Pattern Matching Techniques using Data Mining Methodologies

K Neeharika[1], G Durvasi[2],L Priya Darsini[3]
[1,2,3]*Assistant professor, Department of IT*
Andhra Loyola Institute of Engineering and Technology, Vijayawada-08.

Abstract **-** Data mining is booming and most developing research area in intellect systems and their workgroup application. Data Mining is a logical area with the theme to explore data where data are received from large amount of data set and also convert into understandable format. Pattern matching is a process used to identify the number of occurrence of particular pattern in a given input data set. Pattern matching is one of the methods for classification of data, it is used to classify data into predefined groups or classes. In this paper, we proposed utilities made available in Linux to make use in pattern matching. With this approach, the grep family utilities are proposed to apply on data warehouse, and to warehouse the result into a temporary file. For experimentation, this work has used two types of documents, i.e. .txt and .docx. Performance measures used are search time, number of iterations and accuracy.

Keywords **-** Pattern matching, Complexity, Efficiency, Techniques, data mining, pattern, utilities, warehouse, grep family.

I. INTRODUCTION

The data mining involves variety of techniques to deduce a valid and useful hidden information by means of understandable correlations and patterns from large amount of data called data ware house. Finding of needful patterns from data or warehouse has different conventions like data pattern processing, knowledge extraction, information extraction, knowledge discovery and information harvesting. Data mining is a well familiar among community of database researchers top level business and statistics personnel. Preparing data ready for mining involve many preprocessing steps referred to Knowledge Discovery in Databases (KDD).

Information Retrieval system is used to identify the documents in a document database which match a user's query. In information retrieval system the text can be divided into two important units, they are the document such as journal paper, book, chapters, sections, web pages, paragraphs, source code of computer program, etc., and the term such as word, pair of words, and phrase within a particular document.

Desktop search is nothing but it performs the searches over the content of the file or document. Pattern matching algorithms are also known as string matching algorithms and these are essential class of string algorithms which supports to discover one or all existences of the string within an enormous group of text. It is an important concept of numerous problems and it is used in various applications such as text mining, data retrieval, DNA pattern matching and finding certain important keywords in security applications. String matching algorithms has two techniques, they are, exact matching and approximate matching. In exact string matching, the pattern is entirely matched with the particular text window of input text and it displays the starting or initial index position. The algorithms which belong to this type are Knuth-Morris-Pratt (KMP), Needleman Wunsch (NW), Dynamic Programming, Boyer Moore and Smith Waterman (SW). In approximate string matching, if certain part of the pattern matched with the selected text window then immediately it displays the output.

Natural Language Processing (NLP) is both a modern computational technology and a method of investigating and evaluating claims about human language itself. Text mining attempts to discover new, previously unknown in-formation by applying techniques from natural language processing and data mining. Clustering, one of the traditional text data mining techniques, is unsupervised learning paradigm where clustering methods try to identify inherent groupings of the text documents so that a set of clusters are produced in which clusters exhibit high intra-cluster similarity and low inter-cluster similarity [1].

Usually, in text mining techniques, the frequency of a term (word or phrase) is computed to explore the importance of the term in the document. However, two terms can have the same frequency in a document, but one term might be contributing more to the meaning of its sentence than the other term. It is important to note that extracting the relations between verbs and their arguments in the same sentence has the potential for analyzing terms within a sentence. The information about who is doing what to whom clarifies the contribution of each term in a sentence to the meaning of the main topic of that sentence. In this paper, a novel concept-based mining model is pro-posed. It captures the semantic structure of each term within a sentence and a document, rather than the frequency of the term within a document only. Each sentence is labeled by a semantic role labeler that determines the terms which con-tribute to the sentence semantics associated with their se-mantic roles in a sentence.

Each term that has a semantic role in the sentence, is called a concept. Concepts can be either words or phrases and are totally dependent on the se-mantic structure of the sentence. When a new document is introduced to the system, the

proposed mining model can detect a concept match from this document to all the previously processed documents in the data set by scanning the new document and extracting the matching concepts.

Need of Pattern Matching: Pattern matching is the process of checking a perceived sequence of string for the presence of the constituents of some pattern. In contrast to pattern recognition, the match usually has to be exact. The patterns generally have the form sequences of pattern matching include outputting the locations of a pattern within a string sequence, to output some component of the matched pattern, and to substitute the matching pattern with some other string sequence (i.e., search and replace). Pattern matching concept is used in many applications Following figure shows the different applications.

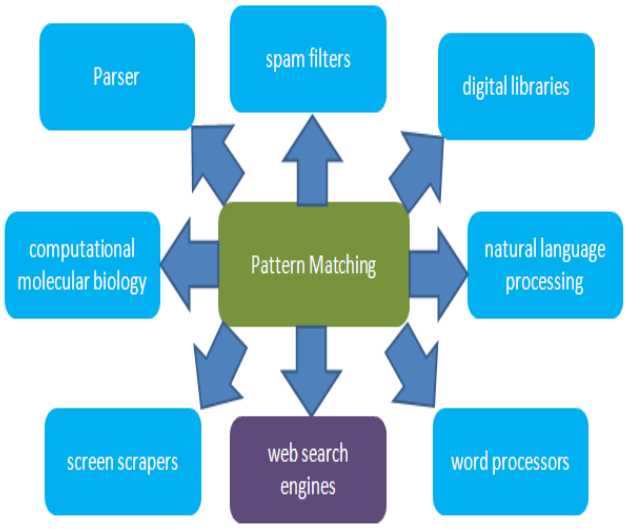

Figure 1: Applications of Pattern Matching

II. LITERATURE SURVEY

The use of web application is increase day by day. The web new web applications are in use for searching data on the internet. String algorithms play an important role for this. Different people are working on software and hardware levels to make pattern searching faster. By applying various algorithms in various applications the approximate best algorithm for different applications is determined [1]. The recommended algorithms give the reduced complexity and also reduced computation time. The algorithms assigned to various applications may not be best optimal algorithm but better than the general algorithms. Rather than applying each algorithm to every application one application is explained with particular optimal algorithm [2]. To support a different type of data, different algorithms are used.

Our exploratory analysis on Medicare fraud detection involves building and assessing three learners on each dataset. Based on the Area under the Receiver Operating Characteristic (ROC) Curve performance metric, our results show that the combined dataset with the Logistic Regression (LR) learner yielded the best overall score at 0.816, closely followed by the Part B dataset with LR at 0.805. Overall, the Combined and Part B datasets produced the best fraud detection performance with no statistical difference between these datasets, over all the learners. Therefore, based on our results and the assumption that there is no way to know within which part of Medicare a physician will commit fraud, we suggest using the Combined dataset for detecting fraudulent behavior when a physician has submitted payments through any or all Medicare parts evaluated in our study.

III. DATASETS

In this section, we describe the CMS datasets we use (Part B, Part D and, DMEPOS). Furthermore, the data processing methodology used to create each dataset, including processing, fraud label mapping between the Medicare datasets and the LEIE, and one hot encoding for categorical variables is discussed. The information within each dataset is based on CMS's administrative claims data for Medicare beneficiaries enrolled in the Fee-For-Service program. Note, this data does not take into account any claims submitted through the Medicare Advantage program. Since CMS records all claims information after payments are made, we assume the Medicare data is already cleansed and is correct. Note that NPI is not used in the data mining step, but rather for aggregation and identification. Additionally, for each dataset, we added a year variable which is also used for aggregation and identification.

Inappropriate payments by insurance organizations or third party payers occur because of errors, abuse and fraud. The scale of this problem is large enough to make it a priority issue for health systems. Traditional methods of detecting health care fraud and abuse are time-consuming and inefficient. Combining automated methods and statistical knowledge lead to the emergence of a new interdisciplinary branch of science that is named Knowledge Discovery from Databases (KDD). Data mining is a core of the KDD process. Data mining can help third-party payers such as health insurance organizations to extract useful information from thousands of claims and identify a smaller subset of the claims or claimants for further assessment.

We reviewed studies that performed data mining techniques for detecting health care fraud and abuse, using supervised and unsupervised data mining approaches. Most available studies have focused on algorithmic data mining without an emphasis on or application to fraud detection efforts in the context of health service provision or health insurance policy. More studies are needed to connect sound and evidence-based diagnosis and treatment approaches toward fraudulent or abusive behaviors. Ultimately, based on available studies, we recommend seven general steps to data mining of health care claims.

IV. PROPOSED METHOD

This algorithm is a proficient string matching algorithm, and it has been the standard point of reference for the string matching problems. This algorithm checks the characters of the pattern from right to left order6. On these terms of

mismatch or a complete match of full pattern, it uses the two functions to shift the window from left to the right and the two functions are good suffix shift and bad character shift. The searching phase of the algorithm in o (nm) time complexity and the best performance is O(n/m).

First calculate the state transition table S from the pattern P, the pattern may be single line, multiple lines or a file. Then set the pointer value and state values. If the pointer value is smaller than the pattern and text value then read the character from right to left, beginning the rightmost one. In this case if match occurs, the return the index of the character, otherwise shift the pointer value again the same process will be done by each level until the pattern found or not found.

```
1. Input: m- Length of Pattern; n- Length of Text
2. Output: text index
3. Calculate state transition table S from the pattern P
4. Set pointer = 0, state = 1 and ppointer = m - 1
5. While pointer < n - m do
      a. Read character T(align + ppointer) into c
      b. (shift, state, ppointer, match, state) = S[state, c]
      c. If match = true then
            i. Return pointer
      d. End if
      e. If state = true AND remaining characters match then
            i. Return pointer
      f. End if
      g. pointer = pointer + shift
6. end while
7. return NIL
```

In the above code, a set of needful functions have been used to prepare the command "state". The list of all files mentioned in this section completes the command with mentioned four options which are discussed in brief in the previous section. The following is the another important script used to perform the task of the command.

For each sentence (in the for loop at line 3) the concepts of the verb argument structures which represent the semantic structures of the sentence are processed sequentially. Each concept in the current document is matched with the other concepts in the previously processed documents. To match the concepts in previous documents is accomplished by keeping a concept list pointer that holds the entry for each of the previous documents that shares a concept with the current document.

V. RESULTS

The experimental setup consisted of three datasets. The first data set consisted of 23,115 ACM abstract articles collected from the ACM digital library. The ACM articles are classified according to the ACM computing classification system into five main categories: general literature, hard-ware, computer systems organization, software, and data. The second data set has 12,902 documents from the Reuters 21578 dataset. There are 9,603 documents in the training set, 3,299 documents in the test set, and 8,676 documents are unused. Out of the 5 category sets, the topic category set contains 135 categories, but only 95 categories have at least one document in the training set.

The parameters are search time, number of iterations and relevancy for various types of inputs. The inputs are single word, multiple words and a file. For this analysis, the existing and enhanced string matching algorithms were implemented by using Java.

Search Time: It refers the time taken for searching the pattern within the input text. It can be estimated by comparison of each character in pattern with the input text.

Iterations: It refers the total number of iterations for matching the pattern with the input text. It is based on the given input document and various algorithms.

Relevancy: It refers the accuracy of the algorithm;

Table 1. Performance analysis of proposed Algorithm for text files

Input	Brute Force Algorithm		
	Time (ms)	Number of Iterations	Relevancy (%)
Single Word	0	2	100
Multiple Words	16	18	100
File	47	42	90

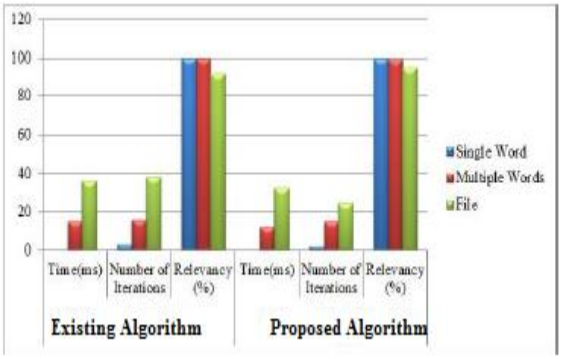

Figure 2:.Performance accuracy of proposed Algorithm.

The percentage of improvement ranges from +27.94% to +98.74% increase in the F-measure quality, and -23.03% to -95.68% drop in Entropy (lower is better for Entropy).

VI. CONCLUSION

This work provided the bridge between the Natural Language Processing (NLP) and the text mining disciplines through the concept-based mining model. There are two components used in this proposed model such as concept-based term analysis and concept-based similarity analysis. The clustering result is improved with the help of semantic structure analysis of the sentences in documents. The concept-based term analysis performed the analysis of semantic structure in both sentence and document levels. Based on the semantic result of the sentence, the concept-based similarity measure predicted the importance of the each concept in the dataset. The robust and an accurate

similarity calculations are achieved based on the concept matching and the similarity estimations. The comparison between the traditional algorithms with the proposed algorithm confirmed the effectiveness of the proposed clustering in large –size data handling.

VII. REFERENCES

[1]. Srikanthan, Sharanyan ,"Implementing the dynamic time warping algorithm in multithreaded environments for real time and unsupervised pattern discovery"., Computer and Communication Technology (ICCCT), 2011 2nd International Conference on 394 – 398, 15-17 Sept. 2011,

[2]. tan Salvador & Philip Chan,"Fast DTW: Toward Accurate Dynamic Time Warping in Linear Time and Space".KDD Workshop on Mining Temporal and Sequential Data, pp. 70-80, 2004.

[3]. Chu, S., E. Keogh, D. Hart & Michael Pazzani. Iterative, Deepening Dynamic Time Warping for Time Series. In Proc. of the Second SIAM Intl. Conf. on Data Mining. Arlington, Virginia, 2002.

[4]. Arcangel, Cory. "On Compression". Retrieved 6 March 2013.

[5]. Boyer RS, Moore JS. A fast string searching algorithm. Communication of the ACM. 1977; 20(10):762–72.

[6]. Pandiselvam P, Marimuthu T, Lawrance R. A Comparative Study on String Matching Algorithms of Biological Sequences, Springer Berlin Heidelberg. 2009; 510–17.

[7]. Charras C, Lecroq T, Daniel J. A Very fast string searching algorithm for small alphabets and long patterns, Combinational Pattern Matching, 9th Annual Symposium, CPM 98 Piscataway, New Jersey, USA. 2005; 1448:54–8.

[8]. Robert S, Boyer B, Moore JS. A fast string Searching Algorithm. Communication of the ACM. 1997; 20(10):762–72.

[9]. Hossein G, Shokoufeh S, Abozar S. A Survey of Pattern Matching Algorithm in Intrusion Detection System Tehran, Iran. Indian Journal of Science and Technology. 2016 Jun; 9(21):1–7.

[10]. Rahul M, Diwate B, Satish J, Alaspurkar A. Study of Different Algorithms for Pattern Matching. International Journal of Advanced Research in Computer Science and Software Engineering. 2013; 3(3):1–8.

[11]. Bhandari J, Kumar A. String Matching Rules Used By Variants of Boyer-Moore Algorithm. Journal of Global Research in Computer Science. 2014; 5(1).

[12]. Shivaji SK, Prabhudeva S. Plagiarism Detection by using Karp-Rabin and String Matching Algorithm Together. International Journal of Computer Applications. 2015; 116(23):1–5.