384	0.0625	0.0167
53	0.1559	0.1360	0.1635
54	0.3580	0.3431	0.3000
55	0.0500	0.0476	0.0962
56	0.0909	0.0827	0.0901
57	0.3392	0.3454	0.3636
58	0.3883	0.3893	0.4180
59	0.2777	0.2400	0.3559
60	0.1515	0.1851	0.0946
61	0.2252	0.2113	0.2162
62	0.3188	0.3111	0.3684
63	0.0967	0.1166	0.1579
64	0.1449	0.1360	0.1649
65	0.2297	0.2297	0.3047
66	0.1641	0.1764	0.1579
67	0.3157	0.2333	0.1290
68	0.1538	0.2413	0.1333
69	0.1142	0.2068	0.1212
70	0.1666	0.2222	0.1803
71	0.5000	0.2222	0.0588
72	0.1428	0.1627	0.1455
73	0.0952	0.1206	0.1087
74	0.0447	0.0422	0.0517
75	0.1627	0.1428	0.2453
76	0.0593	0.0542	0.0833

Precision			
Query No	TF IDF	TF-Norm	ZoRFIS
77	0.0937	0.1171	0.1892
78	0.1480	0.1455	0.3925
79	0.1686	0.2105	0.2375
80	0.3330	0.2567	0.2727
81	0.1355	0.1551	0.1698
82	0.1888	0.1904	0.2314
83	0.1415	0.1355	0.1566
84	0.1200	0.1237	0.1132
85	0.0869	0.0833	0.1129
86	0.1818	0.1666	0.2180
87	0.0857	0.0886	0.1262
88	0.0571	0.0526	0.0662
89	0.0501	0.0424	0.1176
90	0.1739	0.1726	0.2105
91	0.2844	0.2836	0.3600
92	0.2500	0.4117	0.5077
93	0.0512	0.0500	0.0583
94	0.4687	0.3750	0.4340
95	0.0540	0.0459	0.0577
96	0.0533	0.0476	0.1154
97	0.0482	0.0473	0.0593
98	0.0689	0.0833	0.0806
99	0.0750	0.0937	0.0615
100	0.0888	0.0888	0.1471
MAP	**0.1797**	**0.1851**	**0.2095**

Figure 5.1 shows the comparison of precisions values of all three implemented schemes. i.e. comparison of TF-IDF, TF-Norm and ZoRFIS.

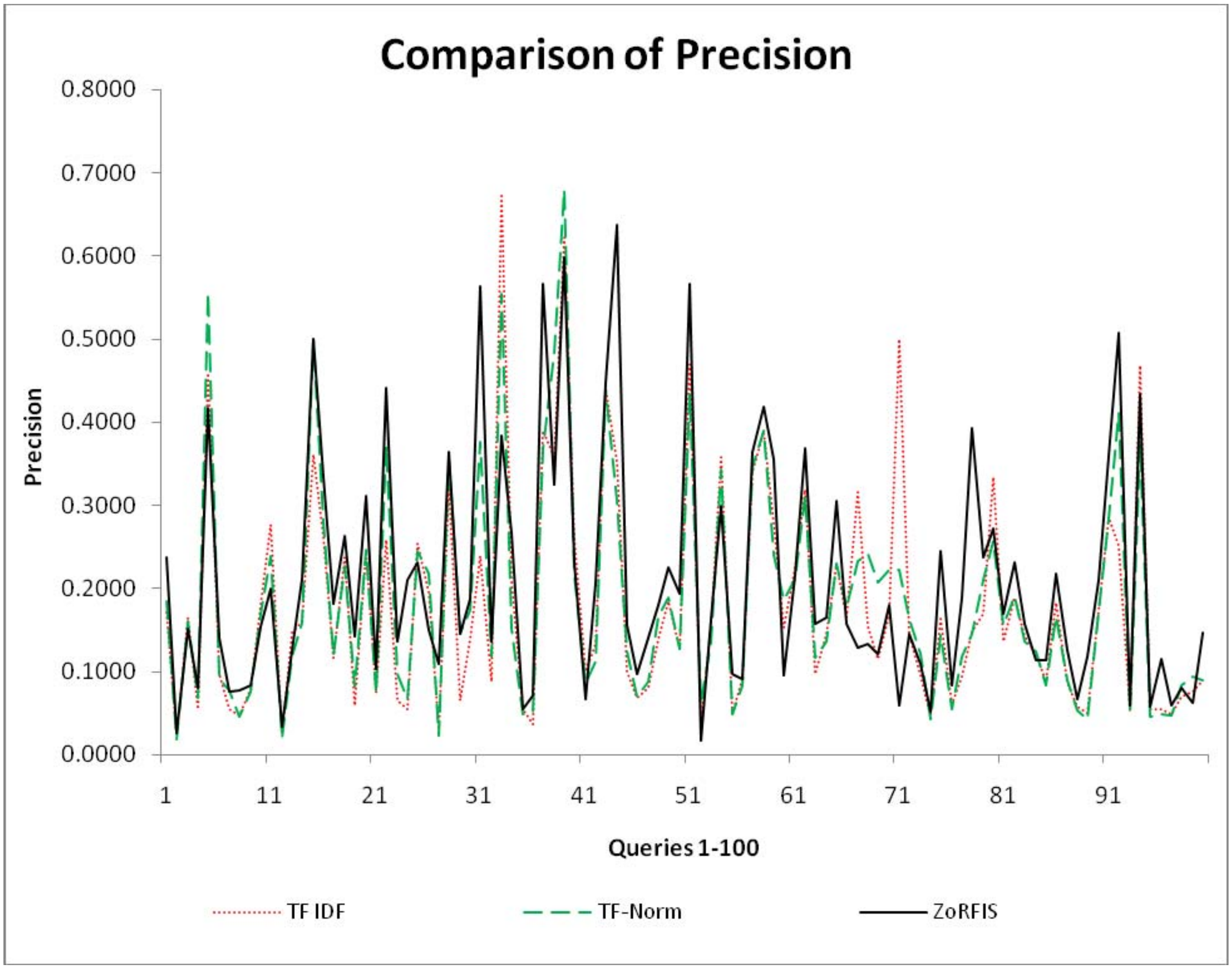


Figure 5-1: Precision values of all 100 queries

5.3 Recall

Recall is defined as the ratio of relevant documents retrieved to the total number of existing relevant documents (that should be retrieved). Recall can be seen as a measure of completeness. [7]

This is the fraction of the relevant documents in the collection that were returned by the system [11].

$$Recall = \frac{|\{Relevant\ Documents\} \cap \{Retrieved\ Documents\}|}{|\{Relevant\ Documents\}|}$$

$$Recall = \frac{\#\,(Relevant\ Items\ Retrieved)}{\#\,(Relevant\ Items)}$$

Where numerator represents the relevant documents that are retrieved and denominator represents the total number of relevant documents in a collection. A Recall score of 1.0 means that all relevant documents were retrieved by the search.

If the total number of relevant documents is 10 out of which 7 are retrieved. Then

Recall = 7 / 10*100= 70 %

Table 5.2 shows the recall of 100 queries against all documents in the corpora.

Table 5-2 : Recall Values of 100 queries

Recall			
Query No	**TF IDF**	**TF-Norm**	**ZoRFIS**
1	0.4411	0.4706	0.6471
2	0.4286	0.4286	0.4286
3	0.3720	0.3953	0.3721
4	0.4444	0.5556	0.6667
5	0.1297	0.1527	0.3282
6	0.6250	0.6250	0.6250
7	0.2500	0.3929	0.2500
8	0.1363	0.1363	0.0909
9	0.8000	0.8000	0.8000
10	0.4800	0.4800	0.5600
11	0.7273	0.7727	0.7273
12	0.4285	0.4285	0.5714
13	0.4583	0.4583	0.4167
14	0.4000	0.5455	0.4545
15	0.0865	0.0962	0.2692
16	0.1667	0.2000	0.1944
17	0.2545	0.3091	0.3273
18	0.6667	0.7143	0.7143

Recall			
Query No	TF IDF	TF-Norm	ZoRFIS
19	0.3182	0.5909	0.6818
20	0.6304	0.6739	0.6304
21	0.3600	0.4000	0.2800
22	0.1143	0.1857	0.2714
23	0.0857	0.1429	0.1714
24	0.0645	0.0645	0.1290
25	0.3725	0.4118	0.4314
26	0.2121	0.3030	0.5152
27	0.3636	0.3636	0.3636
28	0.1765	0.2157	0.3137
29	0.0455	0.1136	0.1818
30	0.5294	0.7647	0.7059
31	0.2759	0.4483	0.5345
32	0.1667	0.2667	0.2000
33	0.4737	0.6316	0.6140
34	0.6410	0.6410	0.5385
35	0.5000	0.5714	0.5000
36	0.3333	0.5556	0.8889
37	0.5612	0.6735	0.6939
38	0.1887	0.2830	0.4906
39	0.1420	0.2099	0.4506
40	0.1917	0.2260	0.1438
41	0.5333	0.4667	0.4667
42	0.2766	0.2766	0.2766
43	0.3776	0.4388	0.4286
44	0.5143	0.5571	0.5786
45	0.3500	0.4500	0.5000
46	0.3889	0.3889	0.3889
47	0.1833	0.2333	0.1667
48	0.5333	0.7333	0.8667
49	0.4444	0.4444	0.3889

Recall			
Query No	**TF IDF**	**TF-Norm**	**ZoRFIS**
50	0.4000	0.4400	0.4400
51	0.3444	0.4066	0.1245
52	0.5000	0.5000	0.5000
53	0.7391	0.7391	0.7391
54	0.4677	0.5645	0.5806
55	0.1667	0.1667	0.2083
56	0.3438	0.3750	0.3125
57	0.7451	0.7451	0.7843
58	0.4211	0.4632	0.5368
59	0.1034	0.0828	0.1448
60	0.1471	0.1471	0.2059
61	0.3571	0.3143	0.3429
62	0.1528	0.1944	0.3403
63	0.1538	0.1795	0.2308
64	0.4000	0.4600	0.3200
65	0.1809	0.2872	0.4149
66	0.1746	0.2381	0.2857
67	0.4286	0.5000	0.5714
68	0.1333	0.1556	0.2667
69	0.2667	0.4000	0.5333
70	0.4118	0.4706	0.6471
71	0.3333	0.3333	0.3333
72	0.5385	0.5385	0.6154
73	0.3333	0.3889	0.5556
74	0.4286	0.4286	0.4286
75	0.6667	0.6190	0.6190
76	0.4118	0.4118	0.4706
77	0.1098	0.1585	0.1707
78	0.5844	0.7013	0.5455
79	0.3256	0.3721	0.4419
80	0.5000	0.5588	0.5294

Recall			
Query No	**TF IDF**	**TF-Norm**	**ZoRFIS**
81	0.2857	0.3214	0.3214
82	0.2742	0.3226	0.4516
83	0.5000	0.5000	0.4063
84	0.7500	0.7500	0.7500
85	0.3226	0.3548	0.4516
86	0.5957	0.6170	0.6170
87	0.1579	0.1842	0.3421
88	0.4286	0.5000	0.7143
89	0.8235	0.8235	0.8235
90	0.8571	0.8571	0.8571
91	0.1086	0.1316	0.1184
92	0.0926	0.1296	0.3056
93	0.1111	0.1111	0.3333
94	0.3488	0.2791	0.5349
95	0.4444	0.4444	0.3333
96	0.3333	0.3333	0.2500
97	0.6364	0.7273	0.6364
98	0.2667	0.3333	0.3333
99	0.6000	0.6000	0.8000
100	0.3636	0.3636	0.4545
MAR	**0.3721**	**0.4171**	**0.4531**

Figure 5.2 shows the comparison of precisions values of all three implemented schemes. i.e. comparison of TF-IDF, TF-Norm and ZoRFIS.

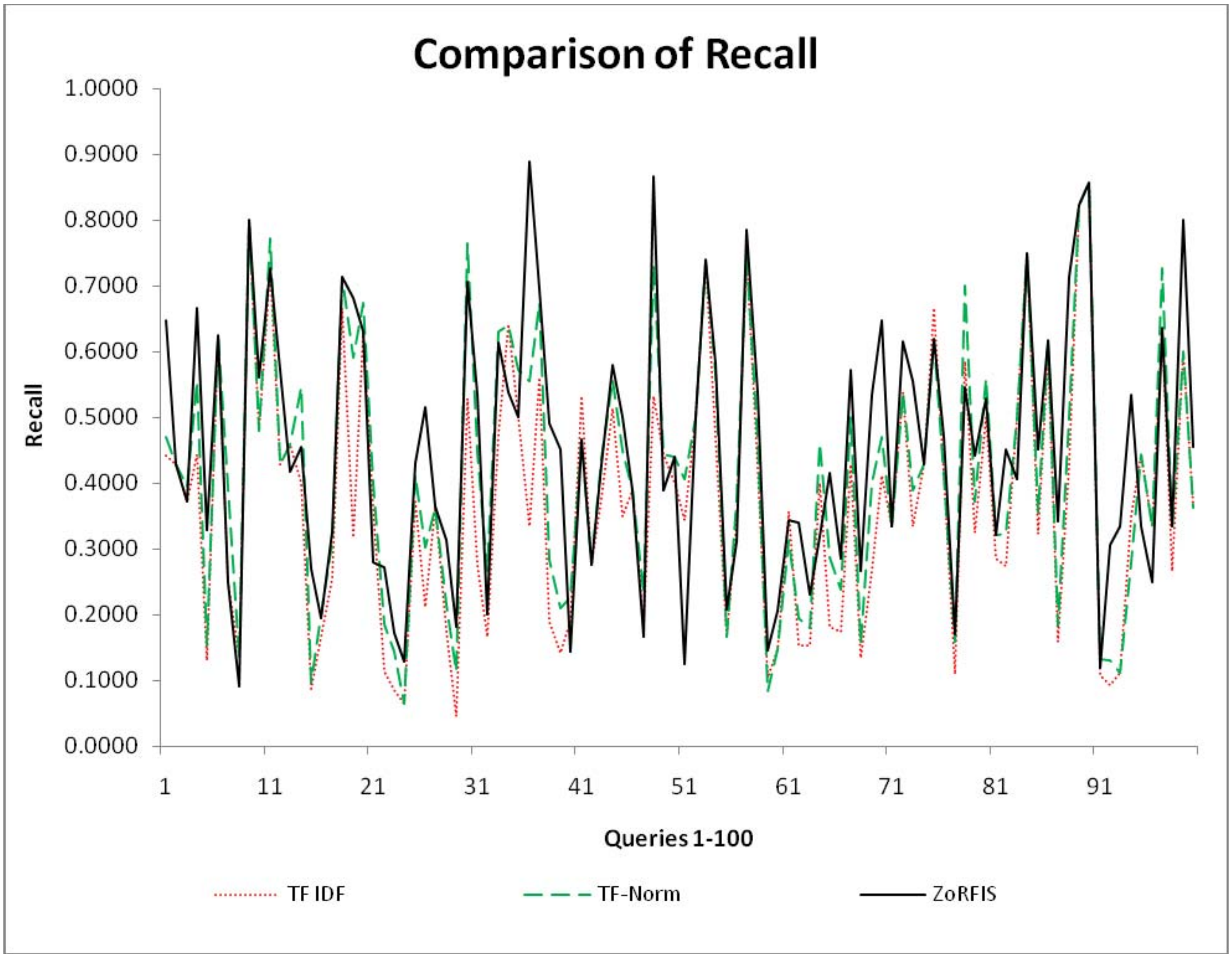


Figure 5-2: Recall Values of all 100 queries

5.4 The F-measure

We are often interested in trade-off between precision and recall; precision and recall are in inverse relationship with each other, one factor may increased at the cost of decreasing the other. For instance, by allowing the system to retrieve more documents often increasing the chances its Recall at the cost of retrieving number of irrelevant documents (decreasing precision). Likewise, to decide fruit is an orange by the classification system may obtain high precision with the classification of fruits with the color as oragnes and exact right shaped. But lowering the recall because of the false negatives from oranges, that doesn’t align with the specification.[16]

In general, scores of precision and recall cannot be examined in separation. Rather the value for one measure at fixed level is compared with the other measure (e.g. precison at recall level of 0.7). F-measure is a metric that combines both precision and recall into a single measure through the weighted harmonic mean.

This is the weighted harmonic mean of precision and recall. It trades off between precision P and recall R [16].

$$F = \frac{(\beta^2 + 1)PR}{\beta^2 \ (P + R)}$$

Where $\beta^2 - \frac{1-\alpha}{\alpha}$

$\alpha \in [0,1]$ and thus $\beta^2 \in [0,\infty]$. The default well adjusted F-measure that fairly weights precision and recall uses the parameters $\alpha = {}^1/_2$ or $\beta = 1$. It is usually written as F1 measure represented by $F_{\beta=1}$ [14].

$$F_{\beta=1} = \frac{2PR}{P + R}$$

Table 5.3 shows the F-Measure of 100 queries against all documents in the corpora.

Table 5-3 : F-Measure of all 100 quiries

F-Measure			
Query No	TF IDF	TF-Norm	ZoRFIS
1	0.2458	0.2645	0.3464
2	0.0422	0.0345	0.0476
3	0.2285	0.2278	0.2148
4	0.0963	0.1204	0.1429
5	0.2023	0.2395	0.3675
6	0.1657	0.1570	0.2290
7	0.0891	0.1293	0.1167
8	0.0689	0.0681	0.0833
9	0.1428	0.1390	0.1509
10	0.2609	0.2500	0.2414

F-Measure			
Query No	TF IDF	TF-Norm	ZoRFIS
11	0.3999	0.3655	0.3137
12	0.0456	0.0384	0.0630
13	0.2221	0.1879	0.1923
14	0.2280	0.2459	0.2890
15	0.1395	0.1613	0.3500
16	0.2034	0.2330	0.2456
17	0.1591	0.1716	0.2338
18	0.3499	0.3447	0.3846
19	0.0993	0.1443	0.2344
20	0.3514	0.3604	0.4172
21	0.1232	0.1257	0.1505
22	0.1584	0.2476	0.3363
23	0.0731	0.1149	0.1519
24	0.0588	0.0655	0.1600
25	0.3015	0.3065	0.3014
26	0.2029	0.2531	0.2313
27	0.0554	0.0420	0.1667
28	0.2278	0.2683	0.3368
29	0.0533	0.1282	0.1616
30	0.2194	0.2826	0.2963
31	0.2560	0.4094	0.5487
32	0.1149	0.1632	0.1622
33	0.5567	0.5901	0.4730
34	0.3144	0.2439	0.3559
35	0.0965	0.0884	0.0972
36	0.0658	0.0943	0.1322
37	0.4602	0.4748	0.6238
38	0.2469	0.3571	0.3910
39	0.2311	0.3207	0.5141
40	0.2204	0.2207	0.1772
41	0.1684	0.1458	0.1167
42	0.1793	0.1584	0.2364
43	0.4066	0.4387	0.4375
44	0.4174	0.3949	0.6067
45	0.1556	0.1934	0.2381
46	0.1156	0.1176	0.1556
47	0.1105	0.1266	0.1527
48	0.2207	0.2715	0.2971

F-Measure			
Query No	TF IDF	TF-Norm	ZoRFIS
49	0.2602	0.2644	0.2857
50	0.1922	0.1964	0.2683
51	0.3971	0.4197	0.2041
52	0.0713	0.1111	0.0322
53	0.2575	0.2297	0.2677
54	0.4056	0.4268	0.3956
55	0.0769	0.0741	0.1316
56	0.1438	0.1355	0.1399
57	0.4662	0.4720	0.4969
58	0.4040	0.4230	0.4700
59	0.1507	0.1231	0.2059
60	0.1492	0.1639	0.1296
61	0.2762	0.2527	0.2652
62	0.2066	0.2393	0.3538
63	0.1188	0.1414	0.1875
64	0.2127	0.2099	0.2177
65	0.2024	0.2553	0.3513
66	0.1692	0.2027	0.2034
67	0.3636	0.3182	0.2105
68	0.1428	0.1892	0.1778
69	0.1599	0.2726	0.1975
70	0.2372	0.3019	0.2820
71	0.4000	0.2666	0.1000
72	0.2257	0.2499	0.2353
73	0.1481	0.1841	0.1818
74	0.0810	0.0768	0.0923
75	0.2616	0.2321	0.3513
76	0.1037	0.0958	0.1416
77	0.1011	0.1347	0.1795
78	0.2362	0.2410	0.4565
79	0.2222	0.2689	0.3089
80	0.3998	0.3518	0.3600
81	0.1838	0.2092	0.2222
82	0.2236	0.2395	0.3060
83	0.2206	0.2132	0.2261
84	0.2069	0.2124	0.1967
85	0.1369	0.1349	0.1806
86	0.2786	0.2624	0.3222

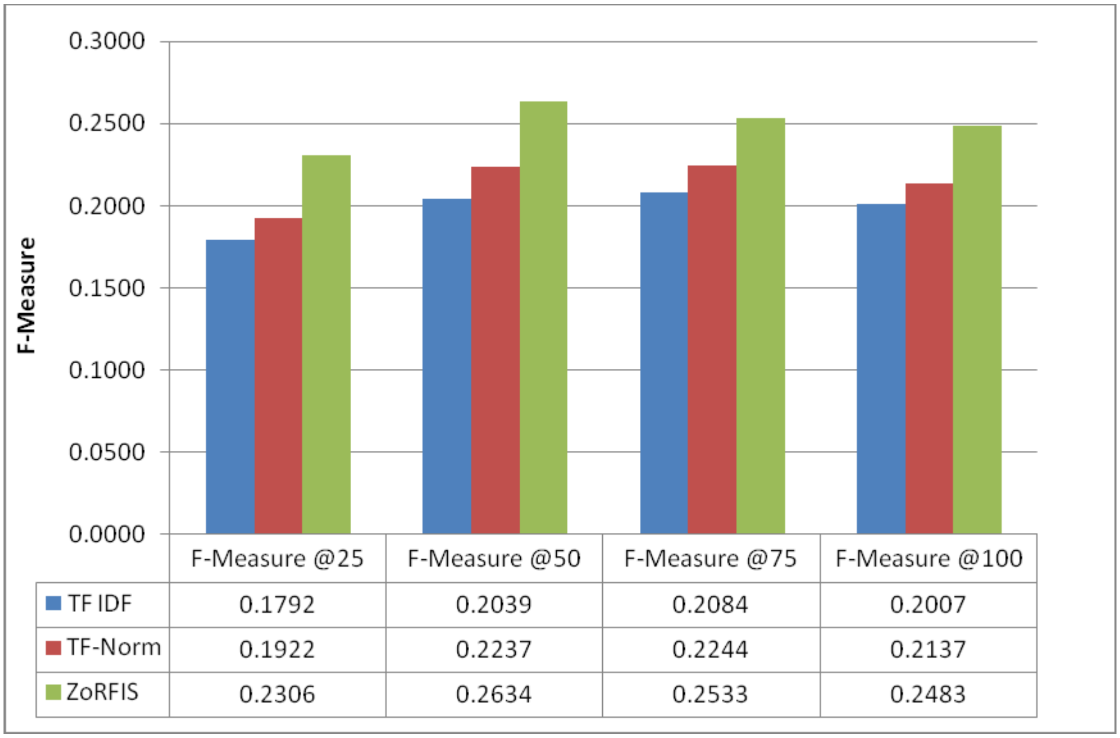
F-Measure			
Query No	TF IDF	TF-Norm	ZoRFIS
87	0.1111	0.1197	0.1844
88	0.1008	0.0952	0.1212
89	0.0945	0.0806	0.2059
90	0.2891	0.2873	0.3380
91	0.1571	0.1797	0.1782
92	0.1351	0.1972	0.3815
93	0.0701	0.0690	0.0992
94	0.4000	0.3200	0.4792
95	0.0963	0.0832	0.0984
96	0.0919	0.0833	0.1579
97	0.0896	0.0888	0.1085
98	0.1095	0.1333	0.1299
99	0.1333	0.1621	0.1143
100	0.1427	0.1427	0.2222
F-Measure @ 100	**0.2007**	**0.2137**	**0.2483**

Table 5.4 shows the average F-Measure for 25, 50, 75 and 100 queries against all documents in the corpora

Table 5-4 : F-Measure for queries @ 25, 50, 75 & 100

Number of Queries	TF IDF	TF IDF Normalized	ZoR-FIS
F-Measure @25	0.1792	0.1922	0.2306
F-Measure @50	0.2039	0.2237	0.2634
F-Measure @75	0.2084	0.2244	0.2533
F-Measure @100	0.2007	0.2137	0.2483

Figure 5.3 shows the comparison of precisions values of all three implemented schemes. i.e. comparison of TF-IDF, TF-Norm and ZoRFIS



	F-Measure @25	F-Measure @50	F-Measure @75	F-Measure @100
🟦 TF IDF	0.1792	0.2039	0.2084	0.2007
🟥 TF-Norm	0.1922	0.2237	0.2244	0.2137
🟩 ZoRFIS	0.2306	0.2634	0.2533	0.2483

Figure 5-3 : Comparison of F-Measure @ 25, 50, 75 & 100 queries

5.5 The Mean Average Precision (MAP)

For a single information need, the precisions for the set of top k documents is averaged with the total relevant documents retrieved till that point is said to be Mean Average Precision (MAP). For instance, From the set of relevant documents $(d_1, d_2, d_3, \ldots, d_n)$ for user information need $q_j \in Q$ and from the top most results R_{jk} is a set of ranked retrieval results until the document R_k is retrieved [12].

$$MAP(Q) = \frac{1}{|Q|}\sum_{j=1}^{|Q|}\frac{1}{m_j}\sum_{k=1}^{m_j} Precision(R_{jk})$$

Table 5.5 shows the average precisions for 25, 50, 75 and 100 queries against all documents in the corpora

.

Table 5-5: Comparison of MAP @25, @50, @75, @100 queries

Number of Queries	TF IDF	TF IDF Normalized	ZoR-FIS
MAP @25	0.1578	0.1732	0.1929
MAP @50	0.1917	0.2035	0.2340
MAP @75	0.1916	0.1979	0.2165
MAP @100	0.1797	0.1851	0.2095

We assessed the performance of our proposed solution against Mean Average Precision values. Figure 5.4 shows the precisions and Mean Average Precision of all 100 queries for our proposed scheme and TF-IDF and TF-Norm Schemes.

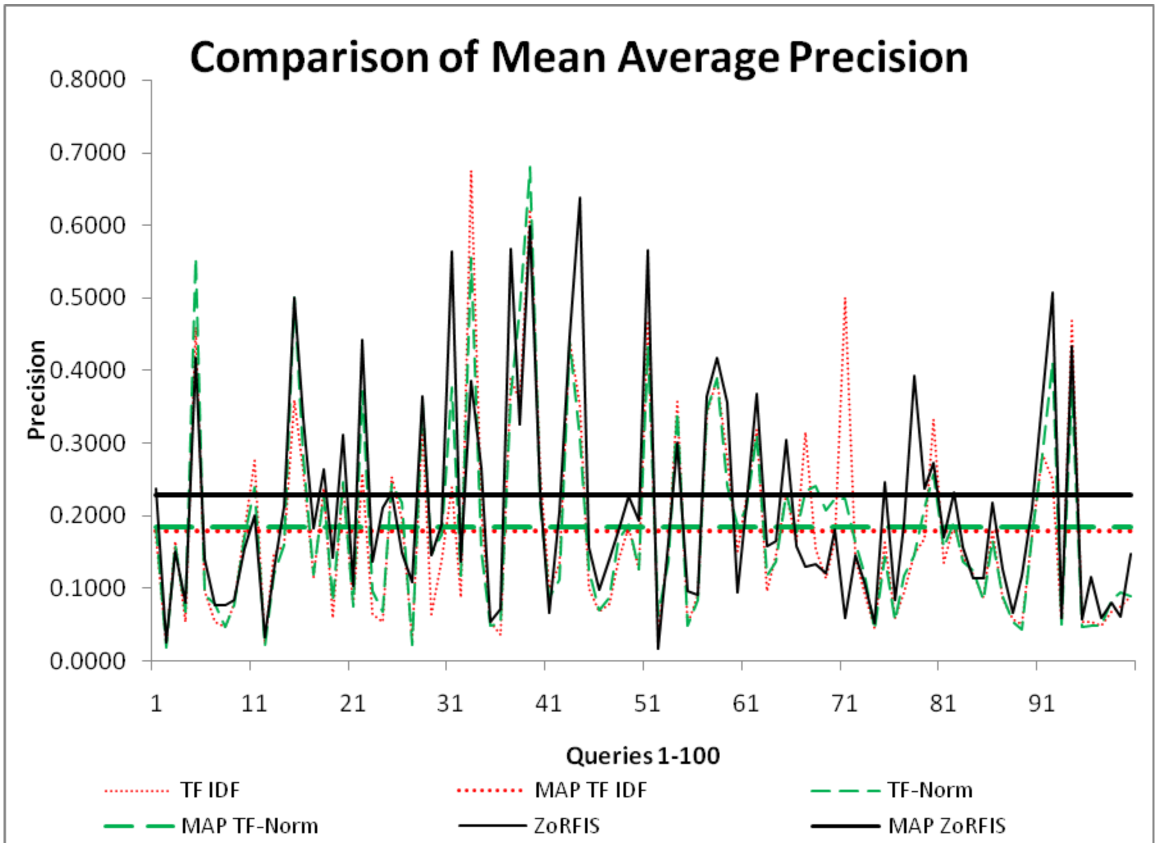
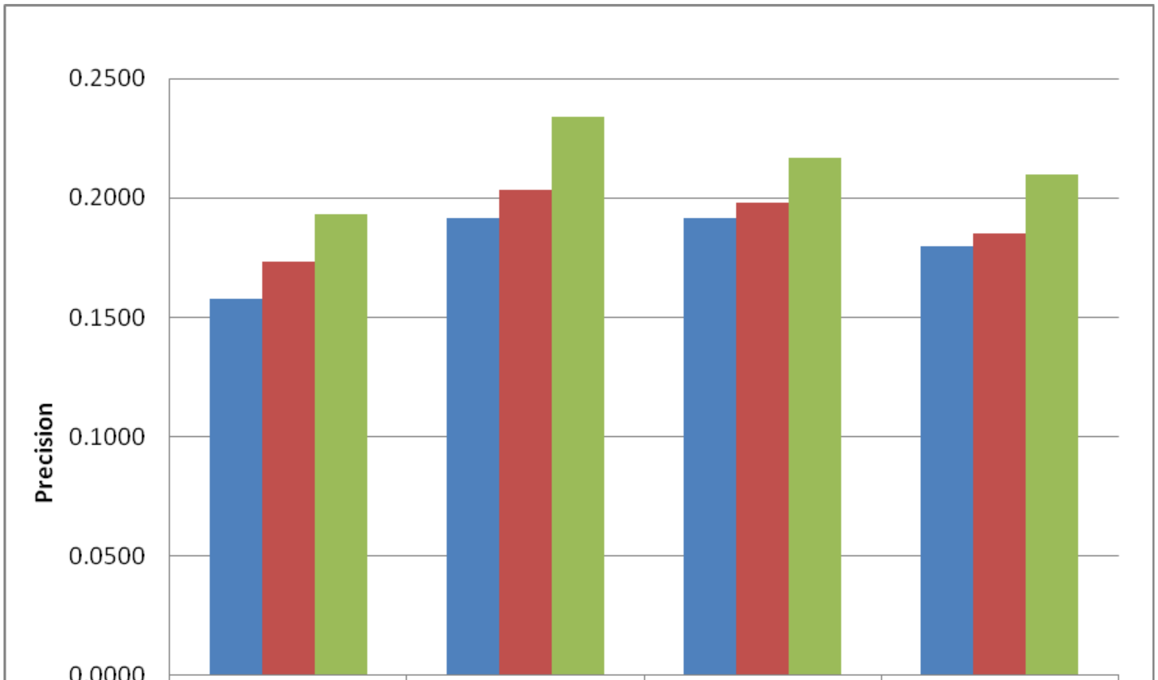


Figure 5-4 : Comparison of Precision and Mean Average Precision for all 100 queries

Figure 5.5 shows the MAP of 25, 50, 75 and 100 queries showing that ZoRFIS has better performance against the exiting approaches.



	MAP @25	MAP @50	MAP @75	MAP @100
🟦 TF IDF	0.1578	0.1917	0.1916	0.1797
🟥 TF-Norm	0.1732	0.2035	0.1979	0.1851
🟩 ZoRFIS	0.1929	0.2340	0.2165	0.2095

Figure 5-5: Comparison of MAP @ 25, @50, @75, @100 queries

5.6 Mean Average Recall

For a single information need, the recall values for the set of top k documents is averaged with the total relevant documents retrieved till that point is said to be Mean Average Recall (MAR). For instance, From the set of relevant documents $(d_1, d_2, d_3, \dots, d_n)$ for user information need $q_j \in Q$ and from the top most results R_{jk} is a set of ranked retrieval results until the document R_k is retrieved [12].

$$MAR(Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} Recall(R_{jk})$$

Table 5.6 shows the average recalls for 25, 50, 75 and 100 queries against all documents in the corpora.

Table 5-6 : Comparison of MAR @ 25, @50, @75, @100 queries

Number of Queries	TF IDF	TF IDF Normalized	ZoR-FIS
MAR @25	0.3636	0.4128	0.4444
MAR @50	0.3606	0.4222	0.4471
MAR @75	0.3604	0.4110	0.4446
MAR @100	0.3721	0.4171	0.4531

We assessed the performance of our proposed solution against against Mean Average Recall values. Figure 5.6 shows the recall and Mean Average Recall of all 100 queries for our proposed scheme and TF-IDF and TF-Norm Schemes.

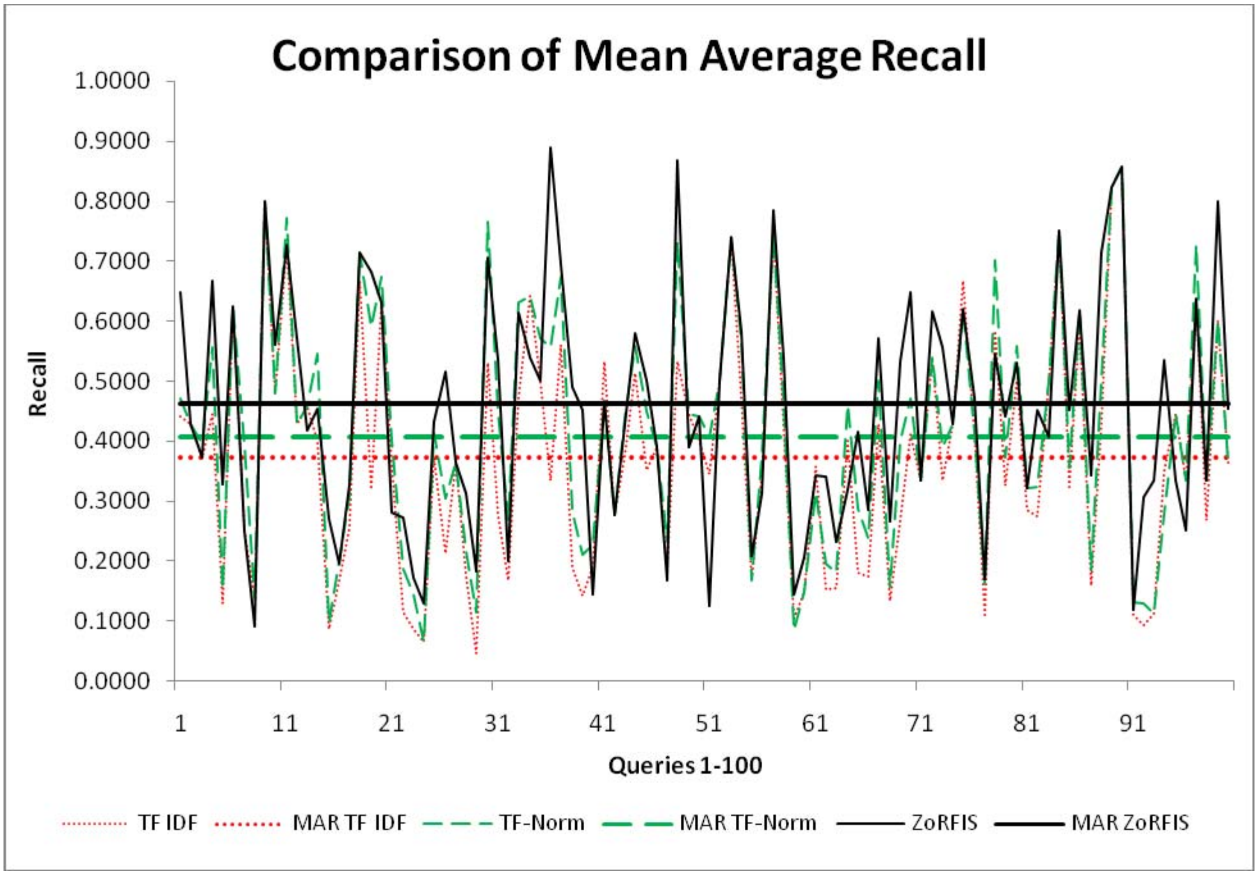
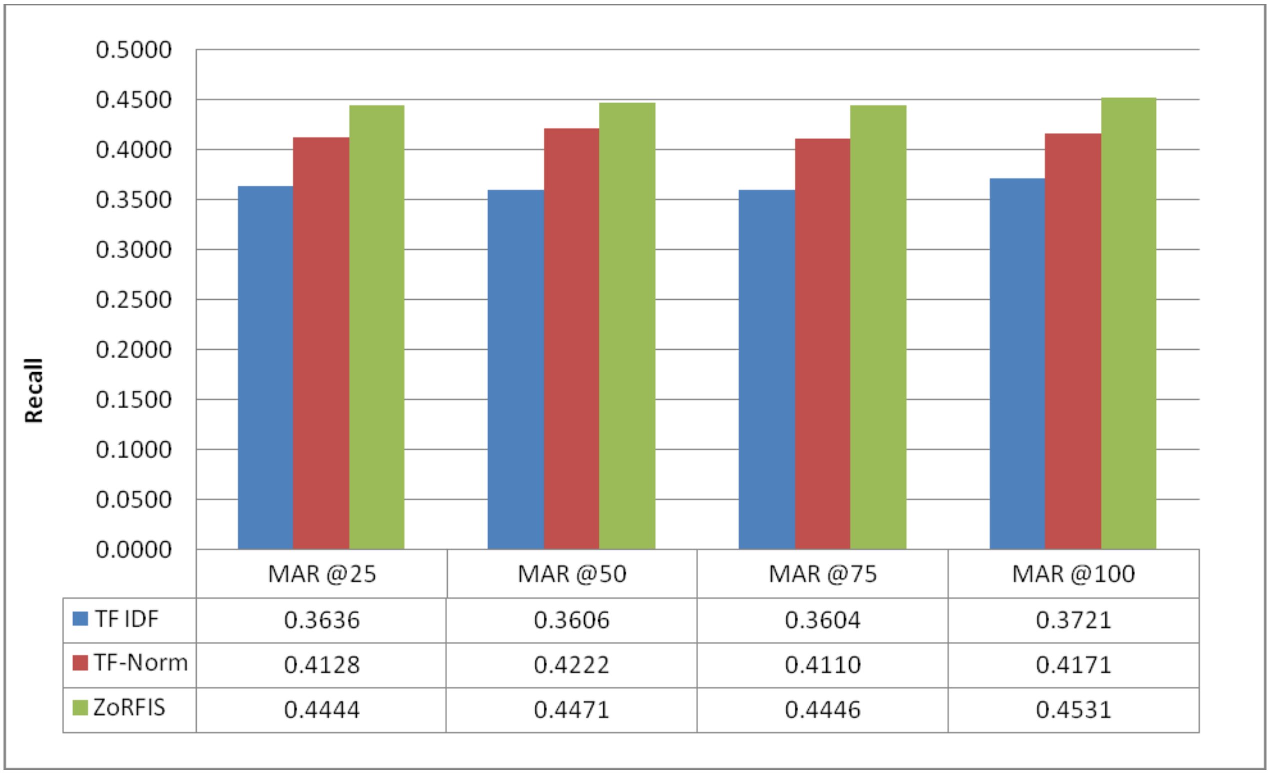


Figure 5-6: Comparison of Recall and MAR for all 100 queries

Figure 5.7 shows the MAR of 25, 50, 75 and 100 queries showing that ZoRFIS has better performance against the exiting approaches.



	MAR @25	MAR @50	MAR @75	MAR @100
🟦 TF IDF	0.3636	0.3606	0.3604	0.3721
🟥 TF-Norm	0.4128	0.4222	0.4110	0.4171
🟩 ZoRFIS	0.4444	0.4471	0.4446	0.4531

Figure 5-7 : Comparison of MAR @25, @50, @75, @100 queries

5.7 **Corpus**

A gold standard Cystic Fibrosis Corpus (CFC)[5] is used to compare the performance of existing and proposed algorithms. The CFC database consists of six files: cf74 to cf79 containing 1,239 documents published from 1974 to 1979. There is also a query a set of 100 queries with the respective relevant documents as answered by 4 different domain experts at scale of 0, 1, 2 from irrelevant, marginally relevant, and relevant.

The CFC database is in SGML format whose parsers is not available for Java. As the proposed system is developed in Java, so a manual parser is written to convert these documents from existing format into XML format, whose parsers are available for almost all languages.

The performance of our proposed approach is compared with existing approaches TF-IDF and Weighted Zone Scoring model. All preprocessing steps have the same effect on performance.

5.8 Conclusion

In this research we introduced a more comprehensive model called Zonal Ranked Fuzzy Inference System for the retrieval of relevant documents. ZoR-FIS calculates score of documents against a user query based on different set of rules. The ZoR-FIS approach has given significant improvements in the major retrieval metrics on comparison of its performance with the performance of existing relevance scoring formulae such as length normalized TFIDF and traditional TFIDF. The thorough analysis of returned results shows that ZoR-FIS returns some more relevant information by making synonym and phrase queries that can’t be retrieved by existing techniques.

SECTION VI

CONCLUSION & FUTURE ENHANCEMENTS

6 Conclusions and Future Work

In this research we introduced a comprehensive model called Zone Ranked Fuzzy Inference System (ZoRFIS) for the retrieval of relevant documents. We started with an introduction of Information Retrieval (IR) and briefly explained the history and challenges of IR. We described the logical view of document and explain the steps required to make the logical view of documents for efficient information retrieval. In our context we removed the stop words and then performed stemming on remaining text to make the logical view of documents ready for being processed by existing and proposed algorithms.

Later we explained the background and gave thorough overview of different categories of information retrieval model. There are three famous model in classical theory of information retrieval, named as Boolean, Vector and Probabilistic. Documents is either relevant or relevant in Boolean model. Documents and queries are represented through vectors in vector space models. Therefore its an algebraic model. Query and documents representation are modelled in the framework of probabilistic model that is build on probability theory. So the model is probabilistic as its name implies.

Here it is significant to differentiation of ranking and filtering of documents. In ranking, the documents are numbered only according to its relevance with user's query term while in filtering a user profile is defined in which his interests are recorded and documents that are considered relevant will be filtered. In filtering, documents with the ranking above certain threshold will be selected while the rest will be discarded [2].

To build a model we first have to think how the documents and queries need to be represented. Given these representations, The framework to be modelled is then conceived. This framework should also provide initiation for constructing a ranking function. For instance, in classic Boolean model, the framework is composed of documents and operations to be performed on those documents.

In real world a lot of fuzzy knowledge exists i.e. uncertain, vague, inexact, imprecise, ambiguous or probabilistic knowledge. Reasoning and human perception usually contain fuzzy information. It is hard to answer the questions in Boolean logic based system because such questions do not have exact answers. Only the humans can give the practical answers which are most likely to be true. Expert systems can give such answers with the description of their confidence level. The imprecision and vagueness of facts are used to measure that level. The incomplete and unreliable information can be handled by expert systems with different expert opinions.

The fuzzy logic maps the input space with output space with the use of the rules that is English like if-then statement. All the rules are evaluated at the same time and their order is not important [19]. e.g. If height is tall, The ranges of expected heights need to be defined together with what is meant by the term tall.

The applications of fuzzy logic are increasing drastically by numbers and varieties in recent years. There are countless applications for fuzzy logic and these applications range from predicting genetic traits, medical diagnosis, auto-focus on cameras, temperature control, decision support systems and Natural Language Processing (NLP) etc [4].

Among several combinations of soft computing methodologies, the most prominent at this occasion is that of hybrid neuro-fuzzy systems combining neuro-computing and fuzzy logic. A useful approach designed for this objective is called Adaptive Neuro-Fuzzy Inference System (ANFIS).

The existing IR models discussed above suffer from some problems. Boolean retrieval models are simple to implement but not very much effective. Vector Space Model is effective but too much pre-processing and disk space is required. Our model is a hybrid, using vector space model for information retrieval and logic based boolean model for document scoring. Based on fuzzy set theory and fuzzy logic, the proposed model gives simplicity of logic based models and the performance and flexibility of vector space models.

Lot of work has been done in the field of information retrieval but this system includes some special features that is useful in finding most relevant documents in the system by using fuzzy inference system. Most significant features included in this model for better relevance scoring of documents are weighted zone scoring and query expansion by means of semantic and phrase queries. In weighted zone scoring weights are assigned to zones according to their importance. Sometimes different authors use different words to define the same concepts, In such situations its always better to replace the words in queries with its synonyms for better matching of documents but its logical to give low membership values to these synonyms but it has their own importance. Most of the time user is interested to search for documents having the same order of query words in the documents. So the queries are redefined to generate multiple phrase queries having all consecutive combination of original query words. Here it is important to give high membership value to phrase queries.

Four input fuzzy variables are used in the proposed schemes and one output variable. Four input variables to be used are tf (defines the frequency of term in a document), idf (inverse document frequency) that defines how many documents contains the term its log of inverse of document frequency, overlap defines the weight of query term in a query, synonym query words has low overlap so get low membership value in fuzzy inference system while phrase queries gets high overlap so get higher membership value in fuzzy inference system. The last input fuzzy variable used is zone that describes the zone of query word in a document. Its logical to give higher membership value to this variable if the term occurs in title zone and give low membership value if the term occurs in Authors section of the document.

Evaluating the effectiveness of IR system is non-trivial process. The effectiveness of IR systems can be measured in many different ways, the most widely used statistical classification is precision and recall. In IR scenario, precision and recall are defined in terms of set of retrieved documents and set of relevant documents. Other common measures to be used are Mean Average Precision (MAP), F-measure, E-measure, Fallout etc. The ZoRFIS approach has given significant improvements in the major retrieval metrics on comparison of its performance with the performance of existing relevance scoring formulae such as length normalized TFIDF and traditional TFIDF. The thorough analysis of returned results shows that ZoRFIS returns some more relevant information by making synonym and phrase queries that can't be retrieved by existing techniques.

Future Enhancement

Although the best effort has been made to make the efficient and perfect fuzzy based information retrieval system but there is still a room available for its further enhancements. Following are the enhancements possible in our proposed system to optimizing the performance of ZoRFIS.

As we model our system using Adaptive Neuro Fuzzy Inference System (ANFIS) by giving limited data set to train our Fuzzy Inference System. The performance of system may be improved by providing the large data set. One of the limitations in our system is that we don't have any approach for the selection of parameters while training ZoRFIS. These techniques can be developed to select the optimized parameters for training of data. e,g, On training ZoRFIS using ANFIS, it asks for error tolerance and value for epochs. We have used a random number approach and tried different values for these parameters until an improved fuzzy inference system has been generated. So the methodology can be developed for the selection of these parameters.

ANFIS also asks from the user the no of membership functions for each fuzzy variable. Giving higher values will make the system close to training data set and too many rules will be generated. Less values for no of membership functions do not be closer than actual training data but the minimum rules will be generated by the system and hence the execution speed of overall system will be improved. A criteria or approach for the tradeoff between Performance and Execution speed can be developed to overall optimizing the performance of ZoRFIS.

The proposed model is tested on CFC (Cystic Fibrosis Collection) corpus which is the collection of 1279 medline documents and 100 cf queries. The given system can be tested on different corpus then compare and analyze the performance of given scheme with the existing schemes. More on that the given system can be tested on real data to better analyzing its performance.

Sugeno & mamdani are two types of fuzzy inference systems. The characteristics of the both were already discussed in our literature. We have used sugeno for our system and the reason has already been discussed. The proposed system can also be made by using mamdani and then we can compare the performance of mamdani and sugeno type fuzzy inference systems.

SECTION VII

REFERENCES

7 References

[1] Baeza-Yates R.and Ribeiro-Neto B. (1999) “Modern Information Retrieval”. ACM Press.

[2] Raghavan, and H. Schütze. (2008) “Introduction to Information Retrieval”, by C. Manning, P.. Cambridge University Press.

[3] N.O. RUBENS, “The application of fuzzy logic to the construction of the ranking function of information retrieval systems”, Computer Modelling and New Technologies, 2006, Vol.10, No.1, 20-27

[4] http://www.mathwork.com Fuzzy logic Toolbox User’s Guide (2004), The MathWorks Inc.reterived on dated 10 February 2010

[5] Shaw et al, “The Cystic Fibrosis Database: Content and Research Opportunities. LISR (13) (1991) 347-366.

[6] Bordogna G.and Pasi G. (1995) “Handling vagueness in information retrieval systems”. In Proceedings of the Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems, Nov. 20-23, 1995, pp.110-114.

[7] J.F. Baldwin, J. Lawry, and T.P. Martin. “A Mass Assignment Theory of the Probability of Fuzzy Events”. Fuzzy Sets and Systems, (83), pp. 353-367, 1996.

[8] J.F. Baldwin, T.P. Martin and B.W. Pilsworth. “Fril Fuzzy and Evidential Reasoning in Artificial Intelligence”. Research Studies Press Ltd, England, 1995.

[9] J.F. Baldwin, J. Lawry, and T.P. Martin. “A Mass Assignment Theory of the Probability of Fuzzy Events”. Fuzzy Sets and Systems, (83), pp. 353-367, 1996

[10] J. Larocca Neto, A.D. Santos, C.A.A. Kaestner, and A.A. Freitas. “Document Clustering and Text Summarization”. In Proceedings of the 4th Int. Conf. Practical Applications of Knowledge Discovery and Data Mining (PADD-2000), London: The Practical Application Company, pp 41---55, 2000b

[11] Masrah Azrifah Azmi Murad and Trevor Martin, “Similarity-Based Estimation for Document Summarization using Fuzzy Sets” Volume (1) Issue (4) 2005.

[12] Mustapha Baziz et.al. “A fuzzy set approach to concept-based information retrieval”, EUSFLAT - LFA 2005

[13] Donald H. Kraft et al. “Vagueness and Uncertainty in Information Retrieval: How can Fuzzy Sets Help?” In Proceedings of the 2006 International Workshop on Research Issues in Digital Libraries, Kolkata, West Bengal, India, December, 2006.

[14] M. W. Berry, S. T. Dumais, and G. W. O’Brien. “Using linear algebra for intelligent information retrieval”. SIAM Review, 37(4), 1995, 573-595, 1995.

[15] Michel Beigbeder and Annabelle Mercier. “An Information Retrieval Model Using the Fuzzy Proximity Degree of Term Occurences”. 2005 ACM Symposium on Applied Computing.

[16] Bhaskar Karn. “Information Retrieval System using Fuzzy Set Theory – The Basic Concepts”.

[17] Marti A. Hearst. “Untangling Text Data Mining”. School of Information Management & Systems University of California, Berkeley

[18] Flexible Information Retrieval [vagueness & uncertainty in information retrieval]. How can fuzzy sets help?

[19] NRC FuzzyJToolkit for the Java Platform, User’s Guide, Version 1.10a

[20] http://www.mathwork.com Fuzzy logic Toolbox User’s Guide (2004), The MathWorks Inc.reterived on dated 10 February 2010

Appendix A

Appendix A : Fuzzy Logic Toolbox

A-1 Fuzzy Logic Toolbox Description

In MATLAB, different set of functions are available in fuzzy logic toolbox that are used to create fuzzy systems. You can even develop your own Java programs that call on fuzzy systems you build with MATLAB. The user have the option to either sue the command line environment or the GUI based toolbox to fulfill the given task. Mathworks control library is available that allows to call MATLAB functions from within java programs.

Three types of tools are available in this toolbox:

- Graphical interactive tools
- Command line functions
- Simulink blocks

The command line function can be called from within the application. Some specialized functions are written in MATLAB files that implements the algorithms of fuzzy logic. You can view and edit these functions in a file using the statement

type function_name

GUI based interactive tools are also available to access these functions. The environment for FIS analysis, design and implementation is offered by these GUI-based tools.

Simulink environment is another category used to build high speed fuzzy systems and is the combination of blocks.

Most of the human reasoning is directly link with fuzzy rules that why fuzzy logic toolbox is much powerful and influential. By providing an efficient system for computation, the human reasoning becomes possible with this toolbox.

A-2 What Can Fuzzy Logic Toolbox Software Do?

Fuzzy logic toolbox enables to create and edit the fuzzy systems. These systems can be created manually or generated automatically by using adaptive neuro-fuzzy inference or clustering techniques.

The toolbox allows to run stand-alone Java programs directly. It can be achieved by Fuzzy Inference Engine that reads the fuzzy systems and called it from a java program by using Mathworks control library.

A-3 ANFIS AND ANFIS EDITOR GUI

Fuzzy logic toolbox has ANFIS editor GUI and the function anfis, that will be discussed in this section. These tools models the data by applying fuzzy inference techniques. The parameters defines the shape of membership functions as we have seen in other fuzzy inference systems. But in ANFIS, the parameters for membership functions are chosen automatically by looking at the data.

A-4 Model Learning and Inference through ANFIS

For some modeling situations membership functions cannot be determined by just looking at the data, In such situation the data will be modelled by neuro-adaptive learning method. The information for the data set is learnt through fuzzy modelling procedures by using Neuro-adaptive learning mechanism. Membership function parameters are computed through fuzzy logic toolbox that allows to map the input and output data by fuzzy inference system. This can be done by using a function called anfis available in fuzzy logic toolbox.

A-4.1 FIS Structure

It's a network type structure to model input/output data in such a way that the data to modelled will corresponds to its respective input and output membership functions.

A-4.2 Parameter Adjustment

These parameters can be measured with the use of gradient vector. This gradient vector measures, how good the input/output dataset be modelled by fuzzy inference system for a given set of parameters. Optimization techniques can be used after getting the gradient vector in order to change the parameters and minimizing the errors. This error measure can be calculated with the sum of the squared difference between actual and desired outputs. These parameters are either estimated by using a back propagation algorithm or with a mixture of back propagation and least square method, the combined approached is called hybrid approach.

A-4.3 Constraints of anfis

ANFIS is too complicated as compared to simple fuzzy inference systems and all fuzzy inference systems do not support it. Only the sugeno-type systems support ANFIS and it must possess the following properties:

- The sugeno type systems must be of zeroth or first order.
- Only the single output is obtained by weighted average defuzzification. All output membership functions can either be linear or constant and must have the same type.
- Rules cannot be shared. Every rule corresponds to one membership function only and vice versa so the number of rules and output membership functions are equal.
- All rules have equal unity weight.

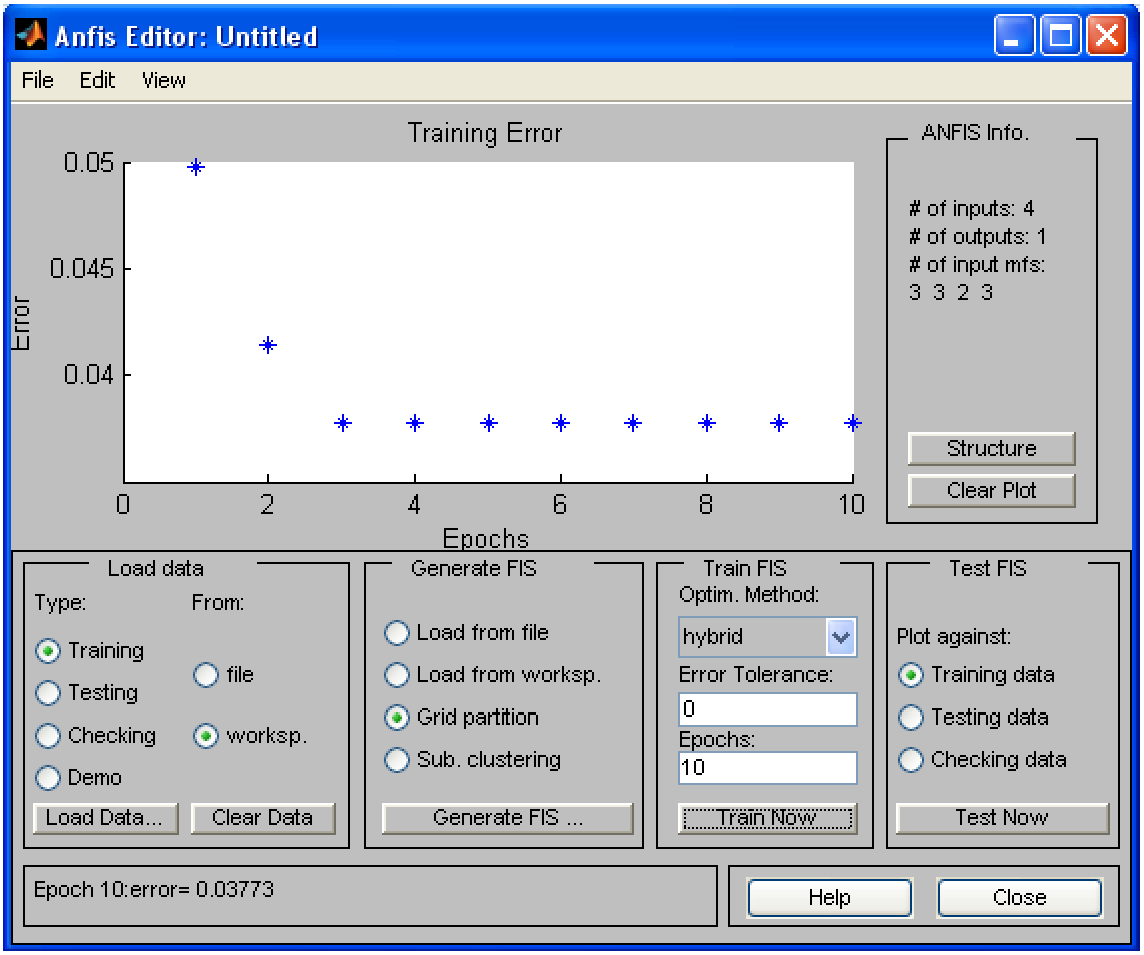
ANFIS doesn’t allow all the customization options that are available in general fuzzy inference systems. You must use the membership and defuzzification functions that are provided else you can’t make your own.

Sugeno-type systems can be created, trained and test by using ANFIS editor. Following command is used to start ANFIS editor.

```
anfisedit
```

Figure A-1 shows the ANFIS editor window contain four different areas. Following task are supported by GUI:

1. Loading, Plotting, and Clearing the Data
2. Generating or Loading the Initial FIS Structure
3. Training the FIS
4. Validating the Trained FIS



ANFIS Info.
# of inputs: 4
# of outputs: 1
# of input mfs:
3 3 2 3

Load data
Type: ● Training ○ Testing ○ Checking ○ Demo
From: ○ file ● worksp.

Generate FIS
○ Load from file
○ Load from worksp.
● Grid partition
○ Sub. clustering

Train FIS
Optim. Method: hybrid
Error Tolerance: 0
Epochs: 10

Test FIS
Plot against: ● Training data ○ Testing data ○ Checking data

Epoch 10:error= 0.03773

Figure A-1: ANFIS editor window

A-4.4 Loading, Plotting, and Clearing the Data

The first step towards building a FIS is to load a training data set that consists of desired inputs and output data. That data set is a table of two or more columns in which the last column represents the output as shown in table A-1.

Table A-1 : Training Data

tf	idf	zone	overlap	Score
0.005586592	0.324119469	0.5	1.096710205	0.000992919
0.005882353	0.324119469	1	1.096710205	0.002090971
0.009009009	0.324119469	1	1.096710205	0.003202389
0.009803922	0.324119469	1	1.096710205	0.003484952
0.014388489	0.324119469	0.5	1.096710205	0.002557303
0.016304348	0.324119469	1	1.096710205	0.005795627
0.018867925	0.324119469	1	1.096710205	0.006706889
0.022222222	0.324119469	0.5	1.096710205	0.003949613
0.025641026	0.324119469	1	1.096710205	0.00911449
0.026086957	0.324119469	1	1.096710205	0.009273003
0.02739726	0.324119469	1	1.096710205	0.009738771
0.031007752	0.324119469	1	1.096710205	0.011022175
0.005555556	0.432460612	16	1.49271137	0.024815123
0.005681818	0.432460612	2	1.587962963	0.007803766
0.006451613	0.432460612	16	1.49271137	0.028817562
0.006535948	0.432460612	1	1.49271137	0.004219208
0.007407407	0.432460612	0.5	1.49271137	0.002390885
0.007407407	0.432460612	4	1.248975876	0.016003937
0.007407407	0.432460612	16	1.49271137	0.033086831
0.007692308	0.432460612	2	1.587962963	0.010565099
0.008	0.432460612	2	1.587962963	0.010987703
0.008064516	0.432460612	2	1.587962963	0.011076313
0.008695652	0.432460612	1	1.49271137	0.005613382
0.00877193	0.432460612	16	1.49271137	0.039181774
0.009345794	0.432460612	16	1.49271137	0.041745067
0.010526316	0.432460612	2	1.587962963	0.014457504
0.010582011	0.432460612	2	1.587962963	0.014533999
0.010638298	0.432460612	16	1.49271137	0.047518321
0.012048193	0.432460612	2	1.587962963	0.016547745
0.015748031	0.432460612	0.5	1.49271137	0.005082983
0.015748031	0.432460612	4	1.248975876	0.034024118
0.015748031	0.432460612	16	1.49271137	0.070342082

Following steps are involved to load the dataset in Load data section of ANFIS GUI.

1. Choose the data type either as training, testing or checking.
2. Select data from MATLAB workspace of from a file.
3. Load the data.

After the data is loaded in a system it will be displayed in the plot. Blue color diamond, circles and plus symbols represents training, checking and testing data.

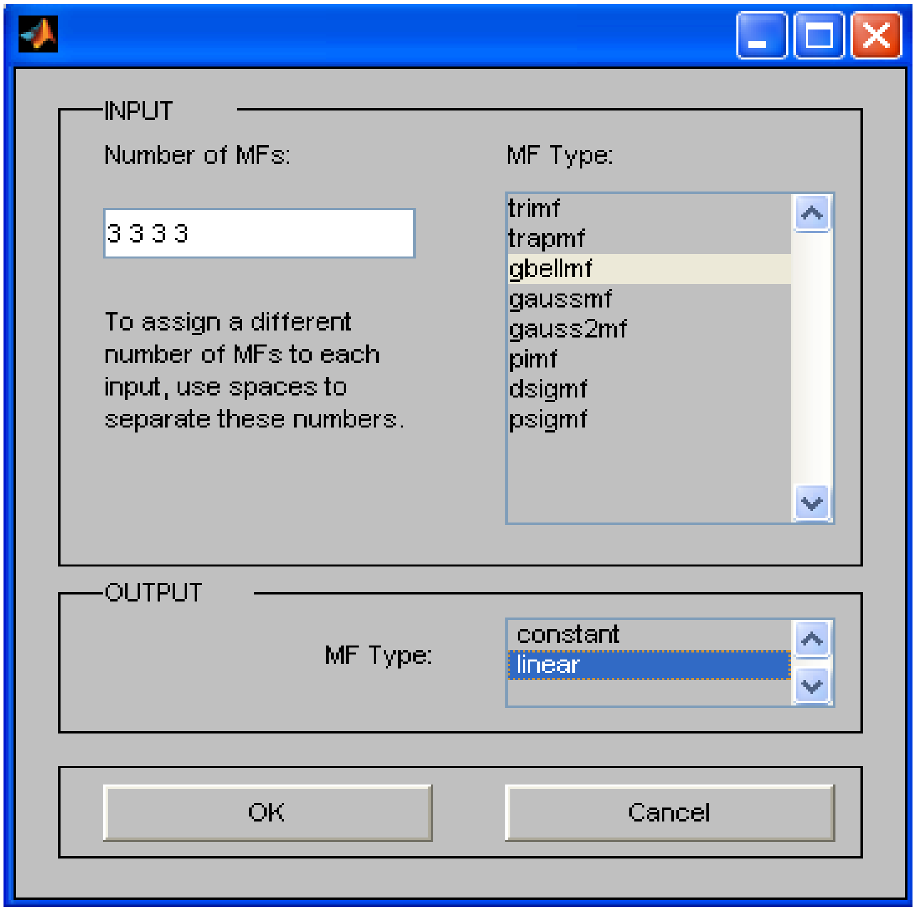
Following steps are used to clear the data from ANFIS GUI and from the plot:

1. Choose the respective data type from the Load data section of GUI.
2. Click Clear Data.

A-4.5 Generating or Loading the Initial FIS Structure

The next step involved before training the fuzzy inference system is to build the initial FIS structure. Perform the following task to specify the initial structure.

- From the MATLAB workspace, Load Sugeno-type FIS.
- Select one of the following partitioning method to generate the initial fuzzy inference system model
  - Grid partition— Creates a Sugeno-type FIS having single-output using grid partitioning.
  - Sub. clustering — Use subtractive clustering to create starting model for ANFIS training.
- Select the number of membership function for each input variables by giving single space and choose the membership functions types as shown in figure A-2.



INPUT

Number of MFs: 3 3 3 3

To assign a different number of MFs to each input, use spaces to separate these numbers.

MF Type: gbellmf

OUTPUT

MF Type: linear

Figure A-2: Input and output MF types

Click on Structure to see initial structure of FIS model in graphical representation as shown in Figure A-3.

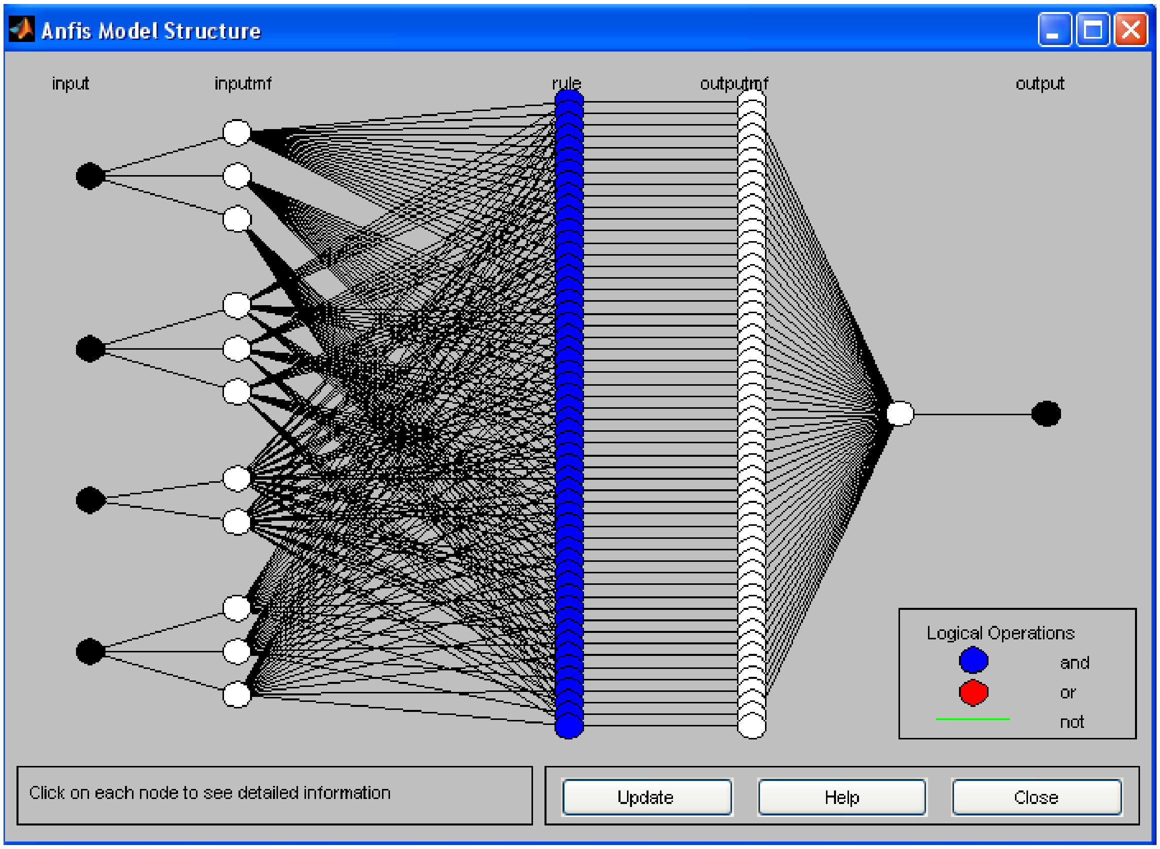


Figure A-3 : ANFIS Model Structure

A-4.6 Training the FIS

The next stop involved in building ANFIS is to train the FIS. Following steps shows how the FIS is trained.

1. select backpropaga or hybrid as the optimization method in Train FIS portion of GUI.

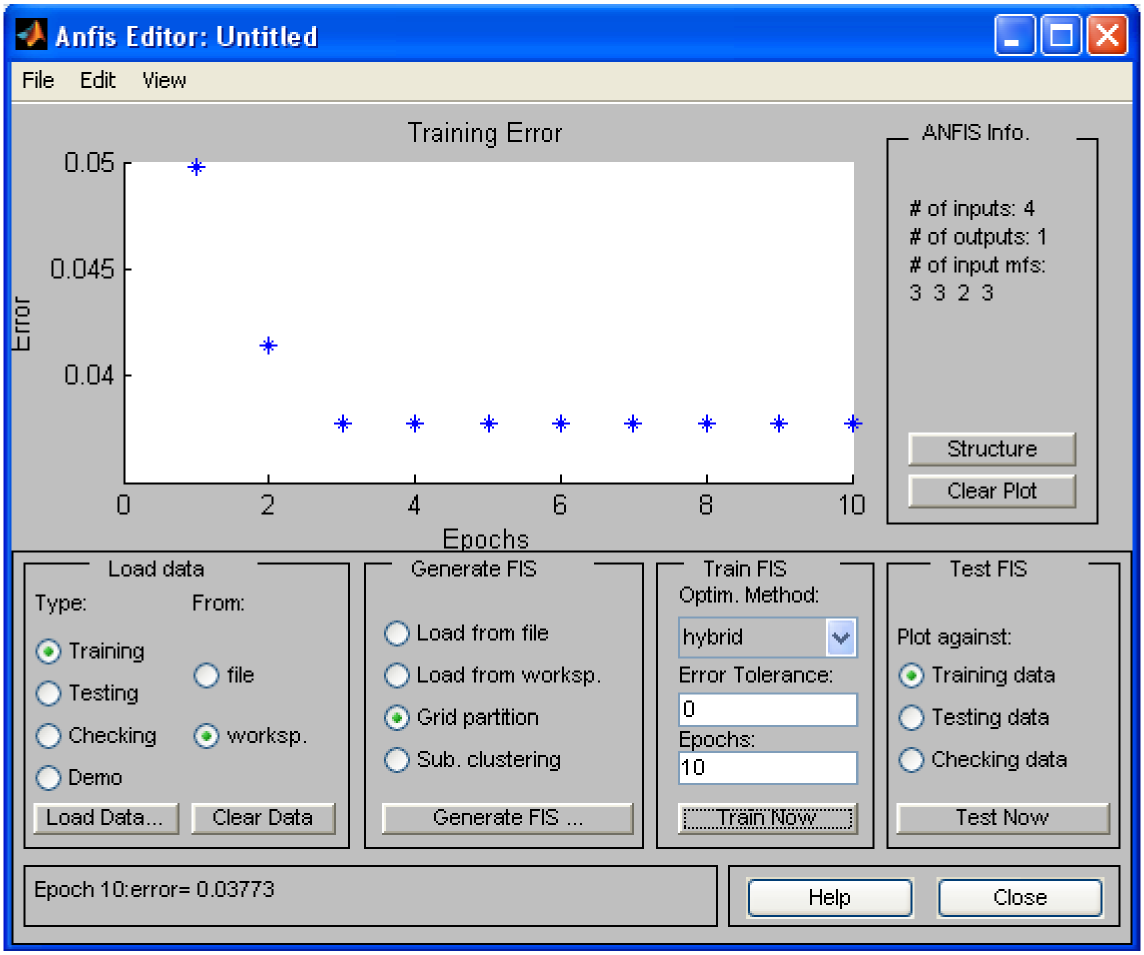
   The optimization method best estimates the parameters for membership functions. The hybrid approach is a combination of backpropagation gradient and least-squares method.

2. Enter the Error Tolerance and Epochs as the stopping criteria for training.

Until either the training error goal is reached or the maximum epoch is achieved, the training process continues.

3. Click Train Now to train the FIS.

   The error plots will be displayed and the parameters for membership function are adjusted after this action as shown in Figure A-4.



# of inputs: 4
# of outputs: 1
# of input mfs:
3 3 2 3

Type: ● Training ○ Testing ○ Checking ○ Demo

From: ○ file ● worksp.

○ Load from file
○ Load from worksp.
● Grid partition
○ Sub. clustering

Optim. Method: hybrid

Error Tolerance: 0

Epochs: 10

Plot against: ● Training data ○ Testing data ○ Checking data

Epoch 10:error= 0.03773

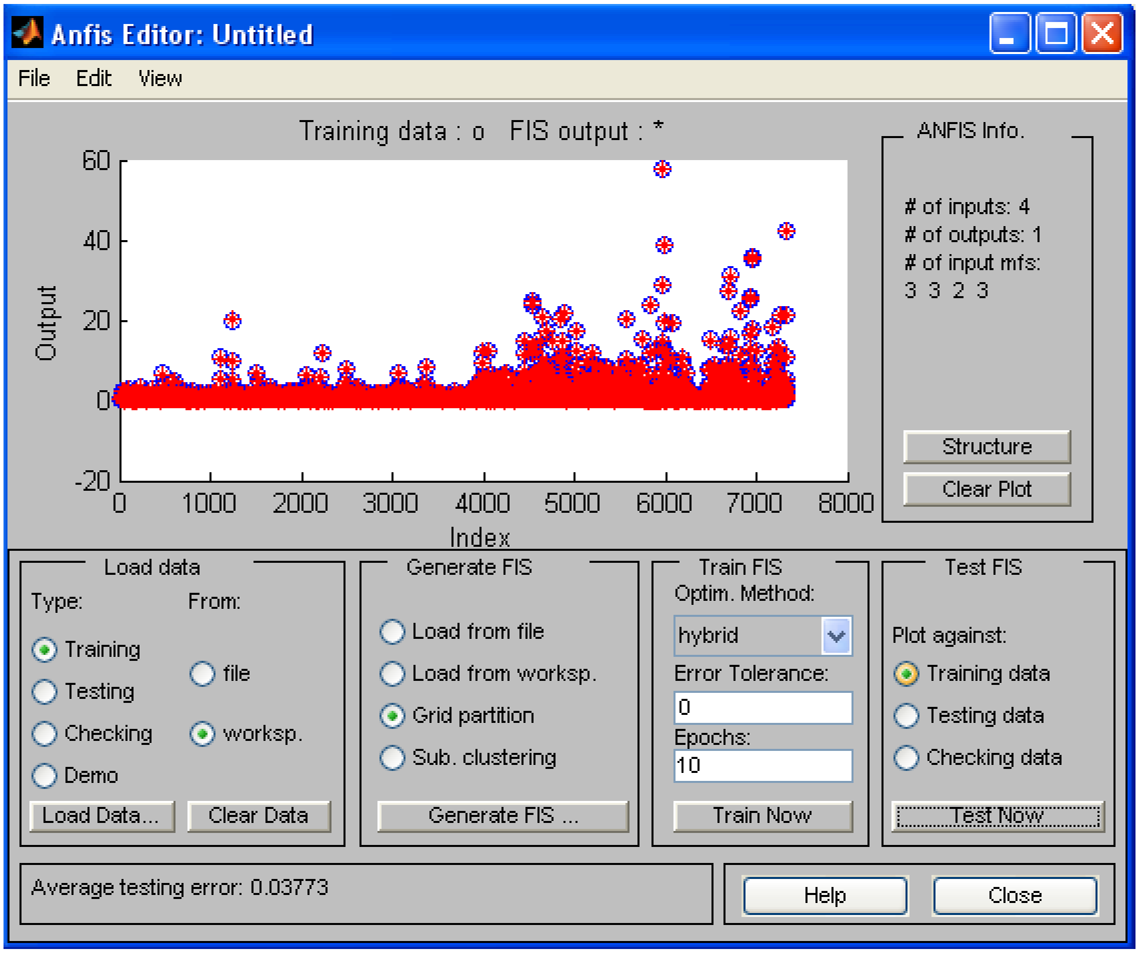
Figure A-4: ANFIS Editor

A-4.7 Validating the Trained FIS

The last step involved after training the FIS is to validate the model. To validate the FIS

1. Load Testing or Checking data.

2. Click Test Now, This action will plot the test data shown in red as shown in figure A-5.



# of inputs: 4
# of outputs: 1
# of input mfs:
3 3 2 3

Load data — Type: ● Training ○ Testing ○ Checking ○ Demo; From: ○ file ● worksp.

Generate FIS: ○ Load from file ○ Load from worksp. ● Grid partition ○ Sub. clustering

Train FIS — Optim. Method: hybrid; Error Tolerance: 0; Epochs: 10

Test FIS — Plot against: ● Training data ○ Testing data ○ Checking data

Average testing error: 0.03773

Figure A-5: ANFIS Editor (Validating Data)

Appendix B

Appendix B : Word Net

WordNet is a large words of English language lexical database. English words are classified into synonyms sets called synsets, gives concise, common definitions, and keep trace of several semantic relations between these synsets. Synsets are connected through conceptual-semantic and lexical association. It serves two reasons: one is to make a combination of thesaurus and dictionary that is intuitively more utilizable, and the other is to assist automated text analysis and artificial intelligence applications. WordNet's is a useful tool for natural language processing (NLP) and computational linguistics. The software tools and database are freely available for download and use under a BSD style license.

A lexical database is a lexical resource (database consisting of one or several dictionaries) that has an associated software environment database which allow access to its contents. The database may be custom-designed for the lexical information or a generic database into which lexical information can be entered.

Information usually stored in a lexical database involve lexical category and synsets, as well as semantic relations between numerous sets of words.

B-1 Java API for WordNet Searching (JAWS)

Java applications uses an API JAWS (Java API for WordNet Searching) that provides the facility to retrieve data from WordNet database. Wordnet versions 2.1 and higher are supported by this fast and simple API and can be used with Java jdk versions 1.4 and higher.

B-2 Configuring Wordnet with your Application

The applications must have to perform these steps to use JAWS:

1. Download and install the full version of wordnet.
2. Download the JAR (Java Archive) file contains the compiled JAWS code.
3. Add the path of downloaded JAR file in JVM (Java Virtual Machine) PATH environment variable.

B-3 Specifying the Database Directory

The Wordnet installation directory contains subdirectory dict in which Wordnet database files can be found. i.e. the directory C:\WordNet-3.0\dict\ contains the database files for wordnet if it is installed in directory :\WordNet-3.0\.

The wordnet.database.dir property can either be set externally or done through code. The System class has a method setProperty() used to set wordnet.database.dir property from with in the code as in following example:

```
System.setProperty("wordnet.database.dir", "C:\WordNet-3.0\dict\");
```

But to set the property externally depends on the way the application is executing, either from command line or from the IDE. The IDE's provides the support for setting the system properties, so these properties need to be set from IDE if you have to run your code with in IDE. For example, "VM Arguments" are specified in eclipse and the list of arguments will include the entries to run the code:

```
-Dwordnet.database.dir=C:\WordNet-3.0\dict\
```

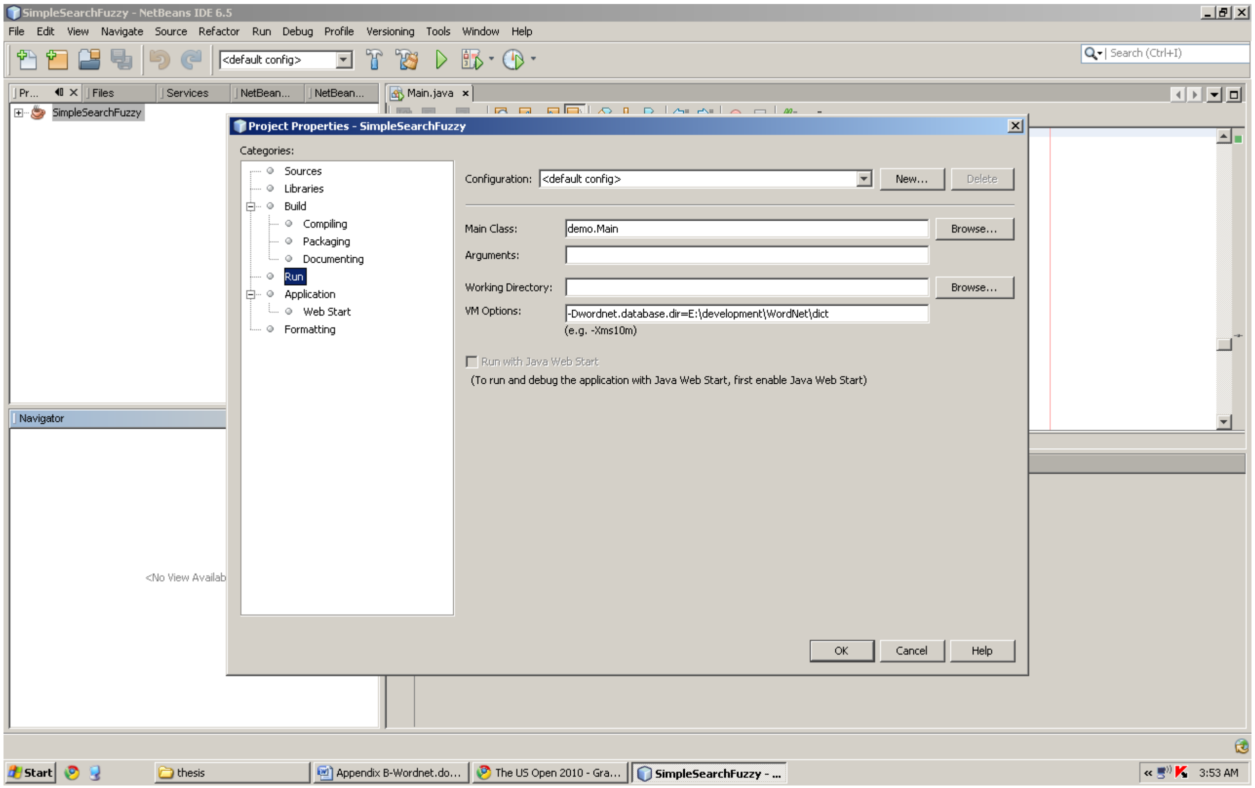
The –D option can be used as an alternate, if you are using command line to run your application.

B-4 Starting Your Application

Consider the following:

- Let the JAR file is downloaded in your C:\myApp\src directory that contains the windows based executable JAWS code.

- The database files exists in C:\WordNet-3.0\dict\ directory as shown in Figure B-1 because we have installed the wordnet in C:\WordNet-3.0\ directory.



Project Properties - SimpleSearchFuzzy

Configuration: <default config>

Main Class: demo.Main

Arguments:

Working Directory:

VM Options: -Dwordnet.database.dir=E:\development\WordNet\dict

(e.g. -Xms10m)

□ Run with Java Web Start

(To run and debug the application with Java Web Start, first enable Java Web Start)

Figure B-1: Path setting of Wordnet for application

You can start a Java Virtual Machine from the command line, it assumes that a class called MyApp that has the main method as shown below:

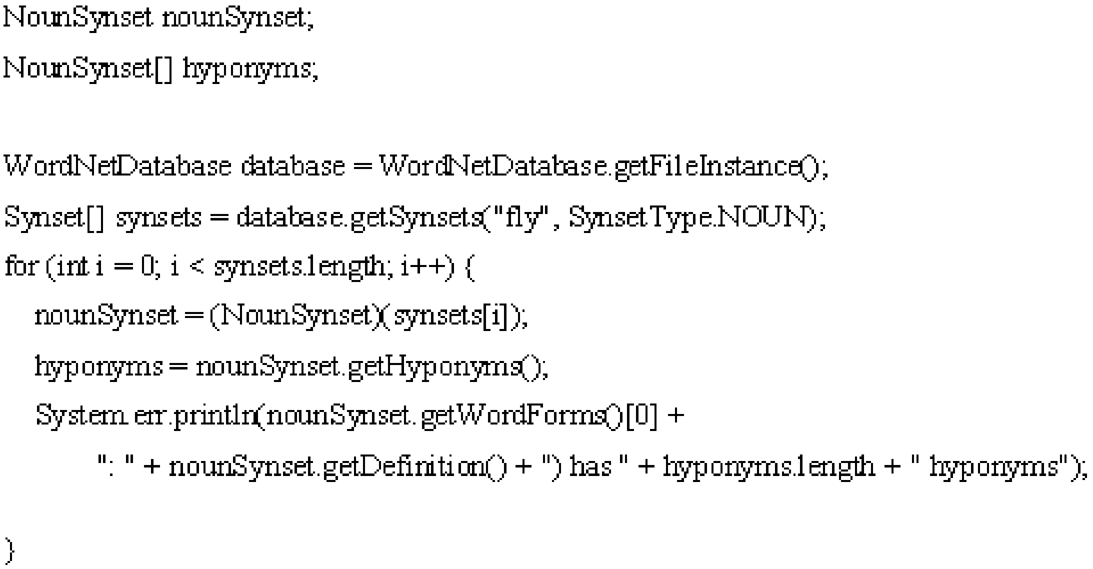
```
java -classpath .;C:\mywork\code\jaws-bin.jar –D wordnet.database.dir=C:\WordNet-3.0\dict
MyApp.
```

B-5 Getting Started With the API

After the application starts, the instance of Wordnet needs to be created first by using JAWS by using following code with the assumption that the classes from package edu.smu.tspell.wordnet must be imported:

WordNetDatabase database = WordNetDatabase.getFileInstance();

When the environment is ready after doing all the settings, the synonyms or synsets from the database can be retrieved as shown in example below. The given example search from database all synsets for the word fly and displays its first normal form with the number of related hyponyms with their description:



```
NounSynset nounSynset;
NounSynset[] hyponyms;

WordNetDatabase database = WordNetDatabase.getFileInstance();
Synset[] synsets = database.getSynsets("fly", SynsetType.NOUN);
for (int i = 0; i < synsets.length; i++) {
    nounSynset = (NounSynset)(synsets[i]);
    hyponyms = nounSynset.getHyponyms();
    System.err.println(nounSynset.getWordForms()[0] +
            ": " + nounSynset.getDefinition() + ") has " + hyponyms.length + " hyponyms");

}
```

Appendix C

Appendix C : Porter Stemmer

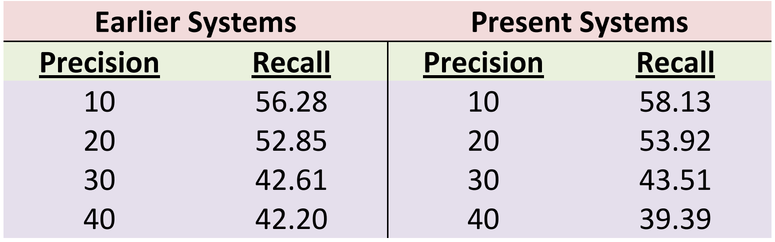
Stemming is the task to get the root or stem of a word. It can be achieved either by using a stem dictionary or by using suffix list for suffix stripping. For every item in suffix list a criteria is mentioned how and when to strip a word. Porter stemmer is an algorithm for suffix stripping to get the root of a term called stem. eliminating suffixes to get stem is an operation which is particularly beneficial in the field of information retrieval. In the field of Information Retrieval, there is a large collection of documents called corpus, each document is represented by words.

In general terms have similar meanings having common stem, for instance

SELECT
SELECTED
SELECTING
SELECTION
SELECTIONS

The advantage of using stripping list over stem dictionary is that it is fast, simple and small in programming code. The efficiency of retrieval system will be improved if such type of terms groups reduced to a single term, and hence in overall the total number of terms in the system is reduced that results in reducing the size and complication of data in system.

Table C-1 : Comparison of Recalls at fixed Precisions



Earlier Systems		Present Systems	
Precision	Recall	Precision	Recall
10	56.28	10	58.13
20	52.85	20	53.92
30	42.61	30	43.51
40	42.20	40	39.39

Earlier Systems		Present Systems	
50	39.06	50	38.85
60	32.86	60	33.18
70	31.64	70	31.19
80	27.15	80	27.52
90	24.59	90	25.85
100	24.59	100	25.85

C-1 Algorithm

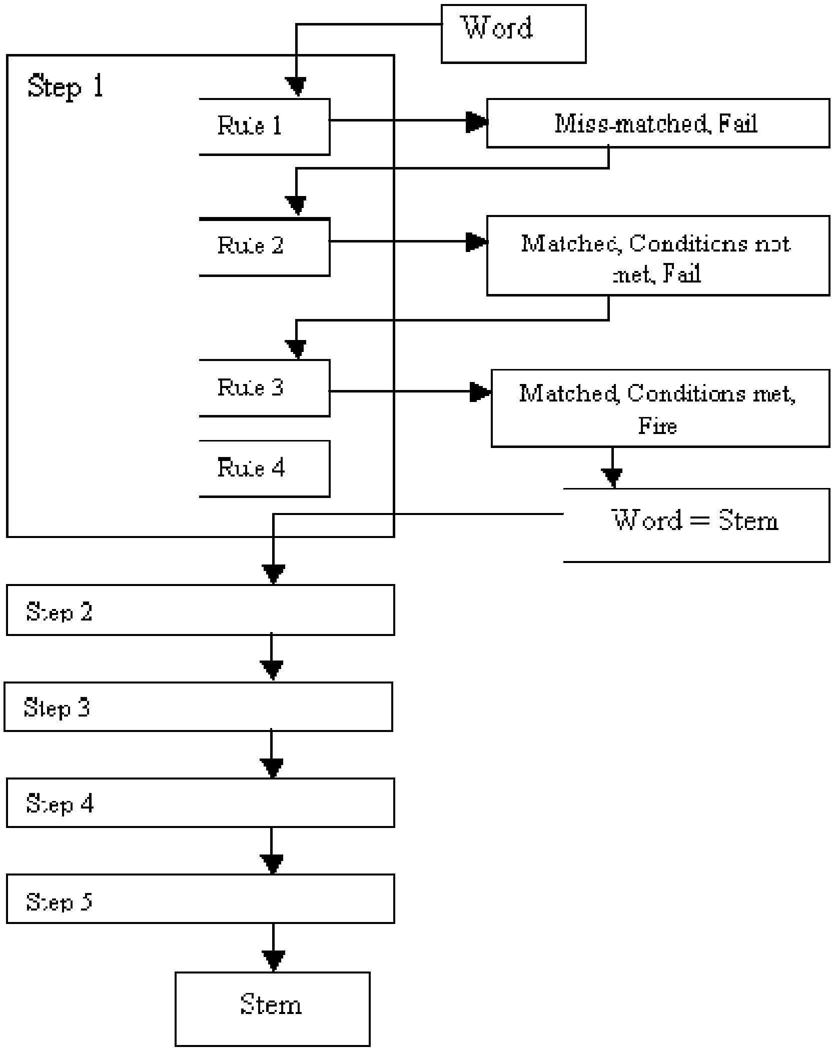


Figure C-2 Flow Chart Porter Stemmer Algorithm