A New Classification Approach using Gapped Subsequences

Kusum Sharma[1], Asha Ambhaikar[2]

[1] M. Tech Scholar, Department of Computer Science & Engineering
RCET, Bhilai, Chhattisgarh, India

[2] HOD, Department of Computer Science and Engineering,
RCET, Bhilai, Chhattisgarh, India

Abstract: ***An important purpose of sequence analysis is to find the distinguishing characteristics of sequence classes. Given in one set of sequence pattern we introduce the problem of gapped subsequences and purpose is to find efficient patterns and provide a classification for these patterns. Since every pattern has a class label so we find these labels. Sequence is an ordered list of events, the pattern we would like to mine is called repeated gapped subsequences which is a subsequence (possibly occurred with some gaps within two successive sequences). To find the patterns we introduce the concept of frequent support to measure how frequently a pattern repeats in a sequence. As compare to other sequence pattern mining problems repeated supports captures not only repetition of patterns in other sequences but also repetition within sequence. Here we present a classification methodology for sequence classification based on these gapped frequent patterns.***

Keywords: Gapped subsequences, frequent support, sequence classification, sequential pattern mining.

1. Introduction

There is various sequence data is available from wide spread of applications, in which sequence of events and transactions correspond to main source of information. Examples of sequential data may include customer purchase histories, software execution patterns, and credit card fraud detection, sequence of words in text data, bio sequences like DNA and protein sequences. The task of discovering frequent subsequences as pattern, as well as classify them is important research area in field of data mining. Classification is a procedure in which given a collection of training data sets , each of one containing a set of attributes , with one of them in the class , to find a model with a class attribute as a function of values for other attributes. The result of the classification is that new sequences are assigned to a class as exact as possible.

Sequential pattern classification is an important data mining problem that arises in many real world applications, such as market data analysis, customer shopping trends, speech recognition, intrusion detection, bio informatics function prediction etc. A rich literature review contributes to it such as [1], [2], [3], [4], [5], [6], [7], [8] and [9]. In this research direction we propose the problem of classification of closed repeated gapped subsequences from large data sets.

Gapped subsequences are a subsequence which appears in sequence data base eventually with gaps between two successive set of sequences. In this paper we give a study about capturing repeated patterns, not only patterns repeated in different sequences but also repeated in same sequence. That means we are going to find the number of occurrences in given sequence data. As we know that subsequences are more informative because they have some class label, so with the help of these labels we can provide classification with the help of these subsequences. Given a sequence data, a sequence classifier assigns a finite class label to this, data mining and machine learning algorithms an effective approach to design sequence classifier.

Most of the existing data mining methods are designed for solving some specific problems independently. On the other hand few compound methods of data mining have been used to solve complex sequence data mining problem.

In this work we propose a methodology for sequence classification by using closed repeated gapped sub sequences. The entire work is divided in to two sub parts

1. Pattern extraction by mining closed repeated gapped sub sequences.
2. Classify these patterns according to classifier.

Before starting about these topics we must have to know about repeated gapped sub sequences. Suppose we have two sequences S_1 = AABCABCDEABCABCDEAABC and S_2 = ABCDE. Let us assume that these records are generated by retail shop that records the purchasing patterns of customer. Suppose A = Product Request Placed, B = Product Request in Process, C = Product Purchased, D = Product Delivered, and E = Product Request Cancelled.

Consider patterns AB (Customer process a request after customer place an order for one of them). We have total 5 instances Consider DE (Deliver the product to customer after the customer cancels the request). Although, these subsequences repeat frequently in a given sequence. As analyzing these two sequences subsequence ‘AB’ repeats more frequently than DE. This information is more useful to differentiate the purchasing trends.

Now, we consider only the non overlapped features of a given sequence. Repeated sub sequences are informative but due to large sum of counting, it is not useful for defining the support. These non overlapped features are used to calculate how frequently a pattern appears in a sequence data.

2. Literature Survey

In the data mining community, the computation of the sequential patterns has been studied since 1995, e.g., [6, 7, 8, 10, 11, 12, 13]. It has lead to several algorithms that can process huge sets of sequences. These algorithms use three different types of approaches according to the way they evaluate the support of sequential pattern candidates. The first family contains algorithms that are based on the A-Priori scheme [1] and that perform a full scan of the database to evaluate the support of the current candidates, e.g., [1, 2, 10]. In these approaches, a particular effort is made to develop specific structures to represent the sequential patterns candidates to speed-up the support counting operations (e.g., as used in [10]). The second family (e.g., [4, 6, 9, 11, 13]) contains algorithms that try to reduce the size of the dataset to be scanned by performing projections of the initial database. The last family (e.g., [11, 13]) concerns algorithms that keep in memory only the information needed for the support evaluation. These algorithms are based on the so-called occurrence lists that contain the descriptions of the location where the pattern occur in the dataset. The projection database and occurrence list approaches seem to be more efficient than the first one in the case of low support threshold and long sequential patterns since the occurrence lists and the projected databases become more and more smaller.

Apriori-Based Method (GSP: Generalized Sequential Patterns) (Srikant & Agrawal, 1996) The Apriori property of sequences states that, if a sequence S is not frequent, then none of the super-sequences of S can be frequent. Vertical Format-Based Method (SPADE: Sequential Pattern Discovery using Equivalent Class) (Zaki, 2001). This is a vertical format sequential pattern mining method. SPADE first maps the sequence database to a vertical id-list database format which is a large set of items <SID (Sequence ID), EID (Event ID)>. Sequential pattern mining is performed by growing the subsequences (patterns) one item at a time by Apriori candidate generation FreeSpan (Han, Pei, Asl, Chen, Dayal, & Hsu, 2000)& Prefix Span (Pei, et al., 2001)These methods help in avoiding the drawbacks of the Apriori based methods. FreeSpan (Frequent pattern projected Sequential pattern mining) uses frequent items to recursively project sequence databases into a set of smaller projected databases and grows subsequence fragments in each projected database. This process partitions both the data and the set of frequent patterns to be tested, and confines each test being conducted to the corresponding smaller projected database Constraint Based Methods

Conventionally, users can specify only min_sup as a parameter to a sequential pattern mining algorithm. There are two major difficulties in sequential pattern mining: (1) effectiveness: the mining may return a huge number of patterns, many of which could be uninteresting to users, and (2) efficiency: it often takes substantial computational time and space for mining the complete set of sequential patterns in a large sequence database. To prevent these problems, users can use constraint based sequential pattern mining for focused mining of desired patterns. Mining frequent episodes in a sequence of events studied by (Mannila, Toivonen, & Verkamo, 1997) [6] can also be viewed as a constrained mining problem, since episodes are essentially constraints on events in the form of acyclic graphs. The classical framework on frequent and sequential pattern mining is based on the anti-monotonic Apriori property of frequent patterns. Mining Closed Repetitive Gapped Subsequences (Ding, Lo, Han, & Khoo, 2009) Patterns often repeat multiple times in a sequence e.g., in program execution traces, sequences of words (text data), credit card usage histories. Given two sequences like S1 = AABCDABB, S2 = ABCD, is pattern AB more frequent then CD? To answer this question, one needs to define a notion of repetitive support, sup (P) as max {|INS|: INS is a set of non-overlapping instances of P}. The aim is to maximize the size of the non overlapping instance set

3. Problem Statement

The problem of classification of closed repeated gapped subsequences. Given a sequence database and support threshold MinFreSupport, to find the complete set of frequent pattern with frequently repeated support not less than MinFreSopport. After finding these frequent non overlapped patterns we can give a classification based on such support.

Our problem definition is more crucial when repetition of pattern in same sequence is more important. Following are the some examples.

Repeated subsequences may correspond to frequent share marketing behaviors to forecast share marketing trend over a set of long share marketing record tracks. In above discussed example S_1 and S_2 some patterns (share market behaviors) like DE it appears in every sequence only once or twice but AB appears many times in one sequence but may not be repeated frequently in other but it is still useful with the help of classification to analyze share market behavior.

Repeated gapped subsequence can also be used in biological sequences like DNA, RNA and protein sequences to analyzing and finding new patterns for further changes in genomic analysis.

These are also used to represent frequent software program execution behavior. Program execution is a combination of different set of paths depending on the input, a set of program execution path each corresponding to potentially different sequences should be analyzed. As well as due to presence of loops and decisions patterns can repeats many time within each sequence and corresponding instance in sequences may have large gaps. Program execution frequent patterns may appear in many different sequence or repeat frequently only in few sequences.

3.1 Contributions

We propose and study the problem of classification of closed repeated gapped subsequences. As one step ahead our work is complement in research field of sequential data mining. We define support/frequent growth i.e. occurrence of pattern repeated in different sequence and also repeated with in sequence which represent some interesting class labels over a long data sequences.

4. Methodology

4.1 Repeated gapped subsequences

4.1.1 Definitions

Let ε be a set of events. A sequence S is an ordered list of events denoted by S = <e_1, e_2, e_3,e_4,……..> where $e_i \in \varepsilon$ is an event. An input sequence database is a set of sequences denoted by SD = {S_1,S_2,S_3……S_n}.

4.1.2 Subsequence

Sequence S = e_1,e_2,e_3…..e_i is a subsequence of other sequence S' = e_1',e_2'……e_j' (i≤j), denoted by S≤ (Subset) S'(S' is a super sequence of S)
A frequent pattern P_f = e_1,e_2,e_3,…..$e_{m\ is}$ also a sequence. For two pattern P_f and P_f' if P_f is a subsequence of P_f'. P_f is a sub pattern of P_f' and P_f' is a super patterns of P_f.
Instances of Patterns : For a pattern P_f in a sequence database SD = {S_1,S_2,S_3…..} if <l_1,l_2,l_3,….> is a position of pattern P = <e_1,e_2,e_3….> in S ∈ SD.
For defining MinFreSupport , we have to capture both the occurrence of pattern in different sequences as well as in the same sequence.

4.1.3 Overlapped Pattern instances

Two instances of patterns P_f = < $e_1e_2e_3$….> in SD = (S_1,S_2,S_3 ……) are overlapped if positions in both sequences are equal or non over lapped if positions are not equal.

We use S(P_f) to denote the set of set of instance of P_f in S and use SD(P_f) to denote the set of instances of P_f in SD.

4.2 Mining Closed Repeated Gapped Subsequences

Based on definition of repeated support a closed pattern P_f is said to be frequent if support(P_f) ≥ MinFreSupport where MinFreSupport is defined by user. Our aim is to find all frequent patterns in sequence data and MinFreSupport.

4.3 Classification of Closed Repeated Gapped Subsequences

Next step in our work is classify the given sequence data according to extracted features with the help of previous steps. The classification methodology consists of two stages. In the first stage sequence classification model based on patterns extracted from previous step is created. The first stage is similar to the CBS_CLASS algorithm which builds sequence classification model based on extracted patterns. This methodology is introduced here to add some extra weights with every extracted patterns.

4.4 Algorithm for pattern extraction and classification

Algorithm 1 MinSupComp(SD, P_f): Compute Support (Set)

Input: sequence data SD= **{**S**1**, S**2**, . . . , Sn**}**;
Pattern P_f = e**1**e**2** . . . e**m**.
Output: support set I of pattern P_f in SD.
1: I ← **{(**i, _l**1**_**)** | for some i, S**i[**l**1]** = e**1}**;
2: for j = **2** to m do
3: I ← PATgrow**(**SD, e**1** . . . e**j**−**1**, I, e**j)**;
4: return I (|I| = sup**(**P_f**)**);

Algorithm 2 PATgrow(SD, P, I, e): Pattern Growth

Input: sequence data SD = **{**S**1**, S**2**, . . . , Sn**}**; pattern P_f = e**1**e**2** . . . e**j**−**1**; leftmost support set I of P_f; event e.
Output: a leftmost support set I+ of pattern P_f ◦ e in SD.
1: for each S**i** ∈SD s.t. I**i** = I ∩ S**i(**P**)** ≠**Φ** (P_f has Instances in S**i**) in the ascending order of i do
2: last position ← **0**, I+**I** ← **Φ**;
3: for each **(**i, _l**1**, . . . , l**j**−**1**_**)** ∈**i** = I ∩ S**i(**Pf**)**
In the right-shift order (ascending order of l**j**−**1**) do
4: l**j**← next**(**S**i**, e, **max{**last position, l**j**−**1})**;
5: if l**j**= ∞ then break;
6: last position ← l**j**;
7: I+**I** ← I+^{I} ∪**{(**i, _l**1**, . . . , l**j**−**1**, l**j**_**)}**;
8: return I+ = ∪**i**≤**i**≤**N**I+^{i};
Subroutine next**(**S, e, lowest**)**
Input: sequence S; item e; integer lowest.
Output: minimum l s.t. l > lowest and S**[**l**]** = e.
9: return **min{**l | S**[**l**]** = e and l > lowest**}**;

Algorithm 1 can compute the left most support set I if pattern Pf in SD.
Pattern Growth Algorithm 2 : Pattern growth operation PATGrow (SD,Pf,I,e) is an important algorithm for computing repeated support as well as finding closed frequent patterns. Given a support set I of pattern Pf in sequence data SD and an event e it extends I to I^{+} Pfoe. To achieve this , for each instance **(**i, _l**1**, . . . , l**j**−**1**_**)** ∈**i** = I ∩ S**i(**Pf**)** we find the minimum l**j ,** lj > max(last_Position, l_{j-1}) and Si[lj] = e by calling next**(**S**i**, e, **max{**last position, l**j**−**1}.** When such l_{j-1} cannot be found stop scanning, otherwise continue till get the lowest last position as minimum support.

Algorithm 3 GSgrow: Mining All Frequent Patterns

Input: sequence data SD = **{S1, S2, . . . , Sn}**;
Threshold MinFreSupport.
Output: **{P$_f$ |** sup(**P$_f$**) ≥**M**inFreSupport**}**.
1: **E** ← all events appearing in SD; Fre← **Φ** ;
2: for each **e** ∈**E** do
3: **P** ← **e**; **I** ← **{(i, (l))** | for some **i, S**i[**l**] = **e}**;
4: mineFre(SD**, P$_f$, I**);
5: return Fre;
Subroutine mineFre(SD**, P$_f$, I**)
Input: sequence database SD = **{S1, S2, . . . , Sn}**; pattern **P$_f$** = **e1e2 . . . ej−1**; support set **I** of pattern **P$_f$** in SD.
Objective: add all frequent patterns with prefix **P$_f$** into Fre.
6: if |**I**| ≥ **M**inFreSupport then
7: Fre← Fre ∪**{P}**;
8: for each **e** ∈**E** do
9: **I+** ← PATgrow(SD**, P$_f$, I, e**);
10: mineFre(SD**, P** ◦ **e, I+**);

Algorithm 4 ClosePATgrow: Mining Closed Frequent

Patterns

... ...
6: if $|\mathbf{I}| \geq$ min sup and °¸(LBCheck($\mathbf{P_f}$) = prune) then
7: if CCheck($\mathbf{P_f}$) = closed then Fre← Fre ∪{$\mathbf{P_f}$};

Algorithm 3 GSGrow shares similarity with other pattern growth algorithms like PrefixSpan in the sense that both of them traverse the pattern in a depth first way.

Algorithm 4 ClosePATGrow is designed to find closed frequent pattern for given sequence data. This algorithm is useful to prune the search space.

These all algorithms are used to find the frequent patterns from given sequence data. Now after finding these all frequent patterns we are going classify them. For classification the data set can be defined by D = {Si,ci} I = 1,2,3….ls where Si is sequence of patterns and ci its Class label with lc different class sets and ls is the number of frequent subsequences in the data. And addition of class weights w (|w| = lc) are required. This stage is realized in four steps and its outcome is the classification confusion matrix.

Step 1 Sequential pattern mining

The training sequences are divided in to lc subsets each one containing all sequences belonging to the same class. Then sequential pattern mining is applied to each frequent pattern generating lc set of sequential patterns satisfying the user defined constraints. The above process is followed by the CBS_CLASS algorithm which mines the whole data set of frequent patterns. The output of this stage is the set of frequent patterns characterizing by the lc classes.

Step 2 Pattern score matrix

After the extraction of sequential pattern, for each class we create a frequent pattern score matrix FSM^j , j = 1,2,….lc (FSM matrix is created for each class. Each FSM^j includes a score that defines the implication of pattern that belongs to class j with all Sj frequent sequences, thus its size is lc x ls. This implication function is defined with the help of scoring function. If a particular frequent pattern that contain any class label, present in the sequence then the element of FSM matrix is equal to the value length (Pm)-1divided by the number of patterns in the particular class. If any pattern not contained in Si Sequence then FSM is equal to zero.

We subtract from the length of the pattern, in order to assign the minimum score is 1 to the minimal pattern. Also the score of a sequence with respect to the class is divided by the number of sequential patterns extracted from this class set. Thus the smaller the number of patterns describing the class a more significant is each one of these pattern.

Step 3 Class matrix calculation and update

With the help of FSM matrices, we derive the class score matrix (CSM), each element of this matrix is the score of the all patterns belonging to jth class for ith sequence. Since the score of specific pattern for a sequence is 0 if this sequence does not contain the pattern, only the patterns included in a sequence contribute to the class score, thus the size of the CSM matrix is lc x ls. Then each row of CSM matrix is multiplied by a parameter which denotes the weight of specific class thus the CSM matrix is updates as follows: row (j) = w(j) . row (j) where row(j) is the jth row of the CSM matrix and w(j) is the jth element of the class weight vector.

Step 4 Confusion matrix calculations

For each sequence Si, a class is predicted based on updated CSM matrix. This predicted class is defined as the class that obtains a higher score for the ith sequences: pci = arg max j=1,2,3…lc (CSM(j,i)). Based on the real class ci and predicted calss pci the confusion matrix for all sequences is calculated.

Algorithm for classification

STEP 1: Sequential pattern mining (SPM)

- For j = 1,…lc (for each class)
  Sj = {Si/ci =j}
  (select all sequences from the data that belong to the jth class)
  Pj = SPM(Sj)
  (perform SPM to these sequences extract the sequential pattern which describe the class)
- End

STEP 2 Frequent Pattern Score Matrix Calculation

- for j= 1,2….lc (for each class)
  for m = 1,,,lj (for each pattern of this class, |pj|= lj)
  for i=1,,,ls (for each sequence in data set)
  if Pm is contained in Si then FSM (m,i) = (length(pm)-1)/Pj else FSM (m,i)=0
  (if the Si sequence contains the Pm pattern i.e. mth pattern of jth class then it assigned the value of (length(pm)-1) divided by the number of patterns describing the jth class else it is assigned the value if zero)

End
End
End

End

STEP 3 Class Score Matrix Calculation and update

- for j= 1,2….lc (for each class)
  for j = 1,2,,,,ls (for each sequence in data set)
  CSM(j,i)=∑FSM (m,i)
  CSM (j,i) = wc(j).CSM(j,i)
  (The CSM value for the ith sequence for the jth class which denotes the ith sequence of the jth class the sum of all frequent pattern belonging to jth class contained in the ith sequence. Each element of column of the CSM is updated by multiplying its class weight)
  End
  End

STEP 4 Confusion Matrix Calculations

for j = 1,2,,,,ls (for each sequence in data set)
pci= arg max(CSM(j,i))
(pci is the predicted class for the i^{th} sequence defined by the row)
CM(ci, pci) = CM(ci, pci)+1

5. Results

We evaluate the scalability of our approach and with the results show its utility. All experiments were performed on Intel Pentium 1.6GHz PC with 1 GB RAM running Windows XP. All algorithms were written in visual C++ and simulate with WEKA tool.

6. Performance Study

We test our algorithms mining all frequent patterns and mining closed patterns to demonstrate scalability approach for frequent pattern extraction and also it is useful to find the reducibility of search space.

Data sets : For performance study we have two type of data sets by UCI Machine learning archive library. One dataset which is in two dimensions for performance study of Finding frequent patterns and second data set in form of sequential data has been used for sequence classification as well as finding patterns with MinFreSupport.

6.1.1 Experiment 1

We have used one of the data set of our performance evaluation with record of 2000 customers. In this data set there are only or few very short repeated patterns. We present the evaluation time of the repeated subsequence after the execution time according to repetition support MinFreSupport. As it can be seen from the figure there is a sharp decrease in the execution time of the repeated sequence phase of the algorithm when the repetition support increases.

6.1.2 Experiment 2

The Second data sets primary protein structure has been used for finding frequent patterns as well as sequence classification into folds and classes. A group of primary protein sequences were taken from the UCI machine library used to several fold of classification. The work is divided into several subgroup of experiments.

1. In the first experiment we use sequences from 12 categories class A and class B folds the training and test set consist of 426 and 331 proteins respectively.
2. In the second experiment we use sequences from 10 categories (class B folds). The training set consist of 402 proteins and test set of 190 proteins.
3. In the third experiment we use sequences from 7 categories (class B folds). The training set consist of 260 proteins and test set of 130 proteins.
4. In the fourth experiment we use sequences from 2 categories (all sequences from class A fold and all from class B folds). The training set consists of 666 proteins and test set of 336 proteins.

Since for all experiment we may give user defined support ,so we see the minimum support to 50% , meaning that a pattern should be present at least in half of the training sequences as value should be increased. Also for each of above experiment we varied the number of gaps when the gaps are increased the number of extracted patterns are quite large. Thus we create a training model based on various experiments for frequent patterns and for calculation of w. In the testing phase , the frequent patterns and w are used to classify the sequence of test set.

Exp 1 |SD train| = 426 |SD test| = 331 lc =12

Repeated Gap	*Patterns*	*Training SD accuracy*	*Test SD accuracy*
1	1568	34.125	22.5
2	3670	35.635	15.6
3	5060	52.253	22.3
4	12635	68.2	34.2
5	25000	61.450	28.1

Exp 2 |SD train| = 402 |SD test| = 190 lc =10

Repeated Gap	*Patterns*	*Training SD accuracy*	*Test SD accuracy*
1	1168	44.125	20.5
2	2240	25.635	18.16
3	5000	62.253	23.3
4	12635	78.2	39.2
5	25000	71.450	37.1

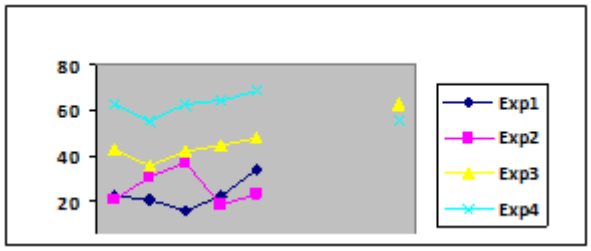
Exp 3 |SD train| = 260 |SD test| = 130 lc =7

Repeated Gap	*Patterns*	*Training SD accuracy*	*Test SD accuracy*
1	568	54.125	42.5
2	1670	59.635	35.6
3	2060	52.253	42.3
4	5065	78.2	44.2
5	10200	71.450	48.1

Exp 4 |SD train| = 666 |SD test| = 336 lc =2

Repeated Gap	*Patterns*	*Training SD accuracy*	*Test SD accuracy*
1	1568	64.125	62.5
2	3670	55.635	55.6
3	5060	62.253	62.3
4	12635	78.2	64.2
5	25000	71.450	68.1

Figure1: Graphical representation of the accuracy of the proposed methodology for the four experiments



7. Conclusion

There is huge wealth of data is available in sequential format like share market data, customer shopping patterns, DNA and protein sequences. In many of this sequential pattern, patterns and behaviors of interest repeat frequently within each sequence. To classify these kind of frequent patterns, we propose a work of classification of closed repeated gapped subsequences.

As a future work, frequent repeated gapped subsequence can be used for multi dimensional data , multivariate data and time series data.

References

[1] Ding, B., Lo, D., Han, J., & Khoo, S.-C. (2009). Efficient mining of closed repetitive gapped subsequence from a sequence database. ICDE 09.
[2] Chen, E., Cao, H., Li, Q., & Qian, T. (2008). Efficient strategies for tough aggregate constraint-based sequential pattern mining. Inf. Sci., 178(6), 1498-1518.
[3] Chen, Y.-L., & Hu, Y.-H. (2006). Constraint-based sequential pattern mining: The consideration of regency and compactness. Decis. Support Syst., 42(2), 1203-1215.
[4] Cheng, H., Yan, X., & Han, J. (2004). IncSpan: Incremental mining of sequential patterns in large database. KDD '04: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (pp. 527-532).
[5] Exarchos, T. P., Tsipouras, M. G., Papaloukas, C., & Fotiadis, D. I. (2008). A two-stage methodology for sequence classification based on sequential pattern mining and optimization. Data Knowl. Eng., 66(3), 467-487.
[6] Lin, N. P., Chen, H.-J., Hao, W.-H., Chueh, H.-E., & Chang, C.-I. (2008). Mining strong positive and negative sequential patterns. W. Trans. on Comp., 7(3), 119-124.
[7] Masseglia, F., Poncelet, P., & Teisseire, M. (2009). Efficient mining of sequential patterns with time constraints: Reducing the combinations. Expert Syst. Appl., 36(3), 2677-2690.
[8] Xing, Z., Pei, J., & Keogh, E. (2010). A brief survey on sequence classification. SIGKDD Explorations Newsletter, 12(1), 40-48.
[9] Yan, X., Han, J., & Afshar, R. (2003). CloSpan: Mining closed sequential patterns in large datasets. Proceedings of SDM, (pp. 166-177).

Author Profile

Kusum Sharma received the B.E. from Pt. Ravi Shankar Shukla University Raipur in Information Technology and now perusing M.Tech. in Computer Technology from CSVTU Bhilai (C.G.). She has published many papers in reputed national and international journal's and conferences. Her research area includes Networking, Data Warehousing and Data Mining, image processing etc.

Prof. Asha Ambhaikar received B.E from Nagpur University, Nagpur, India, later did her post graduation (M. Tech) in Information Technology. She has submitted her PhD in Computer Science and Engineering on the topic "Design and Development of MANET Routing Protocol for Improving Scalability". Currently she is working as Associate Professor & Head of the department of Computer Science and Engineering, Rungta College of Engineering and Technology, Bhilai, India. She is a chairman Board of studies of Information Technology in Chhattisgarh Swami Vivekananda Technical University, Bhilai (C.G.).She is Member of Academic Council in CSVTU Bhilai. She is also a member of Editorial Board and Reviewer of Reputed International Journals. She has published more than 40 research papers in reputed national and international journal's and conferences. Her research area includes Networking, Adhoc Networking, Data Warehousing and Data Mining, Distributed system; signal processing, image processing, information systems and security Cloud Computing etc.