ISSN(Online): 2395-7638

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 4, July 2015

Pattern Discovery using Pattern Revealing in Text Mining

Balmukund Dubey[1], Thirumahal Rajkumar[2], Gayatri Dantal[3]

P.G. Student, Computer Engineering, SLRTC of Engineering, Mumbai, Maharashtra, India[1]

Assistance Professor, Information Technology, TSEC of Engineering, Mumbai, Maharashtra, India[2]

Assistance Professor, Computer Engineering, SLRTC of Engineering, Mumbai, Maharashtra, India[3]

ABSTRACT: In this new modern technology era most of the large amount of information is available in digital form. In this most of the data mining techniques have been perform different knowledge task. Because lots of business Information lives in the form of text formats. Text mining is a variation on the data mining and tries to find out the interesting pattern from the huge database. This Text Mining also called as Intelligent Text Analysis. In this text data mining is process of extracting information. In the last decade people using phrases for representation of large amount of data in the document and topic should be perform better results to the terms in this we examine the number of data mining method to give the effectiveness of the pattern. In that we are using a pattern detection method to solve the term based problem. And also give the better result to information retrieval system.

KEYWORDS: Data mining, Sequential pattern mining using Cmspade and Clasp, Pattern discovery, Text mining.

I. INTRODUCTION

There are contain the large number of the several applications like business management and market analysis and research and, it will profit by the or the user and employment of the information and information extracted from large amount of the database. That useful in the Information discovery will be viewed or look like a because that method of nontrivial is to be extraction of data from massive or huge amount of the databases, information that is local conferred within the knowledge discovery of the method, in the last part unknown and probably helpful for users. Data mining is to be containing the very important absolutely necessary step within the method of knowledge discovery in databases [4]. There are very important parts which contain the many types of data mining techniques are which represent the used sequential pattern mining and closed sequential pattern etc. this two pattern mining technique are very useful in the data mining. The very important concept is to be a Data mining and they can easy to be allow to the process of retrieving or accessing to the interesting or relevant data knowledge from the huge amount of the database or storage area like database [2].

In this proposed system, contain the very important part is to be discovery of patterns will be done efficient through the very important part is to be pattern evolution and another important part is to be pattern deploying technique [3]. And this technique useful in the system and it will not only find the useful patterns but also efficiently use the and update them and to be find the requirement of the relevant data and important requirement of the interesting information from the database. Always very difficult to solved the problem of low frequency and misinterpretation. In that the system is supposed to develop the concept of the knowledge discovery model and this model handle the problem and they can efficiently use and easy to update and understand the patterns [1].

Today Text mining is a very important part on a field of data mining [2] and tries to find the relevant patterns from large databases or documents. The mostly Text data Mining refers to the generally process of extracting interesting information and non-trivial information and knowledge from unstructured text in a database or document [1]. Text mining is a very useful technique that uses the helps the users finds useful interesting information from a large amount of text data. There are contain a Many text-mining methods have been developed in order to achieve the performance of retrieving useful information for users needs [4].

ISSN(Online): 2395-7638

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 4, July 2015

Most of the text mining uses the keyword-based, and others choose the phrase method technique to construct a text to representation of a set of number of documents. It is believed that the phrase-based method should be performs better than the method keyword-based ones as it is considered that more information is carried by a phrase than by a single term. In these phrase-based language was not powerful to the word-based. Because phrases method contain a less ambiguous and also contain a more concise meanings of the individual words or terms, the reasons of the depressing performance of the uses of phrases are: (1) phrases contain the same properties to words, (2) contain a low frequency of occurrence in a document (3) always generated a large number of redundant and noisy phrases in the document [1]

Paper is organized as follows. Section II presents the motivation and related work Section III. Describe the problem formulation and Section IV present the discover pattern and Section V present the deployed pattern Section VI presents experimental results showing results of images tested. Finally, Section VII presents conclusion.

II. MOTIVATION AND RELATED WORK

Lots of data mining methods has been proposed to represent text in past. Sequential mining is also studied in that since the first research work in [1]. Then Apriori like algorithm takes initiative for solving problem face in large dataset [7] but this algorithms perform well in case of short frequency of term. To solve disadvantage of Apriori algorithm, using the CmSpade and ClaSp, in order to improve efficiency of patterns along with many other algorithms are functional proved that they have improve the efficiency and better results, each of these algorithms work on different method of discovering frequent sequential pattern. However, the searching for useful and interesting patterns was still an open research problem The process of knowledge discovery may consist following steps: Data selection, Data pre-processing, Data transformation, Pattern discovery and Pattern evaluation [3]. In Data selection method using the generated to a dataset and selecting this dataset or a subset of data sources where method discovery is to be used. Then using a Pre-processing method which processes the data cleaning and noise removing in the document. It also includes the required information from the selected data fields, providing a strategy to the dealing with missing information or data and accounting the redundant information or data. For the case of web pages then transformation, pre-processed data needs to be transformed into a predefined format, depending on the data-mining task. This process needs to select an adequate type of features to represent data In the presence of these setbacks, sequential patterns used in data mining community [1,7] because patterns enjoy good statistical properties like terms. To overcome the disadvantages of phrase-based approaches, pattern-based approaches (or Pattern Taxonomy Models (PTM) [1,7]) have been proposed for IF from within the data mining community. In these patterns based method improvements of the effectiveness. And In that problem of information redundancy and noise, to solve the redundancy using a PTM method and adopts the concept of closed method, or pruned non-closed patterns method [1]. it is a very difficult issue challenging to the PTM to handle with the low frequency patterns problems because the measures used from data mining (e.g., “support” and “confidences”) to learn the profile turn out be not useful in the filtering stage. By way of, given a topic, and generated the high frequent pattern (normally a short pattern with large support) by a general pattern, or otherwise generated of pattern of low frequency pattern. In these terms indexing where term or words of high frequency (stop words) or very low frequency are not considered in a useful in a method.

III. PROBLEM FORMULATION

Definition (sequence database) Let I = { i_1 , $i_2 \dots \dots, i_l$ } be a set of items (symbols). An *itemset* $I_x = \{i_1, i_2 \dots \dots, i_m\} \subseteq I$ is an not in a order of a set of items. In these lexicographical order $\succ_{lex}$ is defined as any total order on I. Without losing of information, it is assumed that all itemsets are ordered according to $\succ_{lex}$ A *sequence* is an nothing but ordered of list of itemsets $s = (I_1, I_2 \dots \dots, I_n)$ such that $I_k \subseteq I (1 \leq k \leq n)$. A *sequence database SDB* is a contain a list of sequences database *SDB* = (*s*1, *s*2,..., *sp) also* contain a sequence identifiers that is SID = 1, 2...,*p [8]*. **Example.** sequence database is shown in Table 1 in below. It will contain 4 sequences having the SID = 1, 2, 3 and also contain a single letter represents an item. And also contain curly brackets to the each item in the itemset. The initial sequence contain a 5 elements *{a, b}, {c}, {f, g}, {g}, {e}* itemsets. It also tell that items *a* and *b* occurred at the same time in a first sequence, and used by *c*, and then *f, g* and lastly *e [8]*.

ISSN(Online): 2395-7638

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 4, July 2015

Table 1 Sequences in a document

Sid	Sequences
1	{a,b},{c},{f,g},{g},{e}
2	({a, d},{c},{b},{a, b, e, f}
3	{a},{b},{f},{e}
4	{b},{f, g}

Table 2 Sequential patterns found

ID	Pattern	Support
P1	{a},{f}	3
P2	{a},{c},{f}	2
P3	{b},{f,g}	2
P4	{g},{e}	2
P5	{c},{f}	2
P6….	{b}	4

Definition (sequence containment) A sequence $s_a = (A_1, A_2, \ldots A_n)$ is said that it will *contain the* sequence $s_b = (B_1, B_2, \ldots B_n)$ if only if there exist integers $1 \leq i_1 < i_2 < \ldots < i_n \leq m$ such that $A_1 \subseteq B_{i1}$, $A_2 \subseteq B_{i2}$, , ..., $A_n \subseteq B_{in}$ (denoted as $s_a \sqsubseteq s_b$). **Example.** Sequence 4 in Table 1 is contained in Sequence 1.

Definition (prefix) A sequence $s_a = (A_1, A_2, \ldots A_n)$ is a *prefix* of a sequence $s_b = (B_1, B_2, \ldots B_n), \forall n < m$, iff $A_1 = B_1, A_2 = B_2, \ldots A_{n-1} = B_{n-1}$ and the first $|A_n|$ items of B_n according to $\succ_{lex}$ are equal to A_n.

Definition (support) The *support* of a sequence s_a in a sequence database *SDB* is defined as the number of sequences $s \in SDB$ such that $s_a \sqsubseteq s$ and is denoted by $supp_{SDB}(s_a)$.

Definition (sequential pattern mining) Let $minsup$ be a threshold set by the user and *SDB* be a sequence database. A sequence *s* is a *sequential pattern* and is deemed *frequent* iff $supp_{SDB}(s) \geq minsup$. The *problem of mining sequential patterns* is to discover all sequential patterns. **Example.** Table 2 shows 6 of the 29 sequential patterns found in the database of Table 1 for $minsup = 2$.

Definition (closed sequential pattern mining) A sequential pattern s_a is said to be *closed* if there is no other sequential pattern s_b, such that s_b is a super pattern of s_a, $s_a \sqsubseteq s_b$, and their supports are equal. The problem of *closed sequential patterns* is to discover the set of closed sequential patterns, which is a compact summarization of all sequential patterns

Definition (horizontal database format) a *sequence database in horizontal format* is a database where each entry is a sequence. **Example.** Table 1 shows an horizontal sequence database.

Definition (vertical database format) A *sequence database in vertical format* is a database where each entry represents an item and indicates the list of sequences where the item appears and the position(s) where it appears. **Example.** Table 3 shows the vertical representation of the database of Table 1.

ISSN(Online): 2395-7638

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 4, July 2015

From the vertical representation, a structure named *IdList* can be associated with each pattern. IdLists allow calculating the support of a pattern quickly by making join operations with IdLists of smaller patterns. To discover sequential patterns using IdLists, a single database scan is required to create IdLists of patterns containing a single items, since IdList of larger patterns are obtained by performing the aforementioned join operation. Several works proposed alternative representations for IdLists to save time in join operations, being the bit set representation the most efficient one.

A	
SID	Itemsets
1	1
2	1,4
3	1
4	

b	
SID	Itemsets
1	1
2	3,4
3	2
4	1

c	
SID	Itemsets
1	2
2	2
3	
4	

d	
SID	Itemsets
1	
2	1
3	
4	

E	
SID	Itemsets
1	5
2	4
3	4
4	

f	
SID	Itemsets
1	3
2	4
3	3
4	2

G	
SID	Itemsets
1	3,4
2	
3	
4	2

Table 3 the vertical representation of the example database

Algorithm 1 SPADE (*SDB, minsup*)[8]
1. Scan *SDB* to create *V* (*SDB*) and identify F_1 the list of frequent items.
2. **ENUMERATE** (F_1). //call Enumerate method
.....call Enumerate method.....
ENUMERATE (an equivalence class *F*)
1. **FOR** each pattern $A_i \in F$
2. OutputA_i.
3. $T_i := \emptyset$.
4. **FOR** each pattern $A_j \in F$, with $j \geq i$
5. R = MergePatterns(A_i , A_j)
6. **FOR** each pattern $r \in R$
7. **IF** $sup(R) \geq minsup$ **THEN**
8. $T_i := T_i \cup \{R\}$;
9. **ENUMERATE** (T_i)

Candidate Generation in SPADE: The pseudocode of SPADE is shown in Algorithm 1. The SPADE procedure takes as input a sequence database *SDB* and the *minsup* threshold. SPADE first constructs the vertical database *V* (*SDB*) and identifies the set of frequent sequential patterns F_1 containing frequent items. Then, SPADE calls the ENUMERATE procedure with the equivalence class of size 0. An *equivalence class* of size k is defined as the set of all frequent patterns containing k items sharing the same prefix of $k - 1$ items. There is only an equivalence class of size 0 and it is composed of F_1.

The ENUMERATE procedure receives an equivalence class F as parameter. Each member A_i of the equivalence class is output as a frequent sequential pattern. Then, a set T_i, representing the equivalence class of all frequent

ISSN(Online): 2395-7638

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 4, July 2015

extensions of A_i is initialized to the empty set. Then, for each pattern $A_j \in F$ such that $i \succ_{lex}$ j, the pattern A_i is merged with A_j to form larger pattern(s). For each such pattern *r*, the support of *r* is calculated by performing a join operation between IdLists of A_i and A_j . If the cardinality of the resulting IdList is no less than *minsup*, it means that *r* is a frequent sequential pattern. It is thus added to T_i. Finally, after all pattern A_j have been compared with A_i, the set T_i contains the whole equivalence class of patterns starting with the prefix A_i. The procedure ENUMERATE is then called with T_i to discover larger sequential patterns having A_i as prefix. When all loops terminate, all frequent sequential patterns have been output [8]

B: Close sequential pattern (ClaSp):

In this section, we formulate and explain every step of our Clasp algorithm. Clasp has two phases: 1) generates a subset of FS(and subset of FCS) called Frequent Closed Candidates (FCC), that is kept in memory; and 2) Executes a post-pruning phase to eliminate from FCC all non-closed sequences to finally obtain exactly FCS [9].

Algorithm 2 Clasp

1. F_1 = {frequent 1-sequences}
2. $FCC = \emptyset, FCS = \emptyset$
3. **FOR** all $i \in F_1$ **DO**
4. F_{ie} = { frequent 1-sequences greater than i}
5. FCC_i = DFS-PRUNING(I, F_1, F_{ie})
6. $FCC = FCC \cup FCC_i$
7. **END FOR**
8. FCC = N-ClosedStep(FCC)

Ensure: The final closed frequent pattern set FCS

Algorithm 2 ClaSP shows the pseudocode corresponding to the two main steps. It first finds every frequent 1-sequence, and after that, for all of frequent 1-sequences, the method *DFS-Pruning* is called recursively to explore the corresponding subtree (by doing a depth-first search). *FCC* is performed when this process is completed for all of the frequent first (1)-sequences and, once a completed of the Frequent sequence the algorithm stop or ends removing the non-closed sequences that generated in *FCC algorithms*. Algorithm, *DFS-Pruning*, executes recursively both the candidate generation (by means of i-extensions and s-extensions) and the support checking, returning a part of *FCC* relative to the pattern *p* taken as parameter. The method takes as parameters two sets with the candidate items to do s-extensions and i-extensions respectively (and sets).

IV. USING DISCOVERED PATTERN

Algorithm 3 Deploying with Relevance Method [1]
Input: a set of positive documents (D^+), minimum Support , $\min\_sup$.
Output: New set of terms, a set of vectors (Δ).
Method:
1)Δ← ∅
2) FOR each document d in D^+ DO BEGIN
3) Extract 1Term frequent pattern PL from d
4) SP = SPADE (SDB, min_sup) // call Algorithm CmSpade
5) $\vec{d} \leftarrow \emptyset$
6) FOR each pattern p in SP DO BEGIN
7) $\vec{d} \leftarrow \vec{d} \oplus p'$ // p' is the expanded form of p
8) END FOR
9) $\Delta \leftarrow \Delta \cup \{\vec{d}\}$ p
10) END FOR

ISSN(Online): 2395-7638

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 4, July 2015

The input of the algorithm PDM are a set of positive document and a pre specified minimum support. In line 4 of this algorithm, a set of sequential pattern is discovered by calling the algorithm CmSpade for each document. So for only positive document are consider and used in this approach. The use of information from negative document is another issue with reference to pattern evolution which will be investigated. At the next step in 6 to 8, each pattern is firstly transferred into an expanded from and then merged into a temporary storage using pattern composition operator as a result, the deployed pattern for each document is obtained the deployed pattern of five sample document can be expressed as Δ:

$\overrightarrow{d_1}$ = {(carbon, 2), (emiss, 1),(air,1), (pollut, 1)}
$\overrightarrow{d_2}$ = {(greenhouse, 1), (global, 2), (emiss, 1)}
$\overrightarrow{d_3}$ = {(greenhouse, 1), (global, 1), (emiss, 1)}
$\overrightarrow{d_4}$ = {(carbon, 1), (air, 2), (antarct, 1)}
$\overrightarrow{d_5}$ = {(emiss, 1), (global, 1), (pollut, 1)}

We keep Δ as the training results for the further processing in the pattern evolution stage. To deploy patterns, each vector of document in Δ is normalised first and the feature space is updated by summing up the weight value of each corresponding until all vectors in Δ are processed. For instance the gradual updating of feature space for the above mentioned example is as follows.

$\overrightarrow{d_1}$ = {(carbon, 2/5), (emiss, 1/5),(air,1/5), (pollut, 1/5)}
$\overrightarrow{d_1} \oplus \overrightarrow{d_2}$ = {(carbon, 2/5), (emiss, 9/20),(air,1/5), (pollut, 1/5), (greenhouse, 1/4), (global, 1/2)}
.
.
$d^{\wedge}$ = {(carbon, 13/20), (emiss, 67/60),(air,7/10), (pollut, 8/15), (greenhouse, 7/12), (global, 7/6), (antaract,1/5)}

$$W(p) = \frac{|\{da|da\} \in D^+, p \subseteq da|}{|\{da|da\} \in D, p \subseteq da|}$$

Where *D* is the training set of documents and *D*+ indicates the set of positive documents in *D*. Two problems arise when using the above-mentioned weighting function. One is the low pattern frequency problem, which is mainly because it is hard to match patterns in documents when the length of the pattern is long. The other problem is that those patterns, which are specific to a topic may gain a lower score than those for general patterns. In otherwords, the information carried by the specific patterns cannot be estimated by the weighting function. That can be overcome with the following algorithm 1.

The inputs of the algorithm PDM are a set of positive documents and a pre-specified minimum support. In line 4 of this algorithm, a set of sequential patterns is discovered by calling the algorithm CmSpade for each document. So far, only positive documents are considered and used in this approach. The use of information from negative documents is another issue with reference to pattern evolution.

V. DEPLOYED PATTERN EVOLUTION

Algorithm 4 DPEvolving (Ω, D^+, D^-)
Input: A list of deployed patterns Ω; a list of positive and negative documents, D^+ and D^-
Output: A set of term weight pairs $\vec{d}$
Method:
1.$\vec{d} \leftarrow \emptyset$ // It give minimum threshold Value
2. $\tau \leftarrow$Threshold (D^+)

ISSN(Online): 2395-7638

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 4, July 2015

3. FOR each negative document nd in D^- DO BEGIN
4. IF Threshold ($\{nd\}$) $> \tau$ THEN
5. $\Delta_p = \{d_p \in \Omega | \text{termset(dp)} \cap nd \neq \emptyset\}$
6. Shuffling (nd, Δ_p) // call shuffling Algorithms
7. END IF
8. FOR each deployed pattern dp in Ω DO BEGIN
9. $\vec{d} \leftarrow \vec{d} \oplus dp$
10. End for
11. END FOR

The evolution of deployed pattern is implemented by the algorithm 4 DPEvolving. The inputs of this algorithm are a list of deployed pattern Ω, a list of positive and negative document, D^+ and D^-. The output is a set of term weight pairs which can be used directly in the testing phase. Line 2 in DPEvolving is used to estimate the threshold for finding the interesting negative documents. Line 3 to 5 is the process of discovering the offenders of negative document.

Therefore, a set of deployed patterns that share the same patterns with negative document is collected for further processing. Once all the offenders are found, another algorithm 5 shuffling is then called to perform the main task for this algorithm.

Algorithm 5 Shuffling (nd, Δ_p)

Input: a negative document nd and a list deployed patterns Δp.
Output: updated deployed patterns.
Method:
1: FOR each deployed patterns dp in Δ_p DO BEGIN
2: IF $termset(dp) \subseteq nd$ THEN // complete conflict offender
3: Ω=Ω−$\{dp\}$
4: ELSE // partial conflict offender
5: $offering' = (1 - \frac{1}{\mu}) \times \sum_{t \in termset(dp)} \{t.weight | t \in nd\}$
6: base = $\sum_{t \in termset(dp)} \{t.weight | t \notin nd\}$
7: FOR each term t in $termset(dp)$ DO BEGIN
8: IF $t \in nd$ THEN //shrink offender weight
9 $t.weight = (\frac{1}{\mu}) \times t.weight$
10: ELSE // shuffle weight
11: $t.weight = t.weight \times (1 + offering \div base)$
12: END IF
13: END FOR
14: END IF
15: END FOR

The task of (algorithm 5) Shuffling is to tune the weight distribution of terms within a deployed pattern. A different strategy is dedicated in this algorithm for each type of offender. As stated in line 3 in the (algorithm 5) shuffling, for the type of complete conflict offenders, the deployed patterns are removed from the deployed pattern set Ω since all element within the deployed pattern are held by the negative document indicating they can be discarded for preventing interference from these possible "noises".

The parameter *offering*' is used in line 5 for the purpose of temporarily storing the reduced weight of "Y" type terms in a partial offender. The *offering'* is a part of offering. The *offering* is the sum of weight of terms in a deployed pattern where these terms also appear in a negative document. Given a deployed pattern dp and a negative document nd, the value of *offering* can be estimated by the following equation:

ISSN(Online): 2395-7638

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 4, July 2015

$$offering(dp) = \sum_{(t_{,t.weight})\in\beta(dp),t\in end)\in end} t.weight$$

VI. RESULTS AND DISCUSSION

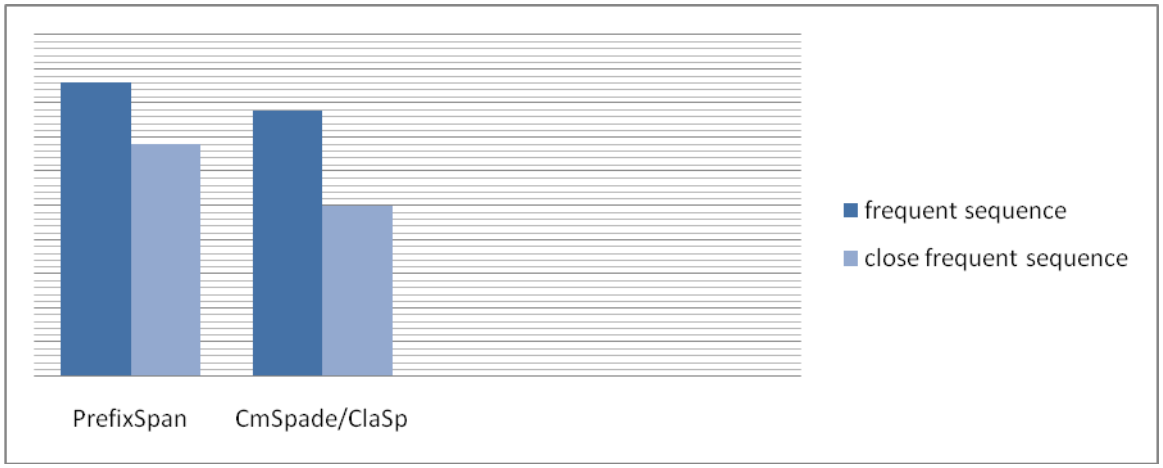

Fig 1 Comparisons between Prefixspan and Cmspade/Clasp

In this Figure 1 shows the comparison of Prefixspan and Cmspade/Clasp algorithms. This is based on the frequent sequences and close frequent sequences. In this Prefixspan algorithm generate a large close frequent pattern but Cmspade/Clasp generate a small frequent close pattern.

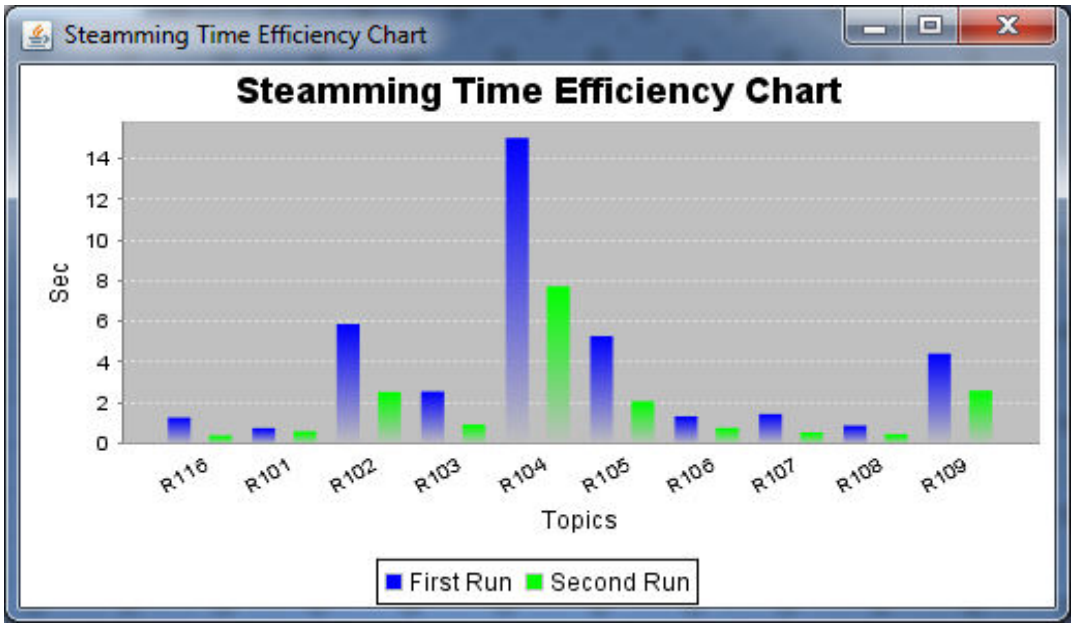

Fig 2 Graph of Steamming time efficiency chart

ISSN(Online): 2395-7638

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 4, July 2015

In this Figure 2 here we present the Steamming time Efficiency chart. In that we taking a ten topics and check the timing when we run first time it will take more time but when we run second time it will take less time compare to first time.

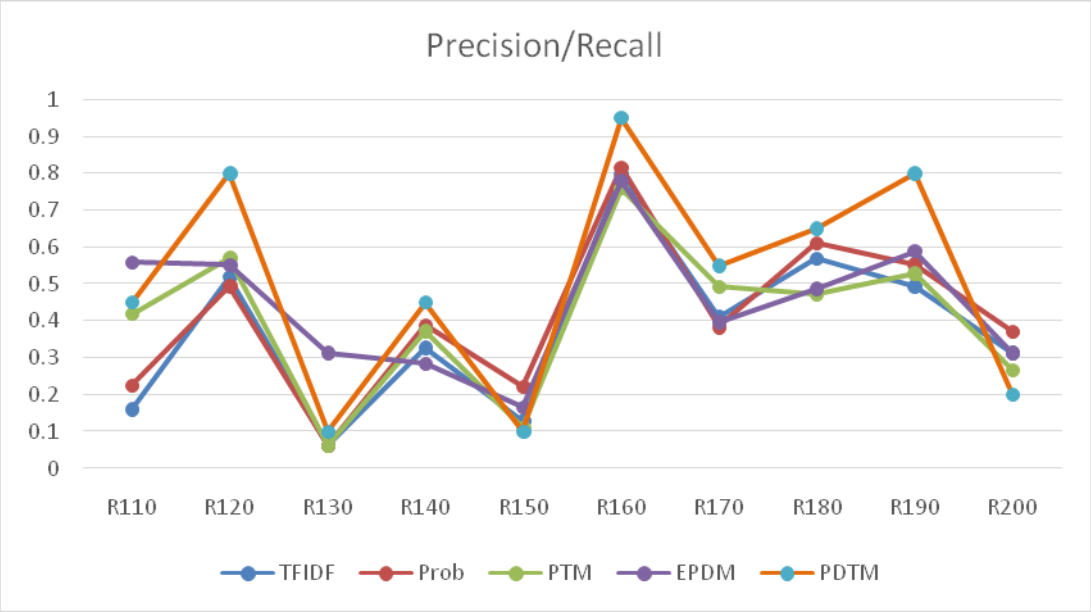

Fig 3 Comparison of precision and recall for different method on topic

In this Figure 3 we have comparison of the precision and recall method on the top ten topics which is based on the support. In this we have show that PTM method better on the Prob and TFIDF and EPDM is more powerful in PTM but PDTM is better of all method.

VII. CONCLUSION

In this we investigated issues for mining Frequent and Closed frequent sequential patterns in large data sets and generated the large amount of inefficiency and redundancy of frequent sequential pattern mining problem. In this we formulated Clasp to mine frequent closed sequences efficiently. This is the first piece of work to solve closed sequential pattern mining problem. In this Experiment we show that Clasp outperforms PrefixSpan by more than one order of magnitude and it is capable of mining long frequent sequences in a large data set with low minimum support without loss of information.

REFERENCES

[1] Operational Pattern Revealing Technique in Text mining 2014 IEEE Students Conference on Electrical and Computer Science 978-1-4799-2526-1/200.
[2] Ning Zhong, Yuefeng Li Effiective pattern discovery in text mining IEEE Transaction on knowledge and data engineering, vol,no.1 january 2012.
[3] S-T, Wu.Y.Li, Y.Xu, B.P chen. and "Automatic pattern taxonomy extraction for the web mining". In processing of the IEEEJWI/AM international conference on web intelligence (W104), pages 242-248,2004
[4] S-T Wu.Y.Li,Y.Xu "Deploying approaches for patterns refinements for the text mining". In proceeding of ICDM, pages 1157-1161,2006
[5] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu, PrefixSpan: Mining sequential patterns efficiently by prefix projected pattern growth, Proceedings of Int. Conf. on Data Engineering (ICDE02), Heidelberg, Germany, 2001, pp. 215-224.
[6] M. Zaki Spade: An efficient algorithm for mining frequent sequences, machine learning vol,40,2001,pp.31-60.
[7] Agrawal, R,Ramakrishnan, S: Mining sequential patterns. In proc.11th international conferencedata engineering,pp.314.IEEE(1995).

ISSN(Online): 2395-7638

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 4, July 2015

[8] Fournier –Viger Antonio Gomariz, manuel campos and Rincy Thomas "Fast vertical mining of sequential patterns using co-occurance information" V.S Tesing et al (eds):PAKDD 2014,part 1, LNAI 8443,pp.4052. Springer international publishing Switzerland 2014.

[9] Antonio Gomariz manuel campos, Roque marin and bart Goethals. Clasp: efficient algorithms for mining closed sequence J.pei et al:.PAKDD 2013,part 1, LNAI 7818,pp.50-61.Springer-verlag berlin Heidelberg 2013