Information Retrieval from Digital Libraries Using Probabilistic - Possibilistic Inferences

K R Chowdhary

Abstract

An Information Retrieval-cum-Extraction system for retrieving information from Digital Libraries using combination of Bayesian Probabilistic and fuzzy logic based possibilistic inferences has been developed and tested. The proposed method resolves the true similarity between documents and information need specified in the form of user query using fuzzy techniques. Final results have been found to be impressive and are close to an idealistic situation.

Keywords: Information Retrieval, Information Extraction, Probabilistic, Possibilistic, Bayesian Inference Networks

1. Introduction

Digital libraries (DLs) contain enormous volume of documents, in order of millions, and even more. An interested user specifies his need in terms of certain keywords to retrieve the document(s) fulfilling this need. Retrieving a relevant document is a big challenge, due to volume of text - as searching the entire text in real-time is non-feasible, due to inherent ambiguity in the text, and due to lack of any structure in the language text [Chowdhary and Bansal, 2001].

A Probabilistic Information Retrieval (IR) system ranks the documents in decreasing order of their probability of relevance to the user's information needs, and a probabilistic Information Extraction (IE) system locates the chunks of desired information based on their probability of relevance and browses them from the documents already retrieved. A fuzzy logic based possibilistic approach takes care of inherent vagueness due to imprecise representation of information. A probabilistic - possibilistic based IR system has been considered as a better approach compared to other methods due to their theoretical soundness. One major difficulty in probabilistic IR method is to find a suitable model for evaluating relevance of documents to user needs, which is theoretically sound and computationally efficient. The approach suggested in this paper makes use of Bayesian networks - an extension of the basic theory of probability, for representation of dependencies [Chowdhary, 2004], [Chowdhary and Bansal 2006].

A conceptual model for representation of documents and queries has been presented. This is followed by necessary derivations for probabilistic – possibilistic method using Bayesian inference networks’ and fuzzy logic for IR. These methods have been later applied for document retrieval from DL and extraction of information from the retrieved relevant documents. Following essential pre-conditions are necessary:
(i) Retrieval accuracy is dependent on the representation of queries and documents, and not directly on the queries and documents.

(ii) Representation of queries and documents are plagued by a variety of uncertainties.

Figure 1 Conceptual Model for IR.

2. Conceptual Model for IR

A conceptual probabilistic model [Crestani, 1998] shown in figure 1, has an event space, represented by \( Q \bowtie D \), where \( Q \) represents all the set of queries, and \( D \) the set of all the documents in the DL. The queries and documents are represented by descriptors, each of which is a set of terms or keywords. Each descriptor is a binary valued vector, in which each element corresponds to a term or keyword. A query is an expression of information need, which is regarded as unique event, i.e., two same queries are treated as different events.

Basic objects of an IR system are – a finite set of documents \( D = \{d_1, d_2, d_3, \ldots\} \) and queries (information needs) \( Q = \{q_1, q_2, q_3, \ldots\} \) submitted to the system. Let us consider that \( R \) be a set of possible relevance judgments for documents set \( D \) and queries \( Q \). In case of Boolean IR, \( R = \{\_R, R\} \), i.e. a document is either relevant or not. Hence, relevance relationship between the query set and document set can be regarded as a mapping \( r: Q \bowtie D \mapsto R \). However, IR systems do not deal directly with the documents and the queries, but with their representations. For example, index terms are representations for a document and Boolean expression of terms are for a query. Let \( Q' \) and \( D' \) be the representations of queries and documents, respectively, and \( a_Q \) be a mapping from \( Q \) to \( Q' \) and \( a_D \) a mapping from \( D \) to \( D' \). Thus, two documents with same set of terms will be mapped onto the same representation.

To make the models more general, a further mapping is introduced from representation to descriptions of the objects. For example, queries and document representation in the form of index terms may be described by supplementing with weights of each term. Let these descriptions be \( Q'' \) and \( D'' \) for query set and document set, respectively. Let the corresponding mappings be \( b_Q \) and \( b_D \) respectively. Thus, the relevance relation between query and document set should be based on their description. The new value of the relevance function is therefore, represented by the expression \( r: Q'' \bowtie D'' \mapsto R \),
which maps query-document pair to a ranking value or relevancy value - a real number. In response to a query $q_j \in Q$, documents $d_k \in D$ are ranked according to descending order of $r(q_j', d_k')$. The function of an IR system, which ranks the documents in the order of their relevancy for a query $q_j'$ is to calculate relevance and rank every document $d_k'$ in the collection of documents $D$. However, for the sake of simplification, the description and representation have been treated identical, and both are represented in the form of set of terms.

The probability that a document $d_k$ is relevant to the query $q_j$, can be expressed by $P(R|q_j, d_k)$ as per the conditional probability [Trivedi, 1988]. A precise definition of probability of relevance depends on the definition of relevance. The relevance is to some extent subjective and depends on number of variables concerning - the document, the user, and the information need of the user. A perfect retrieval is far from achievable, however, optimal retrieval can be defined for probabilistic IR, because it can be proved theoretically with respect to representations (or descriptions) of documents and information needs [van Rijsbergen, 1979].

Let the queries and documents are described by sets of index terms. Let $T = \{t_1, t_2, \ldots, t_n\}$ denotes the set of terms in the collection of documents in DL. A query $q_j$ is a subset of terms belonging to $T$. Similarly a document $d_k$ is a subset of terms belonging to $T$. For the purpose of retrieval, each document is described with the presence/absence of these index terms. Therefore, any document $d_k$ is represented with a binary vector:

$$\bar{x} = (x_1, x_2, \ldots, x_n)$$  \hspace{1cm} (1)

where $x_i = 1$ if $t_i \in d_k$, and for $t_i \notin d_k$, $x_i = 0$. A query $q_j$ is represented in the same manner. Main task of an IR system based on relevance model is to evaluate the probability that a document being relevant. This can be done by estimating the probability $P(R|q_j, d_k)$, for every document $d_k$ in the collection. Since relevancy for all the documents is evaluated for a single query, the term $q_j$ can be dropped, and relevancy can be expressed by the Bayes theorem as follows [Trivedi, 1988]:

$$P(R|\bar{x}) = \frac{P(\bar{x}|R)P(R)}{P(\bar{x})}$$  \hspace{1cm} (2)

where,

$P(R|\bar{x})$ is probability of relevance, given that the document is $\bar{x}$.

$P(\bar{x}|R)$ is probability of randomly selecting the document with description $\bar{x}$ from the set $R$ of relevant documents,

$P(R)$ called prior probability of relevance, is probability that a document randomly selected from the entire collection is relevant,

$P(\bar{x})$ is probability that the selected document has description $\bar{x}$. It is determined as the joint probability distribution of the $n$ terms with in the collection.
3. **Probabilistic-Possibilistic Inference Model**

The basis for use of Bayesian probabilistic inference network [Fung and Favero, 1995; Turtle and Croft, 1990; Darwiche, 2003] - an extension to probability-based retrieval, is a Directed Acyclic Graph where nodes represent propositional variables or constants and edges represents the dependency relationship between these propositions. If a proposition corresponding to a node \( p \) “causes” or implies the proposition represented by node \( q \) “effect”, then it can be represented by a directed graph from \( p \) to \( q \). The node \( q \) contains a link matrix that specifies \( P(p|q) \) for all possible values of two variables. When a node has multiple parents (for query node), the link matrix specifies the dependence of that node on the set of parents and characterizes the dependence relationship between that node and all nodes representing potential causes. Given a set of prior probabilities for the roots of this graph (i.e., documents), these networks can be used to compute the probability of belief associated with all the remaining nodes. Figure 2 shows a document \( d_i \) corresponding keywords \( t_1, \ldots, t_n \) and a queries submitted.

![Figure 2. Basic Inference Network Model.](image)

The inference network associates random variables with the documents, index terms, and user queries. Multiple evidences of query terms in the document’s representation for a given query are combined to estimate the probability that a document satisfies the user’s information need. A document’s variable associated with the document \( d_i \) represents the event of observing the document. The index terms and document variables are represented as nodes in a directed graph. Edges are directed from document nodes to the index term nodes showing that observation of document yields the improved belief on its term nodes. The random variable associated with the user query, also shown by node, models the event that the information request specified by the query has been met. The dependence through the direction of arrows shows that the belief in the query node is function of the beliefs in the nodes associated with the query terms. In a particular case shown in figure 2, document \( d_i \) has \( t_j, t_i, t_j \), and \( t_j \) as its index terms. Similarly, the query \( q_i \) is shown to be composed of query-terms \( t_j, t_i \) and \( t_j \), hence, \( q_i = t_j \cup t_i \cup t_j \), and \( q_2 = t_j \cup t_j \).
Each set of arcs pointing to a node represent a probabilistic dependence between the node and its parents. A Bayesian network represents, through its structure the conditional dependence relations among the variables in the network. These dependence relations provide the framework for retrieving the probabilistic information.

For the purpose of retrieving information, a user specifies one or more topics of interest by way of identifying some document features to be used as evidence for the topics of interest. The IR task using Bayesian inference network is be specified in the form of an algorithm shown in figure 3. The task requires building of inference network for representation of query terms and document features (i.e. terms), and computation of posterior probabilities based on the prior probability of the document.

Algorithm 1: Bayes_inference
1. Build the network representing the query
2. Score each document in the repository as follows:
   a. Extract the features from the document
   b. Label the features in the network
   c. Compute the posterior probabilities of relevance
3. Rank the documents according the posterior probabilities.

Figure 3.: Bayesian Inference based IR.

The term weighting criteria [Sparck Jones, 1972] has been used for feature identification of documents in the Bayesian networks shown in figure 4. The topic of interest (i.e., a query) is shown by terms t_1, t_2 (for example, q = “house loan”, where t_1 = house, t_2 = loan). Hence, there can be one or more document features to examine. Nodes t_i represents the event “the document is related to topic t_i”. The nodes t_11, ..., t_in are document futures to be examined for the topic t_1, and t_21, ..., t_2n are document features to be examined for topic t_2. Thus, nodes t_i represents the event that “feature t_ij is present in the document”. Here, an assumption is made that t_1, t_2, ... have no dependence with each other (shown by absence of arcs between them), similarly t_11, t_12, ..., and t_21, t_22, ... are also assumed to be independence of each other.

The network model shown in figure 4 requires two sets of probabilities to be computed:
(i) Prior probability P(d_i) that the document d_i is relevant to the query topic, and
(ii) The conditional probability P(t_k | d_i) for each feature t_k for a given each topic t_i in query, which shows that - “what is probability that feature t_k is present in a document, given that the document is relevant to query topic”? Next, the task of IR system is to compute the posterior probability P(d_i | t_11, t_21, ..., t_in), which means - “what is probability that document d_i is relevant, given that we have observed the presence or absence of all the features t_i for each document d_i. For the above inference network, the Bayes theorem can be directly applied to obtain the posterior probability, as follows:

\[
P(d_i | t_{11}, ..., t_{in}) = \frac{P(d_i) P(t_{11}, ..., t_{in}) | d_i)}{P(t_{11}, ..., t_{in})}
\] (3)
224

where \( i = 1, \ldots, N \) are set documents in the repository. The topic \( t_i \) has been referred as query terms for a given query, and document features \( t_{ij} \) have been referred as synonyms / related words to the query term.

4. Experimental Results

Given a query \( Q = \{q_1, q_2, \ldots, q_m\} \), where \( q_1, q_2, \ldots, q_m \) are keywords in the query, and documents \( D = \{d_1, d_2, \ldots, d_n\} \), where \( N \) is the total number of documents in the DL, with \( n_i \) as size of each document, it is required to find the document \( d_i \), \( i = 1, 2, \ldots, N \), to which the query is related in the maximum relevance sense.

4.1 Approach: Using Bayes theorem, the probability of the overlap of keywords between the query terms (set \( q \)) and document terms (\( d_i \)) is expressed by [Trivedi, 1988]:

\[
P(d_i \cap q) = P(d_i | q)P(q) = P(q | d_i)P(d_i)
\]

where

- \( P(d_i | q) \) is probability that document \( d_i \) is observed, given that query is \( q \), (called, posterior probability),
- \( P(q) \) is probability of occurrence of query \( q \),
- \( P(q | d_i) \) is probability that query is \( q \), given that document observed is \( d_i \),
- \( P(d_i) \) is probability of occurrence of the document \( d_i \), called prior probability.

Thus, probability that document \( d_i \) is observed, given that query is \( q \), can also be expressed as

\[
P(d_i | q) = \frac{P(q | d_i)P(d_i)}{P(q)}
\]

(5)

Since \( P(q) \) is common for the evaluation of expression for \( P(d_i | q) \) for every document \( d_i \), dropping \( P(q) \) will not effect the ranking order of the document \( d_i \). The new value for \( P(d_i | q) \) we refer as \( RF(d_i | q) \), where \( RF \) is Relevance function for ranking of document \( d_i \) for query \( q \). Thus expression in equation (5) becomes that of (6).

\[
RF(d_i | q) = P(q | d_i)P(d_i)
\]

(6)
First considering that there is only one term \( t_i \) in \( q \), \( P(t_i \mid d) \) is probability of \( t_i \) in \( d \) given that \( d \) has been observed, and \( P(d) \) is probability of observing document \( d \) in the entire lot. That is,

\[
RF(d_i \mid t_1) = P(t_1 \mid d_i)P(d_i)
\]  

(7)

Similarly, it can be computed for each query term \( t_j \). Now, let us consider that \( q \) comprises \( t_1, t_2, \ldots, t_m \). Thus,

\[
RF(d_i \mid t_1, t_2, \ldots, t_m) = P(t_1 \mid d_i)P(t_2 \mid d_i)\ldots P(t_m \mid d_i)P(d_i)
\]

or

\[
RF(d_i \mid t_1, t_2, \ldots, t_m) = \prod_{j=1}^{m} P(t_j \mid d_i)P(d_i)
\]  

(8)

For the sake of simplicity, it is assumed that all the documents are equally likely, thus, \( P(d_1) = P(d_2) = \ldots = P(d_N) \). With this simplification the term \( P(d_i) \) can be dropped from equation (8), being a common multiplier in all the document’s expressions. Now, the relevance function can be computed for every document \( d_i \) for a given query \( q = t_1, t_2, \ldots, t_j, \ldots, t_m \), as follows.

\[
RF(d_i \mid t_1, t_2, \ldots, t_m) = \prod_{j=1}^{m} P(t_j \mid d_i), \quad i = 1, \ldots, N
\]  

(9)

Using fuzzy membership concept, the equation (9) is modified by introducing a fuzzy membership function \( \mu_j \) for each query term \( t_j \). Thus,

\[
RF_{\mu}(d_i \mid t_1, t_2, \ldots, t_m) = \prod_{j=1}^{m} \mu_j P(t_j \mid d_i), \quad i = 1, \ldots, N
\]  

(10)

The query \( q \), comprising \( m \) number of terms can be expressed by \( q = t_1 \cup \ldots \cup t_j \cup \ldots \cup t_m \). When all the \( k \) number of synonyms and related words for each query term \( t_j \) (i.e., \( \{ t_{j_1}, t_{j_2}, \ldots, t_{j_k} \} \)) are accounted in the document \( d_i \), the weights of each term \( t_j \) in the query is expressed by \( \mu_{j_1} = \mu_{j_1}t_{j_1} + \mu_{j_2}t_{j_2} + \ldots + \mu_{j_k}t_{j_k} \). Considering this, the expression for maximum relevance for document \( d_i \) (with size \( n_i \) words) for query \( q \) is given by

\[
RF_{\mu}(d_i \mid t_1, t_2, \ldots, t_m) = \prod_{j=1}^{m} \left( \frac{\mu_{j_1} + \mu_{j_2} + \ldots + \mu_{j_k}}{\sum_{j=1}^{k} \mu_{j_k}} \right), \quad \text{for } i = 1, \ldots, N \quad \text{documents}
\]

4.2 Computation of Results

Twelve numbers of documents (\( d_{q_1} \) to \( d_{q_2} \)) have been taken for this work as a collection. These are accessed for five queries (\( q_1 \) to \( q_5 \)). Each query comprises two keywords and has been expanded [Carppineto et al, 2001] by supplementing it with additional words, which are either synonyms of keywords or their related words.
Table 1: Query terms, their synonyms, and related words for $q_1$.

<table>
<thead>
<tr>
<th>Query Term</th>
<th>Synonyms with Fuzzy membership weights</th>
</tr>
</thead>
<tbody>
<tr>
<td>House(1)</td>
<td>home(.8), building(.7), residence(.3), dwelling(.2)</td>
</tr>
<tr>
<td>Loan(1)</td>
<td>finance(.8), financing(.8), mortgage(.7), borrow(.5), advance(.4), credit(.3)</td>
</tr>
</tbody>
</table>

Table 1 shows the queries, synonyms, and related words with their fuzzy membership weight (i.e., closeness to the query term). For example, in the case of first query, $q_1 = t_1 \cup t_2 = \text{house} \cup \text{loan}$. The term $t_1$ has been expanded in the form of $t_{11}, \ldots, t_{15}$. Here, $t_1 = t_{11}, t_{12}, t_{13}, t_{14}, t_{15} = \text{house, home, building, residence, dwelling}$. Similarly, $t_2 = t_{21}, \ldots, t_{27} = \text{loan, finance, financing, mortgage, borrow, advance, credit}$. Other queries are: - $q_2 = \text{education} \cup \text{innovation}$, $q_3 = \text{home} \cup \text{budget}$, $q_4 = \text{career} \cup \text{prospects}$, and $q_5 = \text{tax} \cup \text{reforms}$. A program, rf.c (for relevance function) computes the relevance function's value for each document. Following rf command returns the text document names along with relevance functions' values, in decreasing order of relevance function. Only those documents' names appear in which relevance function's value is greater than the threshold (i.e., 20% of maximum).

```
c:\> rf 2 docfiles.txt <cr>
```

<table>
<thead>
<tr>
<th>Document Name</th>
<th>Ranking weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>d03.txt</td>
<td>0.175067 * 1E-04</td>
</tr>
<tr>
<td>d11.txt</td>
<td>0.124068 * 1E-04</td>
</tr>
<tr>
<td>d05.txt</td>
<td>0.090724 * 1E-04</td>
</tr>
</tbody>
</table>

In above, 2 in the command line argument stands for query $q_2$, and docfiles.txt is text file containing names of all text documents in the DL.

Algorithm 2: Relevance_function

1. for each document $d_i$, $i = 1, \ldots, N$ do
   a. initialize relevance of $d_i$, $w_{RF_i} = 1$
   b. For each query term $t_j$, $j = 1, \ldots, m$
      i. initialize relevance weight of term $t_j$, $w_{t_j} = 0$
      ii. For each $t_{jk}$ appearance of $t_j$, its synonyms, and related terms in the document $d_i$ do
          a. $w_{t_j} = w_{t_j} + \text{fuzzy weight of } t_{jk}$
          b. $w_{RF_i} = w_{RF_i} \wedge w_{t_j}$
      iii. print $d_i$, $w_{RF_i} / (n_i)^m$

Figure 5: Algorithm for computation of Relevance function of a document.
Table 2 shows values of relevance function for each of the five queries, for each of the 12 documents. The results indicate the ranking of documents based on their relevancy to the queries.

For the query \( q_2 \), only three documents have been returned, the\( d03.txt \) being more closely relevant to the query than \( d11.txt \) and \( d05.txt \). It has been found that documents’ contents show a strong similarity to the queries, in the order of the value of their relevance functions.

The robustness of this method is due to the fact that it returns zero or insignificant value of relevance function for those documents that are not relevant, and hence they can be ignored. In this category are those documents, where only one term from the query has found a match. The evaluation of the proposed method is done using the recall and precision parameters for IR and IE.

Table 3 gives the values of these parameters for the queries \( q_1 \) to \( q_5 \), and their averages for 12 documents. The marginal deviation in precision and recall from 100 percent is due to the fact that some documents’ relevancy is borderline case, due to which they may be considered as relevant or non-relevant when 20% threshold is adopted. The results are close to ideal.
A fairly large number of terms have been incorporated the expanded queries to ensure that no relevant document is missed from retrieval. This has increased the average recall (fraction of relevant documents retrieved) to 93.3%.

However, due to large size of expanded queries in this example, some non-relevant (in fact less relevant) documents have also been retrieved along with maximum number of relevant documents, and this has marginally lowered the precision (relevant document fraction in the retrieved documents), with average precision of 86.6%. Thus, to achieve maximum value of precision as well as recall, there is need of optimum size of expanded queries.

4.3 Information Extraction

Once the relevant document is retrieved (fetched) through the process of IR, it is required to label the relevant text segments in the retrieved relevant documents, through the process of Information Extraction (IE). The IE is a three step process, (i) extraction of those texts segments, (ii) evaluation of their ranking, and (iii) visualization of relevant information text segments in the order of their ranking. It is assumed that segments of these texts are non-overlapping contiguous strings in the form of sentences, each represented by a symbol \( s = \{s_1, s_2, \ldots, s_n\} \) where \( s_i \) are keywords in the current sentence. The relevance ranking of \( s \) for the query \( q = \{t_1, t_2, \ldots, t_m\} \) is represented by \( P(s \mid t_1, t_2, \ldots, t_m) \) and computed similar to the relevance function in equation (10).

First considering \( q = \{t_1\} \), the probability of relevance of sentence \( s \) having \( n \) number of terms, given that query is \( q \), is expressed by,

\[
P(s \mid t_1) = \frac{\mu_{t_1} + \mu_{t_2} + \ldots + \mu_{t_m}}{n}
\]  

(11)

where \( \mu_{t_i} \) is fuzzy relevance relation between query term \( t_i \) and the \( j \)th term (\( s_j \)) in the sentence \( s \). Considering \( m \) number of terms in the query, equation (11) is modified as

\[
P(s \mid t_1, t_2, \ldots, t_m) = \sum_{i=1}^{m} \frac{\mu_{t_i} + \mu_{t_2} + \ldots + \mu_{t_m}}{n}
\]

\[
= \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{\mu_{s_j}}{n}
\]

(12)

where,

- \( m \) is closeness or fuzziness of the \( i \)th term (\( s_i \)) of the current sentence \( s \), with the query term \( t_i \) of query \( q \),
- \( n \) is total number of terms in the current sentence, and
- \( m \) is total number of query terms in the expanded query \( q \).
The presence of query terms in some of the sentences in the retrieved document makes them eligible candidates for relevancy to the query. Higher the occurrence of query terms or their related terms in a sentence, the more strongly it indicates relevance to the information need of the user. Once computed for a given retrieved text document, the value of relevance function \( P(s|t_1, t_2, \ldots, t_m) \) is stored in an array. For final result, the sentences are then displayed (browsed) in the order of relevance function, in merit order of this function’s value, and those less than threshold (20% of the maximum value) are discarded. Longer the sentences, the truncation and rounding off errors will be reduced, and therefore, the result is likely to be more accurate.

Figure 6 shows the algorithm for information extraction and the corresponding program `ie.c` (for Information Extraction) computes the relevance ranking at text segment level for the text document already retrieved through IR. The program `ie.c` is executed with following command format:

```
C:\> ie  query_id  text_documentfile
```

where `query_id` is query number (1 to 5) and `text_documentfile` is the retrieved relevant text file through IR.

For query \( q_1 \), the document `d01.txt` has already been found relevant, where as `d05.txt` has been found non-relevant during fetch phase (table 2). These texts, when processed by IE algorithm (program `ie.c`), following are the results:

---

**Algorithm 3: IE (Information Extraction)**

1. initialize sentences array `tsents[]`, expanded query array `{term, relevance_weight_of_term}`, sentence rank array `{sentence_id, rank}`
2. `text_wordcounter = 0`
3. while `textfile end`
   a. `getchar()`
   b. if word boundary
      store this word into sentences array, `text_wordcounter++`
4. for each text sentence \( s \) in sentence array do
   a. relevance weight of \( s \), `swt = 0`
   b. for each query term \( t \) in query-term-array
      i. search \( t \) in \( s \)
      ii. if match found \( n \) times then
          `swt = swt + n * relevance_weight_of_term`
   c. update rank array for this sentence as \([sentence_id, swt] \)
5. sort sentence in merit order of rank
6. threshold = maximum weight of sentence * 0.20
7. for all the sentences in sentence array do
   if sentence rank is > threshold
      print sentence

---

**Figure 6: Algorithm for Information Extraction.**
A cover for your home loan TIMES NEWS NETWORK
[THURSDAY NOVEMBER 28 2002 12:25:40 PM ]
Mr Raman a senior executive at an MNC walked straight into an insurance office after buying a property
The good news is that housing finance companies and banks which earlier used to lend only 75 85 per cent of the project cost are willing to finance up to 90 per cent
And many are willing to customize the loan to specific needs

5. Discussion and Concluding Comments
It has been demonstrated through experimental results that IR and IE based on probabilistic – possibilistic approach using combination of Baysian inference networks and fuzzy logic provides accurate information retrieval from the texts documents stored in DLs, as well as extraction of information from retrieved documents. The necessary algorithms for determination of relevancy of retrieved documents and extracted information, results, and valuation of results using standard benchmark test – precision and recall, have been presented. The experimental results strongly support the mathematical theory, derived for probabilistic – possibilistic approach for IR and IE.

References

About Author

Dr. K R Chowdhary, Associate Professor and Director, The Department of Computer Science and Engineering & Director University’s computer centre, M.B.M. Engineering College, Faculty of Engineering, J .N.V University, Jodhpur, India.