Mining the Text using Association Rule Mining Technique

Myint Myint Lwin, Myint Thuzar Tun
University of Computer Studies, Maubin
lwin.myintmyint@gmail.com, myintun@myanmar.com.mm

Abstract

As the amount of text available in electronic form continues to increase at alarming rate, the tools to manage these textual resources effectively will become critical. Information Retrieval System tries to save the users access time by classifying the documents and clustering the documents because users spend a lot of time to find documents or information from texts. Therefore, text mining is the most popular and it is necessary to solve this problem. The largest amount of work in text mining has been in the areas of categorization, classification and clustering of documents. Text mining has many methods to find the useful information. Among these methods, association rule mining is very suitable for finding the most frequent words that occur in the document collection. Association rule analysis is the task of discovering association rules that occur frequently in a given text sets. Our proposed system had been developed by applying the preprocessing steps of text mining system and Apriori algorithm for finding the pairs of most frequent words. These frequent words are associated with each other and they can provide the trained texts for the document classification.

1. Introduction

With the growing importance of electronic content and electronic media for storing and exchanging text documents, there is also a growing interest in tools that can help finding and sorting information included in the text documents [17].

Nowadays, the access to a large amount of textual documents becomes more and more effective due to the growth of the Web, digital libraries, technical documentation, medical data, etc [9].

The textual data in these textual documents constitute resources that it is worth exploiting. The analyzing and extracting useful information from documents written in natural language is very hard. Users need tools to compare different documents, rank the importance and relevance of the documents, or find patterns and trends across multiple documents. Manual analysis and effective extraction of useful information are not possible. Thus, text mining has become an increasingly popular and essential theme in data mining. Text mining is similar to data mining: while data mining seeks to discover meaningful patterns implicitly present in data, text mining aims to extract useful information and discovering semantic information hidden in texts. It detects interesting patterns such as clusters, association, deviations, similarities and differences in sets of texts. Among them, association rules are popular representations in data mining but have also been used in text mining.

Association rule mining is the most useful to find most frequent texts in the document corpus and *Apriori* algorithm is the standard algorithm of association rule mining. Therefore, our proposed system use *Apriori* algorithm for mining the frequent words in the text database. In a document database, each document can be viewed as a transaction, while a set of keywords in the document can be considered as a set of items in the transaction. Then, it produces the most frequent texts in the document corpus and they can be used by the document clustering system and the document classification.

Our paper is structured as follows. We start with presenting related work in section 2. In section 3, we present our system architecture with some examples and we discuss the mining steps of text in the text database in section 4. Finally, section 5 concludes this paper.

2. Related Work

The works related to our proposed system are presented here. Our system is based both text mining field and data mining field. Therefore, we present the details of these techniques. The first technique is text mining. A general overview on text mining can be found in [16]. Text mining is

understood as a process of automatically extracting meaningful, useful, previously unknown and ultimately comprehensible information from textual document repositories [12]. Ah-Hwee Tan, Kent Ridge Digital Labs, Heng Mui Keng Terrance [2] present the general framework of text mining. Moreover, Raymond J. Mooney and Un Yong Nahm are described another framework of text mining in [19]. Dunja Mladenic and Marko Grobelnik [6] present the text and web mining method. Brigitte Mathiak and Silke Eckstein [4] present the text mining processes. It can be viewed as an extension of data mining or knowledge discovery from (structured) databases. Haralampos Karanikas and Babis Theodoulidis [8] describe the main text mining operations. The key goal in text mining is to assist in this process by automatically discovering a small set of interesting hypotheses from a suitable text collection.

The second technique is data mining technique. Data Mining (sometimes called data or knowledge discovery) is the process of analyzing data from different perspectives and summarizing it into useful information. The aim of data mining is to find novel, interesting and useful patterns from data using algorithms (methods of finding such information) that will do it in a way that is more computationally efficient than previous methods. In [11] describes the Association Rule Mining, Market Basket Analysis, Boolean association rule, Apriori algorithm and how to mine the data from the database. Mining for association rules in text was first considered in [21] and [22]. In paper [13] presents how to mine the association rules in temporal document databases and strategies for association rules in temporal document databases. Rajman and Besancon [18] consider different methods for associating terms with the text, and the introduction of natural language processing techniques to association rule mining in text. Feldman et al. [23] show how to associate terms with the text based on extracting key terms and phrases from the text. Ahonen et al. [7] describe a general framework for text mining of frequent episodes from the text.

3. System Architecture

In this section, we present the detail of our system with some motivating examples. This system intends to extract the most frequent words from the text database. The system can only accepts text documents or text files which contain the words. But the distinct words in these files are no more than 200 words. Users must choose and give the text files as input to the system. Then, users can mine or find the pair of most frequent words that are associated with each other by selecting the minimum support. Our proposed architecture is shown in Figure-1.

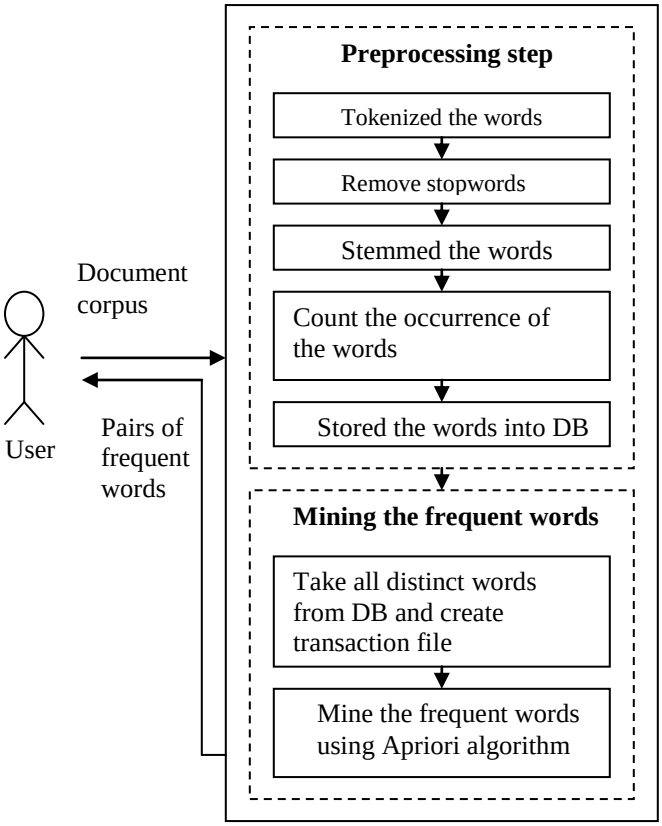

Figure -1: Architecture of proposed system

Our proposed system includes two parts. The first part is pre-processing. In this part, the system accepts the documents from the user and then processes the following steps.

1) *Tokenize the documents into words*: In this step the terms are extracted from the text documents. These terms are essential for mining. This step is a mapping from documents to a list of terms.
2) *Remove the stopwords*: Removing terms known to not be interesting or too frequently occurring, i.e., a, an, and, the, therefore. These terms are in general kept in a stopword list which contains terms that are considered stop words, but it might be that domain-related stop words are added as well.

3) *Stemming*: In this step, we used Porter Stemmer. It is a very widely used and available Stemmer, and is used in many applications. The Stemmer is based on the idea that the suffixes in the English language (approximately 1200) are mostly made up of a combination of smaller and simpler suffixes. This Stemmer is a linear step Stemmer. Specifically it has five steps applying rules within each step. Within each step, if a suffix rule matched to a word, then the conditions attached to that rule are tested on what would be the resulting stem, if that suffix was removed, in the way defined by the rule. Which determines a stem form of a given inflected (play, played, playing) word form generally a written word form. This means that a number of related words all will be transformed into a common term.
4) *Counting the occurrence of each word*: Which counts the occurrence of each word in the documents to store the text database.
5) *Checking and storing*: Which determines the count the words are greater than the threshold and store the words into the database that are satisfied the threshold value.

The second part is mining the frequent words from the text database. In this part, we create a transaction file that contains binary values such as 0 or 1. To create this file, we scan the text database and take all distinct words from the database. In the file, rows represent documents and columns represent all distinct words. If the words present in the document, we set 1 in the intersection of its document and word. Otherwise, we set 0. The examples of text database and transaction file are shown in Figure-2(a) and 2(b).

Doc_ID	List of word_ID
D100	text, data, mine
D200	data, classify
D300	data, associate
D400	text, data, classify
D500	text, associate
D600	data, associate
D700	text, associate
D800	text, data, associate, mine
D900	text, data, associate

Figure-2(a): Example of text database

	text	data	associate	classify	mine
D100	1	1	0	0	1
D200	0	1	0	1	0
D300	0	1	1	0	0
D400	1	1	0	1	0
D500	1	0	1	0	0
D600	0	1	1	0	0
D700	1	0	1	0	0
D800	1	1	1	0	1
D900	1	1	1	0	0

Figure-2(b): Example of transaction file

Then, the documents can be seen as transactions and the words can be seen as item and used the Apriori algorithm for mining the most frequent words for Boolean association rule.

The Apriori algorithm uses a bottom-up breadth-first approach to find the large itemsets. It starts from large 1-itemsets and then extends one level up in every pass until all large itemsets are found. For each pass, say pass k, there are three operations. First, append the large (k-1) - itemsets to L. Next, generate the potential large k-itemsets using the (k-1) – itemsets. Such potential large itemsets are called candidate itemsets C. The candidate generation procedure consists of two steps. They are Join step and prune step.

- **Join step**: generate k-itemsets by joining l_{k-1} with itself.
- **Prune step**: remove the itemset X generated from the join step, if any of the subsets of X is not large. Since any subset of a large itemset must be large.

Algorithm : Apriori. Find frequent itemsets using an iterative level-wise approach based on candidate generation.
Input : Database D, of transactions; minimum support threshold, min-sup.
Output : L, frequent itemsets in D.
Method : 1)L_k=Ø; k=0;
2)C_1=All distinct items in D
3)L_1=Large itemsets in C_1
4)While L_{k+1} is not empty
5)C_{k+1}=Candidate -gen (L_k)
6)L_{k+1}=Large itemsets in C_{k+1}
7)k++

8) Return UL

Figure -3: The Apriori algorithm

The Apriori algorithm performs two kinds of actions, namely, join and prune. In the join component, L_{k-1} is joined with L_{k-1} to generate the potential candidates. The prune component employs the Apriori property to remove candidates that have a subset that is not frequent words in the document corpus.

4. Mining steps of frequent words

In this section, the mining steps of text by scanning the transaction file are presented.

At first, all the documents are scanned in order to count the number of occurrences of each term. The set of candidate 1-termset, C_1, is shown in Figure 4(a).

C_1

Itemset	Sup-count
{text}	6
{data}	7
{associate}	6
{classify}	2
{mine}	2

Figure -4(a)

In this example, the minimum support count is set with 2. Then candidate support count are compared with minimum support count. The set of frequent 1-termsets, L_1, is shown in Figure 4(b). It consists of the candidate1-termset satisfying minimum support count.

L_1

Itemset	Sup-count
{text}	6
{data}	7
{associate}	6
{classify}	2
{mine}	2

Figure -4(b)

To discover the set of frequent 2-termsets, L_2, L_1 is joined with L_1 to generate a candidate 2-termsets, C_2. It consists of 2-termsets is described in Figure 4(c). Then the transaction file is scanned for count of each candidate.

The result of C_2 is shown in Figure 4(d).

C_2

Itemset
{text, data}
{text, associate}
{text, classify}
{text, mine}
{data, associate}
{data, classify}
{data, mine}
{associate, classify}
{associate, mine}
{classify, mine}

Figure -4(c)

C_2

Itemset	Sup-count
{text, data}	4
{text, associate}	4
{text, classify}	1
{text, mine}	2
{data, associate}	4
{data, classify}	2
{data, mine}	2
{associate, classify}	0
{associate, mine}	1
{classify, mine}	0

Figure -4(d)

Now, the set of frequent 2-termsets, L_2, is determined by comparing candidate support count with minimum support count. L_2 is shown in Figure -4(e) and that is satisfied the minimum support count.

L_2

Itemset	Sup-count
{text, data}	4
{text, associate}	4
{text, mine}	4
{data, associate}	4
{data, classify}	2
{data, mine}	2

Figure -4 (e)

L_2 is joined with L_2 to get the set of candidate 3-termsets, C_3. The joined result is {text, data, associate}, {text, data, mine}, {text, associate, mine}, {data, associate, classify}, {data, associate, mine} and {data, classify, mine}. But {text, associate, mine}, {data, associate, classify}, {data, associate, mine} and {data, classify, mine} are removed from C_3 because their subsets {associate, mine}, {associate, classify}and {classify, mine} are not members of L_2. Therefore, final pruning result is shown in Figure -4(f). And then the documents in transaction file are scanned and the support count of each candidate termset in C_3 is accumulated, as shown in Figure -4(g).

C_3

Itemset
{text, data, associate}
{text, data, mine}

Figure -4(f)

C_3

Itemset	Sup-count
{text, data, associate}	2
{text, data, mine}	2

Figure -4(g)

The candidate support count is compared with minimum support count to generate the L_3. It consists the candidate 3-termsets satisfying minimum support count and is shown in Figure -4(h).

L_3

Itemset	Sup-count
{text, data, associate}	2
{text, data, mine}	2

Figure -4(h)

Now, L_3 is joined with L_3 to obtain a candidate set of 4- termsets, C_4. The joined result is {term, data, associate, mine}. But this termset is pruned since its subset {data, associate, mine} is not frequent. Thus, C_4 does not exist and example of our mining steps is completed, having found all the frequent termsets.

5. Experimental Results

In this section, we will report the results of the experiment. Figure-5(a) shows the words in the database after preprocessing step. It includes words of 4 documents. Figure-5(b) shows the pairs of frequent word after mining the frequent words that are mined with minimum support 40%.

Figure-5(a): Words in database

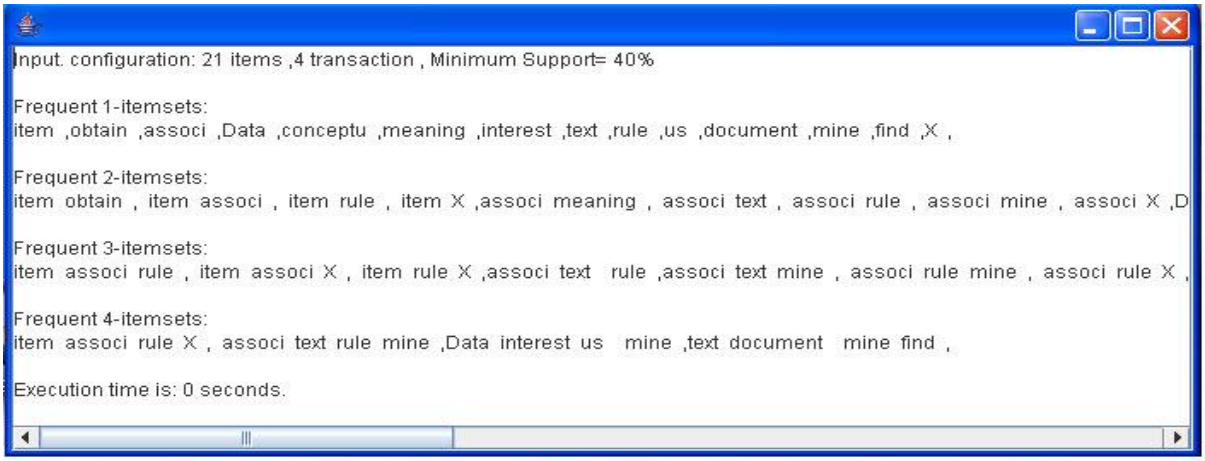

Figure-5(b): The experimental mining results

6. Conclusion

In this paper, a system for mining the text has been developed and it is based on the data mining system and text mining system. And then it had been used Apriori algorithm. Our system is very simple yet powerful because it can obtain more useful and meaningful results, richer text representations and to improve the efficiency of mining terms and their relations for real-world free texts. It can support of mining knowledge and concepts extraction from the documents. Moreover, it can provide document clustering and classifying that are very useful and efficient for Information Retrieval System.

References

[1] A. Murakami, N. Uramoto, H. Matsuzawa, T. Nagano, H. Takeuchi, and K. Takeda "A text-mining system for knowledge discovery from biomedical", Volume 43, Number 3, 2004.

[2] Ah-Hwee Tan, Kent Ridge Digital Labs, (21) Heng Mui Keng Terrance, "Text Mining: The state of the art and the challenges", Singapore.

[3] AMY D. WOHL, "Intelligent Text Mining Creates Business Intelligence".

[4] Brigitte Mathiak and Silke Eckstein, "Five Steps to Text Mining in Biomedical Literature".

[5] Catherine Blake and Wanda Pratt, "Better Rules, Fewer Features: A Semantic Approach to Selecting Features from Text".

[6] Dunja Mladenic and Marko Grobelnik, "Text and Web Mining".

[7] H. Ahonen, O. Heinonen, M. Klemettinen, and I.Verkamo, "Applying data mining techniques

in text analysis", Technical Report C-1997-23, University of Helsinki, 1997.
[8] Haralampos Karanikas and Babis Theodoulidis, "Knowledge Discovery in Text and Text Mining Software".
[9] Hany Mahgoub, Dietmar Rosner, Nabil Ismail and Fawzy Torkey, "A Text Mining Technique Using Association Rules Extraction", International Journal of Computational Intelligence, Volume 4, Number 1, 2007.
[10] Jacky W. W. Wan, Gillian Dobbie, "Mining Association Rules from XML Data using XQuery", The University of Auckland, Private Bag 92019, Auckland, New Zealand.
[11] Jiawei Han, Micheline Kanber, "Data Mining, Concepts and Techniques", page 225-235.
[12] Jon Atle Gulla, "Introduction to Text Mining and Web Mining".
[13]Kjetil Norvag, Trond Oivind Eriksen, and Kjell-Inge Skogstad, "Mining association rules in temporal document databases".
[14] Marcos M. Campos, "What is data mining", January 2006.
[15] Marti Hearst, "What is Text Mining", October 17, 2003.
[16] M. Hearst, "Untangling text data mining", In Proceedings of ACL'99: the 37th Annual Meeting of the Association for Computational Linguistics, pages 3–10, 1999.
[17] Mohanmed M. El Wakil, "Introduction Text Mining".
[18] M. Rajman and R. Besancon, "Text mining: Natural language techniques and text mining applications", In *Proceedings of the seventh IFIP Working Conference on Database Semantics*, 1997.
[19] Pak Chung Wong, Paul Whitney, Jim Thomas, "Visualizing Association Rules for Text Mining", Pacific Northwest National Laboratory.
[20] Raymond J. Mooney and Un Yong Nahm, "Text Mining with Information Extraction".
[21] R. Feldman and I. Dagan. Kdt – "knowledge discovery in texts", In *Proceedings of the First International Conference on Knowledge Discovery (KDD)*, pages 112–117, 1995.
[22] Ronen Feldman and Haym Hirsh, "Mining associations in text in the presence of background knowledge". In *Knowledge Discovery and Data Mining*, pages 343–346, 1996.
[23] Ronen Feldman, Moshe Fresko, Yakkov Kinar, Yehuda Lindell, Orly Liphstat, Martin Rajman, Yonatan Schler, and Oren Zamir. Text mining at the term level. In *Principles of Data Mining and Knowledge Discovery*, pages 65–73, 1998.