Medical Document Classification

[1]Shrutha Kashyap,[2]Sushma M G,[3]Varsha Rajaram,[4]Vibha S
Department of Computer Science,
K. S. Institute of Technology, Bangalore

***Abstract* - One of the most challenging projects in information systems is extracting information from unstructured texts, including medical document classification. We have developed an application that classifies a medical document by analyzing its content and categorizing it under predefined topics. We have fixed the scope to four predefined topics namely, Respiratory, Digestive, Cardiology, Neurology. We use term frequency technology to analyze the medical text documents and classify them based on weightage calculation.**

Keywords - Text mining, Medical documents, term frequency, weightage.

I. INTRODUCTION

Text mining, also referred to as text data mining, roughly equivalent to text analytics, refers to the process of deriving high-quality information from text. High-quality information is typically derived through the devising of patterns and trends through means such as statistical pattern learning. Text mining usually involves the process of structuring the input text (usually parsing, along with the addition of some derived linguistic features and the removal of others, and subsequent insertion into a database), deriving patterns within the structured data, and finally evaluation and interpretation of the output. 'High quality' in text mining usually refers to some combination of relevance, novelty, and interestingness. Typical text mining tasks include text categorization, text clustering, concept/entity extraction, production of granular taxonomies, sentiment analysis, document summarization, and entity relation modelling (i.e., learning relations between named entities). The overarching goal is, essentially, to turn text into data for analysis, via application of natural language processing (NLP) and analytical methods.

It has a broad range of applications including marketing applications, academic application, sentiment analysis, and most commonly, in the field of medicine. One such application in medicine is the classification of medical documents. Consider there is a huge set of medical documents where in humans have to read the documents and classify them as belonging to some category in the medical field. For example, Everyday there are thousands of people visiting the hospital. Each one of them will have different cases, that are to be dealt with different departments and for each case, the hospital management system needs to maintain different records. This is a tedious job when done manually. In order to overcome this issue, it's necessary for the hospitals to maintain documents in a structured way so that it is useful for doctors/researchers to pick the particular category of document for their requirement. This project proposes the classification of huge medical documents automatically.

The scope for the application is as follows:

- We are considering only text documents as input files.
- We are restricting to four categories, namely, Respiratory, Digestive,
  cardiology and neurology.

With these constraints at hand, we go about developing a system that takes medical documents as input. The contents in the documents are parsed. An unnecessary list of keywords is maintained in the database. The words from the documents are compared against this list which results in a set of necessary keywords required for the classification.

Further, we maintain another list of keywords for each predefined topic which is used as the training set.

We compute the term frequency for each word in the necessary list based on which the weights are assigned to each classification. Finally, the document is classified under the topic with the highest weightage.

II. RELATED WORK

Abdullah Muhammad Alghoson from Claremont Graduate University has proposed a method to classify the medical documents [1]. The classification was done using predefined terms from Medical Subject Headings (MeSH). It used a corpus of 50 full-text journal articles (N=50) from MEDLINE, which were already indexed by experts based on MeSH. Using natural language processing (NLP), the algorithm classifies the collected articles under MeSH subject headings. The algorithm's outcome was evaluated by measuring

its precision and recall of resulting subject headings from the algorithm, comparing results to the actual documents' subject headings. The algorithm classified the articles correctly under 45% to 60% of the actual subject headings and got 40% to 53% of the total subject headings correct. This holds promising solutions for the global health arena to index and classify medical documents expeditiously.

Bodenreider[2], uses The Unified Medical Language System (UMLS) which contains semantic information about terms from various sources; each concept can be understood and located by its relationships to other concepts. According to Bodenreider, the UMLS concepts are used to classify condition terms in the Clinical Trials database into broad disease categories in the MeSH database in three major steps: 1) matching condition terms to the UMLS concepts using the exact match technique or normalization techniques such as inflection, punctuation, and case sensitivity.

2) Limiting the UMLS concepts to MeSH subject headings by passing through four consequent steps until the matching process succeeds. The four steps are: using MeSH term synonyms, choosing an associated expression as a translation, selecting a MeSH term using the MeSH hierarchy of concepts, and selecting the non-hierarchical related concept.

3) Assigning MeSH subject headings to the major categories of MeSH subject headings that represent the major disease categories using MeSH trees of hierarchy. The author uses "condition term" as an evaluation unit where the algorithm was applied to classify 12,612 condition terms in the Clinical Trials database, in which 1,823 terms were distinct. The evaluation of the algorithm outcomes was manually reviewed, and the evaluation metrics precision and recall were used. The results were astonishing: 96% of the 1,823 condition terms were successfully classified in MeSH.

III. DESIGN

A high level design flow diagram is used to convey the basic working of the application. Data flow diagrams can be divided into different levels in order to show the working of the system at a low level, such as the inputs and the output, as well as at a high level, such as the underlying functions that drive the system. In this project we have shown three levels of data flow diagrams.

A. Context Analysis Diagram

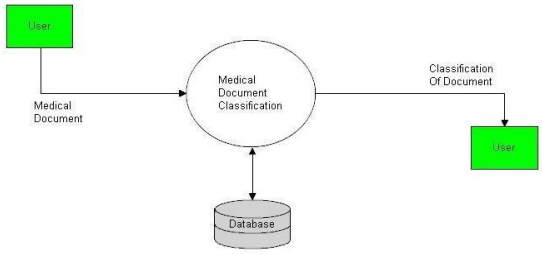

Fig.1. Context analysis diagram

The above figure shows the basic working of the application. The user uploads a medical document to the application from his local file system. The application parses and compares the document against the list present in the database. The output shows the classification of the document.

B. Document Classification Process

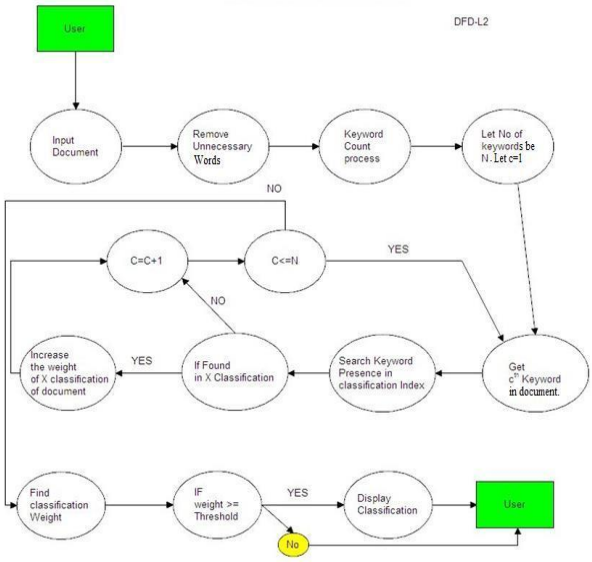

Fig.2. Document Classification Process

The above figure describes the detailed procedure of classification of a document. As mentioned earlier, initially the user gives the document that he desires to classify as the input. The document is parsed and the unnecessary words like for, it, the, coming etc. are removed and the remaining keywords present in the document are extracted. N is a variable initialized to the number of keywords found in the document. Now the system has to match the keywords found with the predefined

keywords. To do the same, an index C is initialized to 1. The first keyword found is taken and a query is sent to the database in order to check for its presence in any of the classifications. If found in classification X, the corresponding weight of that classification is incremented. The index C is advanced to the next keyword and the procedure continues until all the keywords are checked. Now, the classification with highest weight is considered and checked against a threshold. If its greater than the threshold, the document is classified under that classification. Else, the user is notified.

IV. IMPLEMENTATION

In order to achieve the task classifying text documents, some common text mining techniques such as string tokenizing and term frequency calculation has been used. To begin with, the application is provided with an option to generate the training data, which does the task of loading the training data set, ie, the list words with predefined labels, from the database to the application . This is followed by process of classifying the uploaded document, which forms the crux of this application. The major tasks involved in the classification process are 1) Tokenize 2) Removing of the unnecessary keywords or stop words that includes conjunctions, prepositions, pronouns, articles, etc. 3) Calculate the term frequency of each word and it's weightage. The pseudocode for the removal of unnecessary keywords, and weightage calculation is given as follows:

A. Pseudocode For Removal Of Unnecessary Words

1. Read the file.
2. Using space as the delimiter split the file into 'n' number of tokens.
3. Initialize file 's' to empty.
4. Initialize i=1.
5. Search the presence of ith word in unnecessary list.
6. If not present, append the word into the file 's'.
7. Increment i.
8. Stop.

The first step towards removing unnecessary words is to split each token in the text corpus using space as the delimiter. This divides the document into a number of tokens. Let this number be n. Next we take any empty file, which is indexed by a pointer, say s. This files is meant for containing the final keywords that are obtained after removing unnecessary words.

Initialize a variable i to 1, which is used to index each word in the document. Then search the presence of the ith word in the list of unnecessary words present in the database. If this word is not present in the database, then append it to the file s. Else, increment i by 1, which advances the pointer to the next keyword and repeat the steps 5 though 7, until all the keywords are exhausted.

B. Pseudocode For Weightage Calculation

1. Select the file to find classification
2. File is transferred to web server
3. Unnecessary words has to be removed
4. Let 'n' be the words remaining after removed of unnecessary words
5. Initialize array words[n],count[n]=0
6. Let i=1 and index c=1
7. Fetch ith word from the remaining words
8. Check ith term in word[]. If not found, word[c]=ith word and c++ else point k to ith word and increment count of k.
9. Increment i. If i<=n go to step7
10. Stop.

Weightage calculation refers to evaluation of how much weightage each word has for a particular classification. The file uploaded by the user is transferred to the web server where the weightage calculation algorithm runs. Unnecessary words are removed using the algorithm previously described. The remaining n words are stored in another file, s.

At this stage, we create two arrays: word[n], which is used to index each word in a document; and count[n], which is used to increment the count associated with that word. This count gives the term frequency of each word, ie, the number of times they have been repeated in the document. Both the arrays are initialized to 0. There are two variables i and c. The variable i is used to index each word in the file s containing the necessary words, and c is used to index each word in the word array. Both these variables are initialized to 1.

Fetch the ith word from file s and check its presence in the word array. If it is not found, then add this word to the word array, and increment the index c to point to the next word. If it is found, however, initialize an index k to point to the count associated with the ith word in the file, and increment its count value.
Now, increment i to point to the next word in the file, and repeat the steps 7 through 9 until all the words are exhausted.

C. Pseudocode For Classification

1. Let "n" be the number of classifications available
2. Let class[n] array be initialized to 0.
3. Read the input file.
4. Remove unnecessary words.
5. Now, get unique keywords and get term frequency.
6. Let m be the number of unique keywords.
7. Let c=1.
8. Get cth keyword.
9. Find the classification of cth keyword and let the classification number be k. Increase the weight in array class[k] by 1.
10. c=c+1
11. If c<=n, go to step 8.
12. Find the highest weight in class array & index be j
13. Given document belongs to jth class
14. Print

After removal of unnecessary words, we read the file 's' containing the necessary words. Then we compute the term frequency of each word in the file i.e, the number of times each word is repeated in the file. We consider c which indexes the words in file 's' and initialize it to the first word. We compare it with the training data set to check for its presence. If found, count[c] value is incremented and finally assigned to the weight of the classification. Else we check for the next word. We compare the weights of all the classifications and check if it crosses the threshold. If so, the file is classified under the classification which has the highest weighatge.

D. Computation of weightage for Classification

Weightage of each classification refers to the final weightage obtained on adding the term frequencies of each word falling under that classification. If Wi is the classification for category Xi, then,

$$Wi = Tf \text{ of each word belonging to } Xi \quad (1)$$

This sum is later multiplied by a constant integer value C. This is done in order to uniquely identify the document as belonging to a category. Let the product be P.

$$P = Wi * C \quad (2)$$

The product P is then compared with a threshold value T. If P is greater than T, Then this weight is the final weightage W. Else, the document cannot be classified.

V. CONCLUSION

With the growing increase of medical articles every day, it is becoming very important to develop tools in order to process and classify them automatically. In this paper, we have proposed an application that classifies medical documents based on analyzing its content and categorizing it under predefined topics. For future work, we can expand the scope to cater to many more categories. Successful implementations of the algorithm could solve many real world problems including medical document indexing, medical email routing, and search directory implementation for medical terms. As a result, that will help indexers to classify documents expeditiously, thus improving the global health as a whole.

ACKNOWLEDGMENT

We would like to extend our gratitude to the Principal, Dr. T. V. Govindaraju, K. S. Institute of Technology, Bangalore, for facilitating us to present the paper .We thank Mrs Archana J K, Asst. Professor for her encouragement as the guide for our project paper. We would also like to thank, Dr. Naveen N C, Professor, and