Sequential Mining Classification

Carine Bou Rjeily
Nanomedicine Lab
Université de Bourgogne
Franche – Comté, UTBM
Belfort, France
carine.bourjeily@utbm.fr

Georges Badr
TICKET Lab
Antonine University
Hadat - Baabda,
Lebanon
georges.badr@ua.edu.lb

Amir Hajjam El Hassani
Nanomedicine Lab
Université de Bourgogne
Franche – Comté, UTBM
Belfort, France
amir.hajjam-el-hassani@utbm.fr

Emmanuel Andres
Université de Strasbourg
Centre Hospitalier
Universitaire
Strasbourg, France
emmanuel.andres@chru-strasbourg.fr

***Abstract*—Sequential pattern mining is a data mining technique that aims to extract and analyze frequent subsequences from sequences of events or items with time constraint. Sequence data mining was introduced in 1995 with the well-known Apriori algorithm. The algorithm studied the transactions through time, in order to extract frequent patterns from the sequences of products related to a customer. Later, this technique became useful in many applications: DNA researches, medical diagnosis and prevention, telecommunications, etc. GSP, SPAM, SPADE, PrefixSPan and other advanced algorithms followed. View the evolution of data mining techniques based on sequential data, this paper discusses the multiple extensions of Sequential Pattern mining algorithms. We classified the algorithms into Sequential Pattern mining, Sequential rule mining and Sequence prediction with their extensions. The classification is presented in a tree at the end of the paper.**

Keywords— sequential pattern mining; sequential rule mining; sequence prediction; sequence database; data mining; algorithms; classification

I. Introduction

Interesting Sequential Patterns (SP)s in a sequence database are extracted using sequential patterns mining algorithms. Other techniques including the sequence prediction and the sequential rule mining are also used nowadays for decision-making purposes. The main idea is to extract frequent subsequences, called patterns, from a massive amount of collected data and understand their relation(s). Many sectors are interested in this technique. For example, it helps in analyzing customers' purchases to improve marketing strategy: Let's say a customer buys a camera and a lens. The next time he comes, he buys a tripod. That information could be used to predict customers' needs by understanding their interests. The company may then offer a tripod or a discount when buying a camera and a lens. The first part of the paper defines important terms and notions. The second shows a survey on the most important sequential mining algorithms, in which we present the proposed classification. We conclude the article with the classification tree.

II. Important terms and definitions

Before presenting the algorithms and their classification, it is important to define some basic terms used in sequential pattern mining in order to understand the mining process.

a) An item is an entity that can have multiple attributes: date, size, color, etc.

b) I = {i1, ..., in} is a non empty set of items. A k-item set is an item set with k items.

c) A sequence "α" is an ordered list of item sets. An item set Xy in a sequence, with $1 \leq y \leq L$, is called a transaction. L denotes the length of the sequence, which refers to the number of its transactions.

d) A sequential Database (SDB) is a list of sequences with a sequence ID (SID) (cf. Table I).

e) A sequence β can have a sub-sequence α, and β a super-sequence of α.

f) A sequential rule, denoted $X \rightarrow Y$, is a relationship between two unordered item sets $X, Y \subseteq I$ where $X \cap Y = \emptyset$. $X \rightarrow Y$ means that if items of X appear in a sequence, items of Y will also occur in the same sequence.

g) The support of a rule r in a sequence database SDB is defined as supSDB(r) = $|\{s|s \in SDB \wedge r \vee s\}| / |SDB|$ = the number of sequences that contains $X \cup Y$ divided by the number of sequences in the database.

h) A rule r is a frequent sequential rule iff supSDB(r) ≥ minsup, with minsup $\in [0, 1]$ is a threshold set by the user.

i) The confidence of a rule r in a sequence database SDB is defined as confSDB(r) = $|\{s|s \in SDB \wedge r \vee s\}| / |\{s|s \in SDB \wedge X \vee s\}|$ = the number of sequences that contains $X \cup Y$, divided by the number of sequences that contains X.

j) A rule r is a valid sequential rule iff it is frequent and confSDB(r) ≥ minconf, with minconf $\in [0, 1]$ is a threshold set by the user.

k) Apriori-based [1]: Many mining algorithms are based on this technique. The main idea is to create a list of the most frequent items with respect to minsup and minconf. The list is increased progressively considering the support and the confidence.

l) Sequential rule mining is to find all frequent and valid sequential rules in a SDB [2].

m) Patten Growth [3] is a method for extracting frequent sequences by partitioning the search space and then saving the frequent item sets using a tree structure. Extraction is done by concatenating to the processed sequence (called prefix sequence) frequent items with respect to its prefix sequence. This method can be seen as depth-first traversal algorithm and eliminates the necessity to repetitively scan all the SDB.

TABLE I. A SEQUENCE DATABASE

SID	Sequence
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}>

III. SEQUENTIAL MINING ALGORITHMS

Our paper focuses on sequential mining algorithms. We are presenting a state of the art of the recent algorithms in which we elaborate a classification regarding their main objectives and principles. Thus, this classification divides the algorithms into three primary types: SP mining, sequential rule mining and sequence prediction. Each of these can be split into different criteria and strategy.

A. Sequential Patterns Mining

1) Frequent sequential patterns mining

It consists of finding subsequences appearing frequently in a set of sequences called sequential pattern or frequent subsequence. The frequency of these patterns is no less than a minimum support threshold minsup specified by the user.

a) GSP, the Generalized Sequential Patterns algorithm [4] is a Apriori-like method and was one of the first algorithms that studied SPs. The database is scanned multiple times and only candidates with minimum support or above are conserved until no new candidates are found. This technique generates an enormous number of candidate sequences and then tests each one with respect of the user-defined minsup.

b) SPADE or Sequential PAttern Discovery using Equivalence classes [5] reduces the search space by aggregating SPs into equivalent classes. It uses a vertical database where each subsequence is originally associated to its occurrence list. Thereby, two k-length sequences are in the same equivalence class if they share the same k-1 length prefix.

c) SPAM or Sequential PAttern Mining [6], is a memory-based algorithm and uses vector of bytes (bitmap representation) to study the existence (1) or absence (0) of an item in a sequence after loading the Database in the memory. Candidates are generated in a tree by an S-extension that adds an item in another transaction, and by an I-extension that appends the item in the same transaction. The candidates are verified by counting the bytes with a value of one with the defined minsup.

d) Prefix-projected Sequential PatterN mining, known as PrefixSpan [7], is a pattern-growth based algorithm that discovers SPs using the idea of projected database. The algorithm studies the prefix subsequences instead of exploring all the possible occurrences of frequent subsequences. Then, it performs a projection on their corresponding postfix subsequences. Frequent sequences will grow by mining only local frequent patterns showing the efficiency of this algorithm.

e) The LAst Position Induction algorithm, LAPIN [8] is used for the extraction of long sequences and the reduction of the search space. The algorithm assumes that the last position of an item s is helpful to decide whether this item could be appended to a frequent sequence of length k in order to get a frequent sequence of k+1 length.

f) CM-SPAM and CM-SPADE [9] are extensions of the two well know algorithms SPADE and SPAM to which is added a new structure called Co-Occurrence MAP (C-MAP). The latter is used to store co-occurrence information by dividing then into CMAPi and CMAPs sub-structures. The first stores the items that succeeds each item by i-extension and s-extension at least minsup times. The i-extension and s-extension are considered non-frequent if the i-extension or s-extension of a pattern P with an item i, if there is an item x in the pattern P that doesn't belong to CMAPs neither CMAPi.

TABLE II. COMPARING SOME FREQUENT SEQUENTIAL PATTERN EXTRACTION ALGORITHMS BASED ON [29]

Algorithm	**Condition**		**Properties**	
	Dataset size	***Min-supp size***	***Execution time (s)***	***Memory consumption (MB)***
AprioriAll	AVG	Low	Very slow	Huge
		Medium	Slow	Huge
	Large	Low	Fail	Fail
		Medium	Very slow	Huge
GSP	AVG	Low	> 3600	800
		Medium	2126	687
	Large	Low	Fail	Fail
		Medium	Fail	Fail
SPAM	AVG	Low	Fail	Fail
		Medium	136	574
	Large	Low	Fail	Fail
		Medium	674	1052
SPADE	AVG	2.5 times less efficient than SPAM		
	Large	More efficient than SPAM		
CM-SPADE	- 8 times more efficient than SPADE - Small memory overhead with large datasets			
PrefixSpan	AVG	Low	31	13
		Medium	5	10
	Large	Low	1958	525
		Medium	798	320
FreeSpan	- Less efficient than PrefixSPan - Less memory Consumption in case of « disk based projection » - Less efficiency in case of « memory based projection »			
LAPIN	AVG	Low	>3600	Fail
		Medium	7	8
	Large	Low	Fail	Fail
		Medium	201	300

Table II above presents a comparision of some frequent sequential patterns mining algorithms. The performance (i.e. execution time and memory consumption) of the algorithms is compared according to the size of datasets and the minimum support values. Note that an average dataset is about 200 KB and a large one is about 800 KB. When an algorithm fails to execute, the memory consumption cannot be measured.

2) Closed sequential patterns mining

A closed sequential pattern (CSP) is not necessary included in another pattern having the same support. The set of CSPs is much smaller than the set of SPs making mining more efficient.

a) CloSpan [10] is based on mining frequent closed sequences instead of exploring all frequent sequences and is used to mine long sequences. Its main advantage is in time and space reduction. The algorithm is divided into two stages. In the first, it generates a set of all frequent sequences and eliminates the non-closed sequences in the second.

b) BIDE+ (BI-Directional Extension) [11] is an extension of the BIDE algorithm that mines closed SPs and avoids problem of the candidate maintenance-and-test paradigm used by CloSpan. It works in a DFS manner in order to generate the frequent closed patterns and consumes less memory compared to the previous version.

c) ClaSP [12] is based on the SPADE algorithm and was the first to mine closed frequent SPs in vertical databases. ClaSP has two phases: The first one generates a subset of frequent sequences called Frequent Closed Candidates (FCC), that is kept in main memory; and the second step executes a post-pruning phase to eliminate from FCC all non-closed sequences to finally obtain exactly FCS.

d) CM-ClaSP [9] is an extension of ClaSP based on the new representation of data called C-MAP as discussed in CM-SPADE and CM-SPAM.

3) Maximal sequential patterns mining

Sequential pattern mining may return too many results making it difficult for the user to understand and analyze. Mining maximal SPs may be a solution. A Maximal SP is a pattern that is not included in another pattern.

a) MaxSP [13] is inspired by the PrefixSpan algorithm. It is based on a pattern-growth algorithm that aims to extract maximal SPs without maintaining candidates. It has an integrated BIDE-like mechanism that checks if a pattern is maximal. MaxSp reduces the redundancy in SPs that could be time consuming and requires a lot of storage space.

b) VMSP [14] (Vertical Maximal Sequence Patterns) is based on the SPAM search procedure that generates the pattern and explores candidate patterns having same prefix in a recursive manner. VMSP integrates three strategies: Efficient Filtering of Non-Maximal Patterns (EFN), Forward Maximal Extension Checking (FME) and Candidate Pruning by Co-Occurrence Map (CPC).

4) Compressing sequential patterns mining

This kind of algorithms is used to reduce redundancy and thus to minimize the size of mining results. GoKrimp and SeqKrimp [15] are two compressing SPs mining algorithms, based on the Krimp algorithm, explore directly compressing patterns and avoid the resource-consuming candidate generation. SeqKrimp utilizes a frequent closed SPs mining algorithm to generate a set of candidate patterns. Then, greedily calculates the benefits of adding extending a given pattern from the candidates. This procedure is repeated until no more useful patterns could be added. GoKrimp uses the same procedures but is an ameliorated version of SeqKrimp. A dependency test is provided to consider only related patterns to extend a given pattern. This technique aims to avoid the excessive tests of all possible extensions and makes the GoKrimp faster than SeqKrimp.

5) Top-k sequential patterns mining

In SP mining algorithm, tuning the minsup parameter to get enough patterns is a difficult and time-consuming task. To remedy to this issue, Top-k Sequential Patterns mining algorithms were implemented to return k SPs.

a) TSP (Top-K Closed Sequential Patterns) [16] uses the concept of pattern-growth and projection based SP mining of PrefixSpan algorithm, then performs a multi-pass mining to find and grow patterns. After closed pattern verification phase, the algorithm applies the minimum length constraint verification, which reduces the search space.

b) TKS (Top-K Sequential Patterns) [17] uses a vertical bitmap database representation. It adapts the SPAM search procedure to explore the search space of patterns to transform it to a top-k algorithm. Then, TSK extends the most promising patterns, meaning that it finds patterns with high support in an early stage and discards infrequent items. Finally, the algorithm uses a PMAP (Precedence MAP) data structure to prune the search space.

B. Sequential Rules Mining

1) Sequential rules

Mining frequent patterns is not sufficient in decision-making. Sequential rules mining and sequence prediction (next section) are necessary. A sequential rule indicates that if some item(s) occur in a sequence, some other item(s) are likely to occur afterward with a given confidence or probability.

a) CMDeo [18], was first designed to explore rules in a single sequence. It explores the search space in breadth-first search and extracts all valid rules of size 1*1 respecting minimum support and confidence. Similar to Apriori, CMDeo generates a huge amount of valid rules by applying a left and a right expansion.

b) RuleGrowth [19] explores sequential rules for several sequences and not only for one. It is based on the pattern-growth approach in finding the sequential rules that explores rules between two items and expands their left and right.

c) CMRules [2], an alternative of CMDeo, searches for the association rules to reduce the search space, then removes rules that do not respect minimum support and confidence.

d) ERMiner (Equivalence class based sequential Rule Miner) [20] algorithm uses a vertical representation to avoid database projection. It mines the search space through equivalence classes to generate rules with the same antecedent or consequent.

2) Top-k sequential rules mining

Specifying the number of sequential rules to be found, may overcome the difficulty in fine-tuning sequential rules parameters like minsup and minconf.

a) TopSeqRule [21] was the first to address the top-k sequential rules mining. It generates rules for several sequences based on the RuleGrowth search strategy integrated with the general process for mining top-k patterns. To optimize results, it first generates the most promising rules and reduces the search space by increasing minsup.

b) TNS [22] is used to discover the top-k non-redundant sequential rules. It adopts the TopSeqRule to mine the top-k rules and adapts it to eliminate redundancy.

3) Sequential rules with window size constraints

a) TRuleGrowth [23] is an extension of the RuleGrowth with a sliding window constraint. It is very useful in the discovery of temporal patterns (patterns that happen within a maximum time interval).

C. Sequence Prediction

In many applications, it is very important to predict the next element in a sequence. Given a set of sequences, the objective is to predict the next element in a sequence S.

a) CPT (Compact Prediction Tree) [24] is a lossless prediction model that uses all information in the sequence for prediction. It consists in two phases training and prediction. The first compresses the sequences in a prediction tree. A given sequence S is predicted by finding all sequences that contains the last x items from S in any order and in any position. CPT is more efficient than other existent algorithm PPM (Prediction by Partial Matching) [25], DG (Dependency Graph) [26] and All-K-th-Order Markov [27].

b) CPT+ [28] is an ameliorated version of CPT where Frequent Subsequence Compression (FSC), Simple Branch compression (SBC) and Prediction with improved Noise Reduction (PNR) strategies were added to improve prediction time and precision.

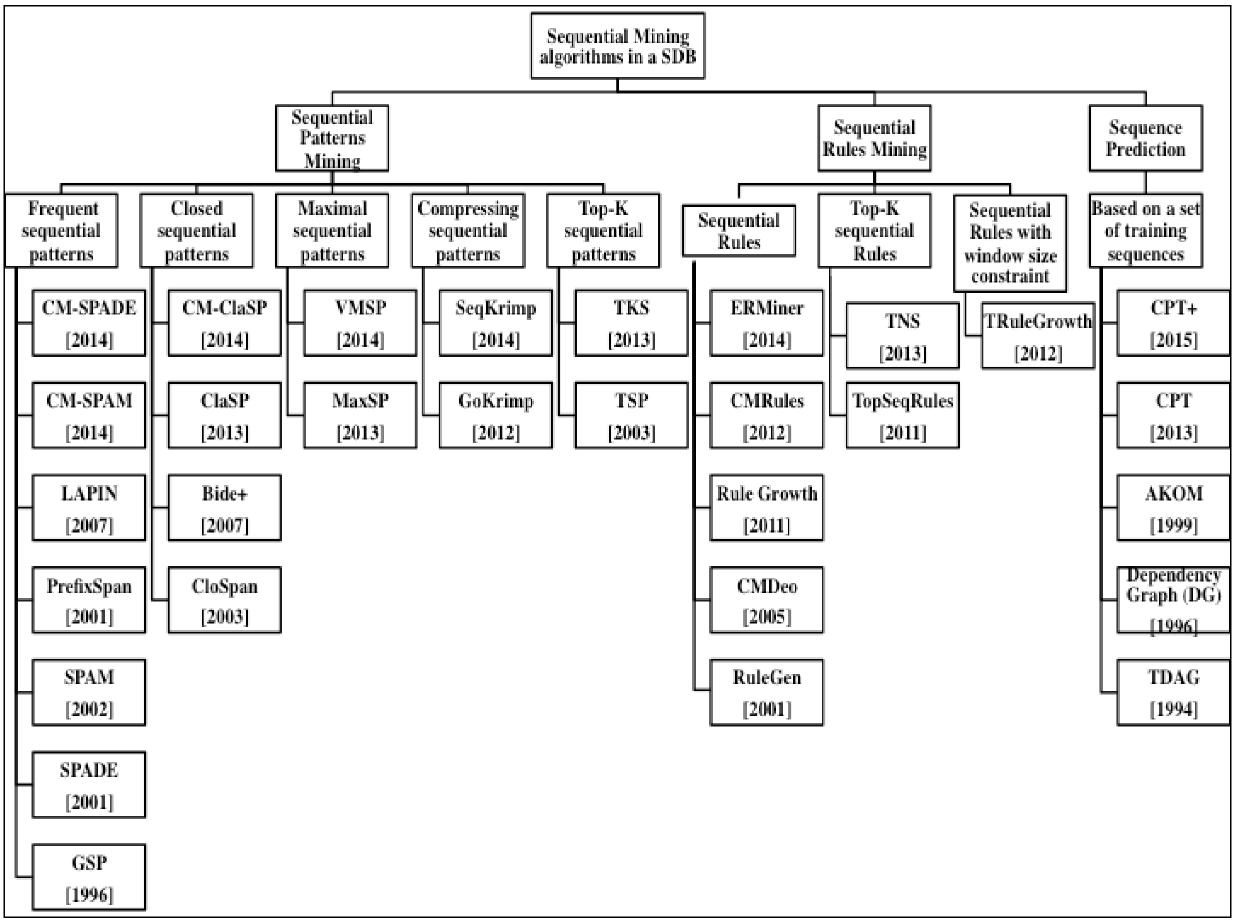

Fig. 1. Sequential mining algorithm classification

IV. Conclusion

In this paper, we presented a classification for Sequential-mining algorithms based on three main axes: frequent sequential patterns mining, sequential rules mining and sequence prediction. We first introduced some important terms in the data-mining domain. A short definition of each axis was presented. Then, we investigated the most important and recent algorithms in each axis with a brief description about their methods and implementations. Our main contribution is expressed by the diagram shown in Fig.1. This diagram consists of a classification tree containing the most recent algorithms and their extensions. This tree can help researchers choosing the appropriate algorithm according to their needs especially when it comes to sequential patterns mining.

References

[1] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases", In ACM sigmod record(Vol. 22, No. 2, pp. 207-216). ACM, 1993.

[2] P. Fournier-Viger, U. Faghihi, R. Nkambou, E. Mephu Nguifo, "CMRules: Mining Sequential Rules Common to Several Sequences. Knowledge-based Systems", Elsevier, 25(1): 63-76, 2012.

[3] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: A frequent-pattern tree approach", Data mining and knowledge discovery, 8(1), pp.53-87, 2000.

[4] R. Srikant, and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements", In International Conference on Extending Database Technology (pp.1-17). Springer Berlin Heidelberg, 1996.

[5] M.J. Zaki, "SPADE: An efficient algorithm for mining frequent sequences", Machine learning, 42(1-2), pp.31-60, 2001.

[6] J. Ayres, J. Flannick, J. Gehrke, J. and T. Yiu, "Sequential pattern mining using a bitmap representation", In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining(pp. 429-435). ACM, 2002.

[7] J. Han, J. Pei, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M.C. Hsu, "Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth", In proceedings of the 17th international conference on data engineering, pp. 215-224, 2001.

[8] Z. Yang, Y. Wang, and M. Kitsuregawa, M., "LAPIN: effective sequential pattern mining algorithms by last position induction for dense databases", In International Conference on Database systems for advanced applications (pp. 1020-1023). Springer Berlin Heidelberg, 2007.

[9] P. Fournier-Viger, A. Gomariz, M. Campos, and R. Thomas, "Fast vertical mining of sequential patterns using co-occurrence information", In Pacific-Asia Conference on Knowledge Discovery and Data Mining (pp. 40-52). Springer International Publishing, 2014.

[10] X. Yan, J. Han, R. Afshar R., "CloSpan: Mining Closed Sequential Patterns in Large Datasets", Proceedings of the 2003 SIAM International Conference on Data Mining, 2003.

[11] J. Wang, and J. Han, "BIDE: Efficient mining of frequent closed sequences", In Data Engineering, 2004. Proceedings. 20th International Conference on (pp. 79-90). IEEE, 2004.

[12] A. Gomariz, M. Campos, R. Marin, and B. Goethals, "Clasp: An efficient algorithm for mining frequent closed sequences" In Pacific-Asia Conference on Knowledge Discovery and Data Mining (pp. 50-61). Springer Berlin Heidelberg, 2013.

[13] P. Fournier-Viger, C.W. Wu, and V.S. Tseng, "Mining maximal sequential patterns without candidate maintenance", In International Conference on Advanced Data Mining and Applications (pp. 169-180). Springer Berlin Heidelberg, 2013.

[14] P. Fournier-Viger, C.W. Wu, A. Gomariz, and V.S. Tseng, "VMSP: Efficient vertical mining of maximal sequential patterns", In Canadian Conference on Artificial Intelligence (pp. 83-94). Springer International Publishing, 2014.

[15] H.T. Lam, F. Mörchen, D. Fradkin, and T. Calders, "Mining compressing sequential patterns", Statistical Analysis and Data Mining, 7(1), pp.34-52, 2014.

[16] P. Tzvetkov, X. Yan, and J. Han, "TSP: Mining Top-k Closed Sequential Patterns", Knowledge and Information Systems, vol. 7, no. 4, pp. 438-457, 2005.

[17] P. Fournier-Viger, A. Gomariz, T. Gueniche, E. Mwamikazi, and R. Thomas, "TKS: efficient mining of top-k sequential patterns", In International Conference on Advanced Data Mining and Applications (pp. 109-120). Springer Berlin Heidelberg, 2013.

[18] J. Deogun, and L. Jiang, "Prediction mining–an approach to mining association rules for prediction", In International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing (pp. 98-108). Springer Berlin Heidelberg, 2005.

[19] P. Fournier-Viger, R. Nkambou, and V.S.M. Tseng, "RuleGrowth: mining sequential rules common to several sequences by pattern-growth", In Proceedings of the 2011 ACM symposium on applied computing (pp. 956-961), 2011.

[20] P. Fournier-Viger, T. Gueniche, S. Zida, and V.S. Tseng, "ERMiner: sequential rule mining using equivalence classes", In International Symposium on Intelligent Data Analysis (pp. 108-119). Springer International Publishing, 2014.

[21] P. Fournier-Viger, and V.S. Tseng, "Mining top-k sequential rules", In International Conference on Advanced Data Mining and Applications (pp. 180-194). Springer Berlin Heidelberg, 2011.

[22] P. Fournier-Viger, and V.S Tseng, "TNS: mining top-k non-redundant sequential rules", In Proceedings of the 28th Annual ACM Symposium on Applied Computing, 2013.

[23] P. Fournier-Viger, C.W. Wu, V.S. Tseng, and R. Nkambou, "Mining sequential rules common to several sequences with the window size constraint", In Canadian Conference on Artificial Intelligence (pp. 299-304). Springer Berlin Heidelberg, 2012.

[24] T. Gueniche, P. Fournier-Viger, and V.S. Tseng, "Compact prediction tree: A lossless model for accurate sequence prediction", In International Conference on Advanced Data Mining and Applications (pp. 177-188). Springer Berlin Heidelberg, 2013.

[25] J. Cleary, I. Witten, "Data compression using adaptive coding and partial string matching", IEEE Trans. on Inform. Theory, vol. 24, no. 4, pp. 413-421, 1984.

[26] V.N, Padmanabhan, J.C. Mogul, "Using Prefetching to Improve World Wide Web Latency", Computer Communications, vol. 16, pp. 358-368, 1998.

[27] J. Pitkow, P. Pirolli, "Mining longest repeating subsequence to predict world wide web surfing", In: USENIX Symposium on Internet Technologies and Systems, Boulder, CO, pp. 13-25, 1999.

[28] T. Gueniche, P. Fournier-Viger, R. Raman, and V.S. Tseng, "CPT+: Decreasing the time/space complexity of the Compact Prediction Tree", In Pacific-Asia Conference on Knowledge Discovery and Data Mining (pp. 625-636). Springer International Publishing, 2015.

[29] N.R. Mabroukeh, and C.I. Ezeife, "A taxonomy of sequential pattern mining algorithms". *ACM Computing Surveys (CSUR)*, *43*(1), p.3, 2010.