http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 6 (4): 547-553, 2007
ISSN 1812-5638

Knapsack Based IR System for Bio-Medical Literature Using Similarity Based Clustering Approach

[1]K. Latha, [1]R. John Regies, [1]S. Archana and [2]R. Rajaram
[1]Department of Information Technology, [2]Department of Computer Science and Engineering,
Thiagarajar College of Engineering, Madurai-15, Tamil Nadu, India (South)

Abstract: Massive amounts of biomedical literature are readily available online in many forms. Amounts of valuable knowledge and relationships are embedded in these resources and need to be properly extracted, discovered and utilized. In this paper we propose a Text Mining Framework to extract the Biomedical documents from the Biomedical Literature which is divided into Four distinct stages: 1.Text gathering, 2. Text preprocessing, 3. Clustering based on similarity and 4. Information Retrieval process using two different dynamic programming methods such as Knapsack and Knapsack based proposed method. The goal of this paper is to use Dynamic methods to the four stages and to display results.

Key words: Dynamic programming, information retrieval, knapsack, text preprocessing, text mining framework, similarity

INTRODUCTION

Biomedical information is growing explosively and new useful results are appearing everyday in research publications. An IR system is used to retrieve data applicable to users need, based on queries posed to the systems by the user. The basic objective of this paper is to enable data mining and dynamic programming techniques (Yamamoto *et al*., 2003) to extract knowledge from such documents and to minimize the number of documents to be checked. First we apply the knapsack algorithm to extract knowledge. Based on this algorithm we have proposed our own algorithm to extract knowledge which provides better results than the knapsack method. The retrieved documents are given as the input to clustering phase, there we apply our proposed method (Similarity based Clustering approach). Finally the results are displayed and compared with existing methods.

MATERIALS AND METHODS

Boolean IR (Yoshioka and Haraguchi, 2005): This approach to IR is based on set theory and Boolean algebra. The user query is made up of terms combined using Boolean relations AND, OR and NOT. The ith vector term is set to 1, if the ith unique word is present in the document and 0 otherwise. This approach is limited in number of ways. Firstly the retrieval decision made using this approach is binary: Documents are either relevant to the query, or irrelevant, with no grading scale. This severely limits this method's performance. Also it is difficult for the users to formulate their requests into Boolean expressions.

Vector model of IR (Possas *et al*., 2005): In this method each document is represented as vector, with element i within the vector taking on a value (usually between 0 and 1) based on the presence or absence of the word indexed by i within a global word list. Determining whether a document is relevant for given query involves computing a similarity measure between document and query and this is done by calculating cosine of angle between the two vectors.

The tf. idf family of schemes (Salton and Buckley, 1998): The term frequency component (tf) of a term t_i, for a document d_j is calculated;

$$tf_{i,j} = \frac{frequency_{i,j}}{\underset{l \in terms}{Max}\,[frequency_{i,j}]} \quad (1)$$

i.e., raw frequency of a term divided by the frequency of most common term in the document. The idf, or inter-document frequency component is computed as;

Corresponding Author: K. Latha, Lecturer, Department of Information Technology, Thiagarajar College of Engineering, Madurai-15, Tamil Nadu, India (South) Tel: 0452-2342475

$$idf_i = \frac{\log N}{n_i} \quad (2)$$

Where N is the total number of documents in the collection and n_i is the number of documents in which the term t_i appears. The component of the weighing vector for a document d_j is of the form;

$$w_{i,j} = f_{i,j} x idf_i \quad (3)$$

The query term weights are calculated by Salton and Buckley formula (Salton Buckley, 1998)

$$w_{i,q} = idf_i x (0.5+0.5 X tf_{i,q}) \quad (4)$$

The drawback of the system is, for example when short queries are most common (in web search engine environment), query weighing vectors should be modified to increase the influence of the second component in Eq. 4.

Benefit of chosen methodology: Dynamic Programming (Yamamoto *et al*., 2003) is an algorithm design method that can be used when the solution to the problem can be viewed as result of sequence of decisions. In this method an optimal sequence of decision is obtained by the Principle of optimality. The Principle of optimality states that an optimal sequence of decision has the property that whatever the initial state and decision are, the remaining decision must constitute optimal decision sequence with regard to the state resulting from the first decision.

In Dynamic Programming we have used Knapsack method (Martello, 1990) and based on this algorithm we propose our own Algorithm to produce the efficient results. We have decided the values of x_i, $1 \leq i \leq n$. First we make a decision on x_1, then on x_2, then on x_3 and so on. An optimal sequence of decision maximizes the objective function $\sum p_i x_i$. It also satisfies the constraints $\sum w_i x_i \leq m$ and $0 \leq x_i \leq 1$.

In this paper we present a system called Text Mining Framework, which consists of the following stages;

- Text gathering
- Text preprocessing
- Clustering (Similarity based approach)
- Information Retrieval (Knapsack and Proposed Knapsack)

For the first stage the documents are collected from the existing biomedical databases such as Pub Med Open Access Initiative (PubMed Open Access Initiative, 2004), National Library Of Medicine, MeSH (Nelson, 2004), MDB. 1000 sample set of documents are collected from various biological domains and these documents are analyzed and given as the input to the second stage.

In the second stage the above documents are preprocessed for decreasing the workload in the further stage. The documents are preprocessed using specialized algorithms such as Stemming (Hull, 1996) Stop word removal algorithms, in order to save computing time.

The third phase focuses on clustering the documents in the previous phase. We present a novel clustering method using the approach of Similarity. In the Last Phase we propose a Dynamic Programming approach (Yamamoto *et al*., 2003) Knapsack Algorithm (Martello, 1990) and knapsack based proposed Algorithm for Information Retrieval.

TEXT GATHERING

The process of text gathering in Biomedical literature involves PubMed Open Access Initiative (PubMed Open Access Initiative, 2004), MedLine, National Library of Medicine, MeSH databases (Nelson, 2004), MDB etc. All the above databases contain more than 12,000,000 references of Biomedical publications. Google finds more PDF documents containing the word protein compared to PubMed (PubMed Open Access Initiative, 2004) and other Biomedical resources, but most of the results are irrelevant. Citation database CiteSeer showed good results only for computer science and information technology publications. Therefore we used PubMed (PubMed Open Access Initiative, 2004) and other Biological databases to obtain large clusters of full text documents, all containing relevant results. Our basic step is to download the documents and to prepare them for further stages. We have downloaded 1000,750 and 500 sample set of documents from three biological domains such as Tuberculosis, Cancer and AIDS. All the documents are stored in a separate centralized directory from where the documents can be processed and retrieved. The Biologists and the researchers who have administrative rights can upload their documents by using our interface. These documents can also be updated and processed for further retrieval. Most of the journals and research publications are encoded in PDF format (PDF Reference 15-v6.pdf, Adobe, 2004). It is necessary to convert these PDF documents (PDF Reference 15-v6.pdf, Adobe, 2004) into the textual documents.

There are conversion tools that convert PDF documents to HTML, doc, but these are not reliable and cost expensive. To successfully employ text mining on PDF (PDF Reference 15-v6.pdf, Adobe, 2004) encoded

paper, it would be advantageous to start with importing new archives, which are developed for PDF (PDF Reference 15-v6.pdf, Adobe, 2004) conversion. We have implemented PDF conversion successfully using java archives.

TEXT PREPROCESSING

Text Preprocessing transforms text into an information rich document matrix. This method indicates the frequency of every term in the document collection. 80-20% rule (i.e.) 80% of the work is done by the preprocessing stage and 20% of the work is done by the remaining stages. Text preprocessing is to prepare the data for the Clustering and Information Retrieval process. The basic steps are;

- Tokenization
- Data cleaning
- Stop word removal
- Stemming (Paice/Husk)
- Identification of most interesting terms

In Tokenization the sentences are tokenized by converting words, numbers and punctuation marks in raw text into separate tokens. Data cleaning process involves removal of the unwanted symbols such as punctuation, special characters and conversion of the upper case letters into lower case letters. We have developed our own algorithm for data cleaning process. Stop word list removal process involves removing stop words such as around, above, below, bottom, far, many, etc. We have collected all the possible stop words and stored in a separate file, each and every token is matched with the stop word in the stop word list file, if it is matched then the tokens are removed from the document. In Stemming there are different algorithms (Hull, 1996) used for the Identification of root words such as Lovin's, Porter's (Porter, 2001), Paice/Husk (Paice and Hooper, 2005), S-removal algorithms. We have analyzed the strength and the similarity of Affix removal stemming algorithms measuring 5 different Metrics as:

- The mean number of words per conflation class,
- The number of words and stems that differ,
- The mean number of characters removed in forming stems,
- The median modified Hamming distance between words and their stems,
- The mean modified Hamming distance between words and their stems.

According to the modified Hamming distance descriptive statistics (Table 1), Paice/Husk algorithm (Paice and Hooper, 2005) is producing more mean modified Hamming distance compared to other algorithms, so that we have selected the Paice/Husk algorithm (Paice and Hooper, 2005) for the effective stemming. This algorithm eliminates the extreme outliers and these outliers are clearly over stemmed (too many characters were removed) for example this reduces ultra nationalism to ultra.

Table 1: Mean modified hamming distance for various stammers

Stammer	Mean modified hamming distance
Lovins	1.72
Paice/Husk	1.98
Porter	1.16
S-Removal	0.03

Table 2: Paice/hushstemmer output

Original word	Stemmed word	Original word	Stemmed word
Respiration	Respir	Pathology	Paty
Tumor	Tum	Virus	Vir
AIDS	AID	Specimen	Specim
Cell	Cel	Tuberculosis	Tuberculos
Genes	Gene	Culture	Cult
Biological	Biology	Healing	Heal

This algorithm contains various stem rules, for example

dei3y> {-ied >-y}

The words with prefix ied should be replaced by y.

de2> {-ed >-}

The words with prefix 'ed' should be removed. In this method the main function is to identify the nouns, verbs and stems. The terms that are grammatically close to each other (cell, cells) are mapped to one term (cell). After applying the Paice/Husk stemming rules (Paice and Hooper, 2005) the original words and stemmed words are listed in Table 2.

CLUSTERING (SIMILARITY BASED APPROACH)

We propose a new approach (Similarity based approach for clustering). This approach is to classify or to group the objects based on attributes or features into K number of groups. K is a positive integer number. The grouping is done by maximizing the similarity between the document and the cluster centroid. Here the centroid is computed by the similarity measure not based on the distance metrics. We have taken 1000, 750 and 500 sample set of documents from various biological domains such as Tuberculosis, Cancer and AIDS. The output of this phase is a set of clusters which is given as a input to the fourth phase (Information retrieval process). Knapsack and Knapsack based proposed Algorithm are applied only for the clusters of the previous phase, so the number of documents to be searched is reduced.

Table 3: Proposed similarity based clustering algorithm results

No. of docs	No. of Iterations	Cluster ID	No. of docs for each cluster	No. of clusters
1000	8	I	487	3
		II	190	
		III	323	
750	13	I	381	3
		II	123	
		III	246	
500	9	I	244	3
		II	91	
		III	165	

Proposed Algorithm (similarity based Clustering approach):

- For each document, Term Vector is identified by Weighing function (from Eq. 3)
- Initialize a cluster and assign a document to the initial cluster and the documents term vector is assigned as the centroid for the initial cluster.
- Select another document and similarity is measured between the document and centroid of each cluster by the formula.

$$\text{Similarity} = \text{cosine}(Q.D_i) = \frac{Q.D_i}{|Q|X|D_i|} \quad (5)$$

If the similarity is 0 for all the clusters, new cluster will be formed and the document is assigned to the new cluster. Otherwise find the cluster with maximum similarity and the document will be assigned to the cluster (Table 3).

- For each cluster the centroid is computed by the average of the term vector of the documents in the cluster.
- Repeat the steps 3 and 4 for all documents.
- Refine the clusters.

Proposed Algorithm vs K-Mean Clustering:

- In our Proposed Algorithm, number of cluster K need not be determined before hand but in K-Means it should be determined before hand.
- In our Algorithm number of documents is not limited, but K-Means is applicable only for limited set of documents.
- In this Algorithm, outliers and extreme outliers can be detected but this not possible in K-Means.

INFORMATION RETRIEVAL PROCESS (KNAPSACK AND PROPOSED KNAPSACK)

Background of knapsack algorithm: Greedy method is applied to solve the knapsack problem (Martello *et al*., 1990) and is applied for document clustering. The n documents are treated as n objects and a knapsack or bag (solution space) object i has a weight w_i and the knapsack has a capacity m. The objective is to obtain a filling of the knapsack that maximizes the total profit earned (p_i, x_i). Since the knapsack capacity is m the total weight of all the chosen objects to be at most m. A solution to the knapsack problem can be obtained by making a sequence of decisions on the variables x_1, x_2, x_3….x_n. A decision on variable x_i involves determining which of the values 0 or 1 is to be assigned to it. The decisions on the x_i are made in the order x_n, x_{n-1}… x_1. Following a decision on x_n there are two possible states: The capacity remaining in the knapsack is m and no profit has accrued or the capacity remaining is m-w_n and a profit of p_n has accrued. The remaining decisions $x_{n-1,}$ ….., x_1 must be optimal with respect to the problem state resulting from the decision on x_n otherwise x_n…… x_1 will not be optimal. Hence the principle of optimality holds. Formally the problem can be stated as;

$$\text{Maximize} \sum_{1\le i\le n} p_i x_i \quad (6)$$

$$\text{Subject to} \sum_{1\le i\le n} wi\ xi \le m \quad (7)$$

$$\text{and } 0 \le x_i \le 1,\ 1 \le i \le n \quad (8)$$

The profits and weights are positive numbers. A feasible solution is any set (x_1, x_2, x_3….x_n.) satisfying Eq. 7 and 8. An optimal solution is a feasible solution for which Eq. 6 is maximized.

Knapsack based proposed algorithm: In this study we propose a new algorithm (Modified Knapsack with additional constraints). A set of 1000 sample documents are taken from Pubmed publications for this research (PubMed Open Access Initiative, 2004). These documents are preprocessed by the Paice/Husk Stemming Algorithm (Paice and Hooper, 2005). Preprocessing includes data cleaning (removal of stop words such as a, as, the and or, under…,). After the completion of preprocessing keywords and the frequency of occurrences are identified then this is given as input to the proposed algorithm. Initially the document is selected randomly then the fitness value of the document is identified. This fitness value is compared with the user defined fitness value and the document is accepted if and only if the fitness value is greater than or equal to the user defined fitness value, otherwise the document is rejected. This process is repeated until the number of documents is equal to the

total number of documents to be displayed. Finally the document sequence is generated. After the completion of each and every iteration the average fitness value is calculated for every sequence and this will be replaced by the old fitness value and will be considered for further evaluations. This facility increases the fitness value of the documents than Knapsack problem. This process is repeated until the two consecutive iterations have the same fitness value. Among the sample set of documents if all the documents fitness value are below the user defined fitness value then a specific Type of Coefficient (TC) is used. Initially this coefficient is set as 5% of the total No. of documents. In this situation the fitness value is decremented gradually and the TC value is either decremented or incremented by using the equations 9, 10 and 11.

$$\Delta f = max\_kd - init\_fit \quad (9)$$

$$\Delta tc = (reject/doc\_opened)^* (\Delta f/init\_fit) \quad (10)$$

$$tc = tc + tc^* \Delta tc \quad (11)$$

Pseudo code (proposed method):

```
Input: Set of clusters from the clustering phase.
Output: Sequence of retrieved relevant documents.
ndocs = Total No. of documents;
ndisp = No. of documents to be displayed;
init_fit. = Initial fitness value (Threshold);
tc = Tuning Coefficient;
Keyword = The word to be searched;
Identify the cluster for the key word;
Repeat
Set the initial value of tc as 5% of ndocs;
Repeat
Select a document randomly;
doc_opened++;
Extract the keywords from the documents
using Paice/Husk Stemming Algorithm;
For the above selected document, find the keyword
density kd;
If kd > init_fit then
Accept the Document and store the Document in the
Solution Space;
accept++;
Else
Reject the Doc; reject++;
If reject = tc then
init_fit = init-fit-1;
max_kd = maximum fitness value for the current
iteration;
Δ f = max_kd – init_fit;
Δ tc = (reject/doc_opened)* (Δ f/init_fit);
tc = tc+tc* Δ tc;
End if
End if
Until accept = ndisp;
init_fit. = Average fitness of the selected sequence;
Accept the documents whose
fitness > init-fit. for the next iteration;
accept = accept – No. of documents removed;
Until previous init_fit = current init-fit;
Return the sequence;
```

In Knapsack (Martello, 1990) the No. of iterations are more compared to our proposed algorithm based on Knapsack and will be generating more sequence of documents which are feasible. But in our proposed method the best optimal sequence is generated.

RESULTS AND DISCUSSION

In Table 4 and 5 the results of Information retrieval methodologies (Knapsack and Proposed method) are compared. Three Key words are considered such as Cell, AIDS and Tuberculosis from three biological domains Cancer, AIDS and Tuberculosis. 1000, 750 and 250 sample set of documents are considered as input. The number of documents to be displayed is 5% of the total number of documents. In Knapsack the number of documents to be searched is greater than the total number of documents (for example) for 1000 sample set of documents the number of documents to be searched is 1277 but in proposed 319. The number of documents to be searched in proposed Knapsack is very less compared to Knapsack and also the maximum number of iterations taken are only 4 but in knapsack it is 10. The time complexity and fitness is not satisfactory for knapsack algorithm compared to proposed knapsack. So Our Proposed Algorithm outperforms the existing Knapsack algorithm by three parameters such as number of documents to be searched, Fitness and Number of iterations.

The Fig. 1 plotted between the Fitness values of the key words Cell and Tuberculosis from Cancer and Tuberculosis biological domains and the number of documents. The proposed algorithm performs better fitness compared to knapsack.

In Fig. 2 the number of documents to be searched in Knapsack is greater compared to the total number of documents, but in proposed the number of documents to be searched is less.

Table 4: Results of similarity based clustering with knapsack information retrieval process with the threshold value of 50%

Key words	No. of docs	Doc Freq (DF)	No. of docs displayed	Cluster id	No. of docs searched	No. of iterations	Fitness (%)
Cell	1000	240	50	I	1277	10	90
	750	181	38	I	980	10	93
	500	120	25	I	628	10	91
AIDS	1000	84	50	II	1617	10	85
	750	63	38	II	1114	10	86
	500	42	25	II	758	10	99
Tuberculosis	1000	198	50	III	1408	10	80
	750	145	38	III	952	10	84
	500	99	25	III	725	10	84

Similarity based clustering with knapsack information retrieval process

Table 5: Results of similarity based clustering with proposed knapsack information retrieval process with the threshold value of 50%

Key words	No. of docs	Doc Freq (DF)	No. of docs displayed	Cluster id	No. of docs searched	No. of iterations	Fitness (%)
Cell	1000	240	50	I	319	4	100
	750	181	38	I	174	4	100
	500	120	25	I	152	4	100
AIDS	1000	84	50	II	220	3	83
	750	63	38	II	117	2	84
	500	42	25	II	78	2	99
Tuberculosis	1000	198	50	III	306	3	90
	750	145	38	III	226	3	92
	500	99	25	III	113	3	100

Similarity based clustering with proposed knapsack information retrieval process

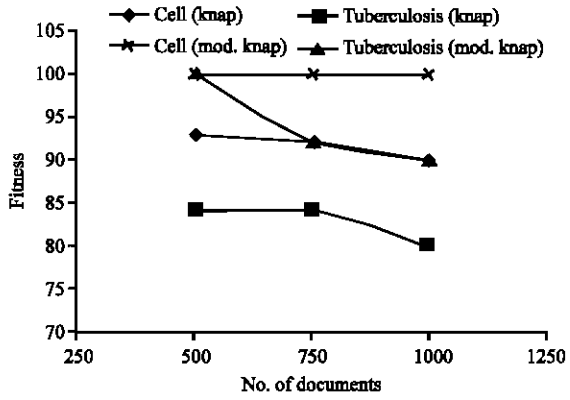

Fig. 1: Knapsack vs proposed knapsack (Fitness)

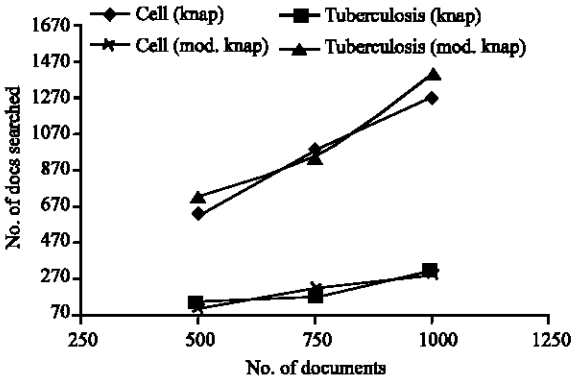

Fig. 2: Knapsack vs proposed knapsack (documents searched)

CONCLUSIONS

A Text Mining Frame work is constructed by the four stages such as text gathering, text preprocessing, clustering and Information Retrieval Process using Dynamic Programming Techniques (Yamamoto *et al*., 2003). In the third stage we proposed Similarity based Clustering approach which is producing better results compared to existing algorithms like K-Means. In the Fourth phase we have proposed an algorithm based on knapsack which outperforms the existing Knapsack algorithm (Martello, 1990). We have implemented all the methods using C Language. The results of knapsack and proposed methods are compared using fitness value of the documents, number of documents searched, number of iterations and the graphs are plotted for various parameters.

REFERENCES

Hull, D., 1996. Stemming algorithms-A case study for detailed evaluation. J. Am. Soc. Inform. Sci., 47: 70-84.

Martello, S., 1990. Knapsack problems: Algorithms and computer implementations, Paolo Toth University of Bologna, Bologna, Italy, Wiley-Interscience Series In Discrete Mathematics and Optimization.

Nelson, S., 2004. Medical Subject Headings Fact Sheet. http://www.nlm.nih.gov/pubs/factsheets/mesh.html U.S. National Library of Medicine.

Paice, C., and R. Hooper, 2005. The Lancaster Stemming Algorithm. Available at: http://www.comp.lancs.ac.uk/computing/research/stemming/.

PDF specification, 2004. http://partners.adobe.com/asn/acrobat/sdk/public/docs/PDFReference15-v6.pdf, Adobe.

Porter, M.F. An algorithm for suffix stripping (reprint), in Readings in Information Retrieval, Morgan Kaufman, http://www.tattarus.org/~martin/PorterStemmer.

Porter, M.F., 2001. Snowball: A Language for Stemming Algorithms. Available at: http://www.snowball.tartarus.org/texts/introduction.html.

Possas, B., N. Ziviani, W. Meira and B. Ribeiro-Neto, 2005. Set-based vector model: An efficient approach for correlation-based ranking.

Pub Med Central Open AccessInitiative, 2004. http://www.pubmedcentral.nih.gov/about/openftplist.html.

Pub Med, http://www.ncbi.nlm.nih.gov/PubMed/, 2004.

Salton, G. and C. Buckley, 1998. Term-Weighing Approaches in Automatic Text retrieval. Inform. Process. Manage., 24: 513-523.

Yamamoto, E., M. Kishida, Y. Takenami, Y. Takeda and K. Umemura, 2003. Dynamic programming matching for large scale information retrieval.

Yoshioka, M. and M. Haraguchi, 2005. On a combination of probabilistic and boolean ir models for WWW document retrieval. Proceedings of NTCIR-4, Tokyo.