Sequential Pattern Sampling with Norm Constraints

Lamine Diop*, Cheikh Talibouya Diop*, Arnaud Giacometti†, Dominique Li† and Arnaud Soulet†
*University Gaston-Berger of Saint-Louis, Senegal, {diop.lamine3;cheikh-talibouya.diop}@ugb.edu.sn
†University of Tours, France, firstname.lastname@univ-tours.fr

***Abstract*—In recent years, the field of pattern mining has shifted to user-centered methods. In such a context, it is necessary to have a tight coupling between the system and the user where mining techniques provide results at any time or within a short response time of only few seconds. Pattern sampling is a non-exhaustive method for instantly discovering relevant patterns that ensures a good interactivity while providing strong statistical guarantees due to its random nature. Curiously, such an approach investigated for itemsets and subgraphs has not yet been applied to sequential patterns, which are useful for a wide range of mining tasks and application fields. In this paper, we propose the first method for sequential pattern sampling. In addition to address sequential data, the originality of our approach is to introduce a constraint on the norm to control the length of the drawn patterns and to avoid the pitfall of the "long tail" where the rarest patterns flood the user. We propose a new constrained two-step random procedure, named CSSAMPLING, that randomly draws sequential patterns according to frequency with an interval constraint on the norm. We demonstrate that this method performs an exact sampling. Moreover, despite the use of rejection sampling, the experimental study shows that CSSAMPLING remains efficient and the constraint helps to draw general patterns of the "head". We also illustrate how to benefit from these sampled patterns to instantly build an associative classifier dedicated to sequences. This classification approach rivals state of the art proposals showing the interest of constrained sequential pattern sampling.**

***Keywords*—Pattern Mining, Pattern Sampling, Sequential Data**

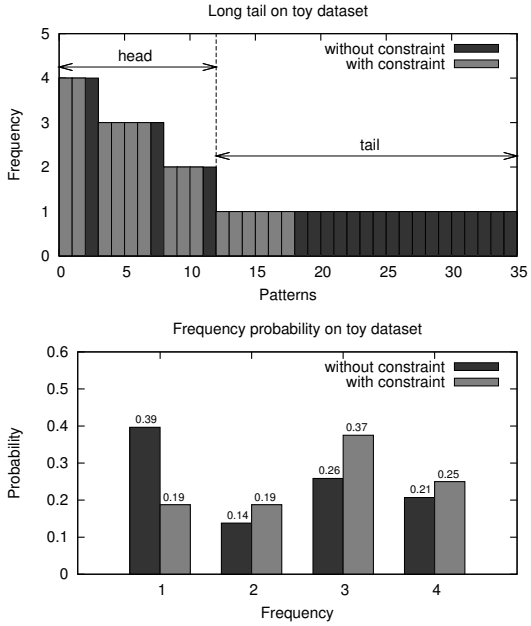

Fig. 1. Impact of the long tail on frequent sequential pattern sampling

I. INTRODUCTION

In recent years, the field of pattern mining has shifted to user-centered methods [1]. Typically, the idea is to be able to capture the feedback of the user during the analysis of the first mined patterns to better choose the next ones. To guarantee this tight coupling between the system and the user, it is then necessary to use techniques that provide results at any time [2] or within a short response time of only few seconds. Pattern sampling is an efficient approach that instantly returns patterns [3], [4], [5], which enables to produce pattern-based models at any time [6]. Introduced in [7], pattern sampling returns a small set of patterns randomly drawn with a probability proportional to an interestingness measure specified by the user. For instance, with frequency, a pattern twice as frequent will be twice as likely to be picked. Sampling methods are particularly efficient and have the advantage of returning patterns with high diversity. To the best of our knowledge, there is no work addressing pattern sampling in sequential data [8]. Yet sequential pattern mining is useful for a wide range of mining tasks and application fields [9] such as web usage mining, text mining, fraud detection and so on.

Unfortunately, a naive pattern sampling according to frequency is not relevant for sequential data because of the pitfall of the long tail. In statistics and business, the long tail of a distribution is its portion having a large number of occurrences far from the central part of the distribution [10]. In our context, the long tail designates the long and rare sequential patterns far more numerous than the short and frequent ones (the "head"). As a result, it is nearly impossible to draw the most general patterns despite the bias of the frequency. This problem is stronger with sequential data than with transactional data because the number of sub-patterns in a sequence is much higher than that in an itemset of the same length. Figure 1 illustrates the long tail problem on the toy dataset provided in Section III. The top histogram shows the frequency of the 35 patterns of the toy dataset (i.e., bars in dark and light grays). We observe that 23 patterns have a frequency of 1 (the tail). Consequently, the bars in dark gray of the bottom histogram show that 39.6% of the patterns drawn according to frequency belong to this tail (with a frequency of only 1). The real-world datasets reveal even much more problematic situations (see the experimental study in Section V). For instance, each of the 10,000 patterns drawn randomly according to frequency on `bms` dataset appears only in a single sequence of the dataset. Of course, these patterns are useless because they correspond more to noise than true patterns describing the data.

To circumvent the pitfall of the long tail, we propose to sample patterns under a constraint on the maximum norm (maximum number of items). This constraint will prevent drawing too specific patterns because too long, but interest-

ingly, still allow to draw non-frequent patterns that describe sequences of rare events. It is really crucial not to force a minimal frequency in order to have a description of rare objects [6]. In Figure 1, a maximum norm constraint of 2 removes all dark gray patterns. Interestingly, much of the tail is cut off. As a result, the bottom histogram shows a significant increase in the probability to draw patterns having frequencies ranging from 2 to 4. Indeed, the probability to draw a pattern with a frequency of 1 has been divided by 2 (the first bar in light gray). To achieve this goal, we would like to use the two-step random procedure [11] which is the most efficient pattern sampling approach in the literature. After a preprocessing phase, this method extracts an exact sample of patterns without rejection. However, extending this approach to sequential patterns is a challenging problem. Indeed, its core requires counting the number of distinct subsequences for each sequence. This task is not easy because a sequence may contain several occurrences of the same subsequence and we want to consider only subsequences of a certain length.

The main contributions of the paper are as follows:

- We propose a new algorithm named CSSAMPLING (Constrained Subsequence Sampling) that samples sequential patterns proportionally to frequency with an interval constraint on the norm. It relies on a constrained two-step random procedure that requires solving two sub-problems: (i) counting the number of distinct subsequences having a maximum norm and (ii) uniformly drawing subsequences. We demonstrate that CSSAMPLING performs an exact sequential pattern sampling according to frequency, and we analyze its complexity on average.
- We present a large set of experimental results for analyzing the behavior of CSSAMPLING. We show on several datasets that our approach is efficient enough to return hundreds of sequential patterns per second. We also highlight the practical interest of norm constraints to better control the quality of the returned patterns and avoid the curse of the long tail.
- Sequence classification is a crucial data mining task useful in a wide range of applications. We investigate how sequential pattern sampling lead to build associative classifiers for sequences. Interestingly, the accuracy of these sample-based classifiers built in a short response time is comparable to that of the methods of the state of the art. Experiments show that it is again essential to use a constraint to draw general patterns contained in the head, and not in the tail.

The outline of this paper is as follows. Section II reviews some related work about pattern sampling methods. Section III introduces basic definitions and the formal problem statement. We present our constrained two-step random procedure for sequential pattern sampling in Section IV. We evaluate our approach in Section V and conclude in Section VI.

II. RELATED WORK

a) Instant discovery of sequential patterns: Sequential pattern mining has been introduced by [8] two decades ago and its usefulness has been widely proved as mentioned in introduction. Since 1995, many methods have optimized the mining of sequential patterns [12], [13], [14] and have introduced variants with constraints [15], [16] or condensed representations [17], [18]. Despite all these advances, sequential pattern mining remains a costly task that often generates too many redundant patterns. Consequently, it is not possible to discover patterns or to build pattern-based models in a short response time. This limit, also reached by other language (e.g., itemset), was circumvented by Monte Carlo tree search [19] or pattern sampling [7]. This kind of instantaneous methods is at the core of many approaches that makes data mining more interactive [3], [4], [5], [6]. But to the best of our knowledge, all these methods have not been applied to sequential patterns. The rest of the related work is devoted to the pattern sampling techniques, which corresponds to our proposal.

b) Output space sampling: Importantly, it is necessary to distinguish between input and output space sampling. The *input space* sampling [20] consists in generating from a sample of data all the patterns that would have been mined from the complete dataset. The *output space* sampling [7] consists in generating a sample of patterns among the patterns that would have been mined from the complete dataset. More formally, pattern sampling [7], [11] aims at accessing the pattern space $\mathcal{L}$ by an efficient sampling procedure simulating a distribution $\pi : \mathcal{L} \rightarrow [0, 1]$ that is defined with respect to some interestingness measure f, i.e., $\pi(.) = f(.)/Z$ where Z is a normalizing constant. As the pattern language is fully addressed proportionally to f, this approach guarantees a good variety of patterns returned to the user unlike heuristic approaches. Several approaches have been proposed for input space sampling of sequential patterns [21], [22], but to the best of our knowledge, this paper proposes the first approach to output space sampling of sequential patterns. Since the complexity of pattern sampling is independent of the language size, it is suitable for structured languages where there is a combinatorial explosion of the number of patterns like subgraphs [23] and even for infinite languages like numerical data [24]. Note that in this paper, we restrict ourselves to frequency as interestingness measure f because we focus more on sequence-specific and constraint-specific issues. It would be natural to extend our approach to other measures (e.g., area or discriminative measures) as done in [11].

c) Pattern sampling techniques: Several procedures have been proposed for the output space sampling of patterns. The first kind of procedure [23], [25] randomly draws a pattern from the search space using a heuristic to favor the patterns that are most relevant according to the interestingness measure f. In practice, these methods return interesting patterns but they offer no guarantee on the quality of the outputted sample. The second kind of procedure [7], [3], [26] is based on Markov chain Monte Carlo algorithms. The idea is that the equilibrium distribution of a random walk corresponds to the desired probability distribution. The limit of such stochastic methods is the convergence speed, which may be slow. The third kind of procedure [11], [24], [27] consists in drawing an instance of the dataset and then drawing a pattern contained in this instance. By judiciously selecting the two draw distributions, it is possible to obtain an exact sampling according to the desired final distribution. Recently, [24] adds a third step for taking into account numeric data where the pattern language is infinite. We opted for such a multi-step random procedure

for its speed and accuracy. Section IV-A underlines specific challenges for achieving this goal in the case of sequences.

Besides the inherent difficulty of addressing sequences rather than itemsets, we also add an interval constraint on the norm of the returned patterns. In the litterature, there are few proposals adding a binary predicate to restrict the sampling. [25] proposes a framework for sampling of *maximal* itemsets from transactional datasets, but it relies on a heuristic random walk with no guarantee. Based on the SAT framework, [28] requires to have a solver integrating efficiently XOR constraints and in practice, it has been implemented only for itemsets. In addition, the authors emphasize that the efficiency of this generic approach will hardly compete with approaches dedicated to a single language and/or class of constraints. In this paper, we propose an efficient method for integrating only constraints on the norm.

III. Problem Statement

This section formalizes the problem of sequential pattern sampling under norm constraints. Before, we recall some preliminary definitions about sequences.

A. Basic definitions

Let $\mathcal{I}$ be a finite set of literals called *items*. An *itemset* X is a subset of $\mathcal{I}$. A *sequence* $s = \langle X_1 \dots X_n \rangle$ defined over $\mathcal{I}$ is an ordered list of non-empty itemsets $X_i \subseteq \mathcal{I}$ ($1 \leq i \leq n$, $n \in \mathbb{N}$). n is the *size* of the sequence s denoted by $|s|$. The *norm* of the sequence s, denoted by $\|s\|$, is the sum of the cardinality of all its itemsets, i.e. $\|s\| = \sum_{i=1}^{n} |X_i|$. In the following, s^l denotes the prefix $\langle X_1 X_2 \dots X_l \rangle$ of s ($0 \leq l \leq n$, $l \in \mathbb{N}$), s^0 being the empty sequence (represented by $\langle\rangle$) and $s[j] = X_j$ denotes the j-th itemset of s ($1 \leq j \leq n$, $j \in \mathbb{N}$). Finally, we denote $\mathbb{S}$ the universal set of all the sequences defined over $\mathcal{I}$, and a sequential dataset $\mathcal{S}$ over $\mathcal{I}$ is a multi-set of sequences defined over $\mathcal{I}$. We recall the definitions of *subsequences* and of *occurrences* of a subsequence:

Definition 1 (Subsequence): A sequence $s' = \langle X'_1 \dots X'_m \rangle$ is a subsequence of a sequence $s = \langle X_1 \dots X_n \rangle$, denoted by $s' \sqsubseteq s$, if there exists an index sequence $1 \leq i_1 < i_2 < \dots < i_m \leq n$ such that for all $j \in [1..m]$, one has $X'_j \subseteq X_{i_j}$. We denote $\phi(s)$ the set of subsequences of a sequence s, i.e. $\phi(s) = \{s' \in \mathbb{S} : s' \sqsubseteq s\}$, and $\Phi(s)$ its cardinality, i.e. $\Phi(s) = |\phi(s)|$.

Example 1: We use the sequential dataset $\mathcal{S}$ presented in Table I as a running example. This dataset contains 4 sequences s_1, s_2, s_3 and s_4 defined over the set of items $\mathcal{I} = \{a, b, c, d\}$. For example, the size of $s_1 = \langle (ab)c \rangle$ is equal to 2, i.e. $|s_1| = 2$, whereas its norm is equal to 3, i.e. $\|s_1\| = 2 + 1 = 3$. Moreover, we have $s_1^0 = \langle\rangle$, $s_1^1 = \langle (ab) \rangle$, $s_1^2 = s_1$, $s_1[1] = (ab)$ and $s_1[2] = c$. Finally, the set $\phi(s_1)$ of subsequences of s_1 is defined by $\phi(s_1) = \{\langle\rangle, \langle a \rangle, \langle b \rangle, \langle c \rangle, \langle (ab) \rangle, \langle ac \rangle, \langle bc \rangle, \langle (ab)c \rangle\}$. Thus, we have $\Phi(s_1) = 1 + 3 + 3 + 1 = 8$. The number of subsequences $\Phi(s_i)$ of all sequences $s_i \in \mathcal{S}$ is detailed in Table I. The notation $\Phi_{[m,M]}(s_i)$ is formally defined in the Section IV-A. Intuitively, it represents the number of subsequences of a sequence s_i whose norm is between m and M.

It is important to note that a subsequence $s' = \langle X'_1 \dots X'_m \rangle$ may occur several times in a sequence $s = \langle X_1 \dots X_n \rangle$ if there exist several index sequences $1 \leq i_1 < i_2 < \dots < i_m \leq n$ such that for all $j \in [1..m]$, one has $X'_j \subseteq X_{i_j}$. In that case, there are several *occurrences* of the subsequence s' in s. The next definition explains how each occurrence is represented:

TABLE I. A sequential dataset $\mathcal{S}$ with 4 sequences

Sid	Sequence of itemsets	#occurrences	$\Phi(s_i)$	$\Phi_{[1,2]}(s_i)$
s_1	$\langle (ab)c \rangle$	8	8	6
s_2	$\langle (ab)c(ac) \rangle$	32	25	12
s_3	$\langle c(ac) \rangle$	8	7	5
s_4	$\langle (ab)(cd) \rangle$	16	16	10

Definition 2 (Occurrence): An ordered list of n itemsets $o = \langle Z_1 \dots Z_n \rangle$ is an occurrence of a subsequence $s' = \langle X'_1 \dots X'_m \rangle$ in a sequence $s = \langle X_1 \dots X_n \rangle$ if there exists an index sequence $1 \leq i_1 < \dots < i_m \leq n$ such that for all $j \in \{i_1, \dots, i_m\}$, one has $Z_{i_j} = X'_j \subseteq X_{i_j}$, and for all $j \in [1..n] \setminus \{i_1, \dots, i_m\}$, one has $Z_j = \emptyset$. This index sequence, called signature of o, is unique by definition.

Example 2: For the sequence $s_2 = \langle (ab)c(ac) \rangle$, $o_1 = \langle (a)(c)\emptyset \rangle$ and $o_2 = \langle (a)\emptyset(c) \rangle$ are two occurrences of its subsequence $s'_2 = \langle (a)(c) \rangle$. Moreover, the index sequences $\langle 1, 2 \rangle$ and $\langle 1, 3 \rangle$ are the signatures of o_1 and o_2, respectively. In Table I, the number of occurrences of all its subsequences is given for each sequence (e.g., there are 32 occurrences for 25 distinct subsequences in s_2).

B. Problem of sequential pattern sampling under constraint

A pattern sampling method aims at randomly drawing a pattern X from a language $\mathcal{L}$ according to an interestingness measure f. $X \sim \pi(\mathcal{L})$ denotes such a pattern where $\pi(.) = f(.)/Z$ is a probability dristribution over $\mathcal{L}$. In our case, we focus on the frequency which is an intuitive interestingness measure for experts and is an essential atomic element to build many other interestingness measures (like area or discriminative measures):

Definition 3 (Frequency): The frequency of a subsequence $s \in \mathbb{S}$ in the sequential dataset $\mathcal{S}$, denoted by $freq(s, \mathcal{S})$, is defined by: $freq(s, \mathcal{S}) = |\{s' \in \mathcal{S} : s \sqsubseteq s'\}|$.

Our goal is to randomly draw sequential patterns according to frequency under norm constraints. Given two integers m and M such that $m \leq M$, we denote $\mathbb{S}_{[m,M]}$ the set of sequences of $\mathbb{S}$ whose norm is between m and M, i.e. $\mathbb{S}_{[m,M]} = \{s \in \mathbb{S} : m \leq \|s\| \leq M\}$. The problem can finally be stated as follows:

Given a sequential dataset $\mathcal{S}$, two integers m and M, we aim at randomly drawing a subsequence $s \in \mathbb{S}_{[m,M]}$ with a probability distribution $P(s)$ proportional to its frequency in $\mathcal{S}$ i.e., $P(s) = \frac{freq(s,\mathcal{S})}{\sum_{s' \in \mathbb{S}_{[m,M]}} freq(s',\mathcal{S})}$.

One of the advantages of frequent pattern sampling [11] is to remove the minimum frequency threshold (always difficult to set) while our problem introduces two thresholds: m and M. Nevertheless, they are easier to set because their range is much smaller ([1..10] in our experiments) than that of the minimum threshold of frequency.

Example 3: Table II represents the set of all subsequences of sequences in $\mathcal{S}$ with a norm between $m = 1$ and $M = 2$, and gives the frequencies in $\mathcal{S}$ of all these subsequences.

TABLE II. SUBSEQUENCES IN $\mathbb{S}_{[1,2]}$ OF SEQUENCES IN $\mathcal{S}$

Pattern s	$freq(s,\mathcal{S})$	**Pattern** s	$freq(s,\mathcal{S})$
$\langle a\rangle$	4	$\langle ac\rangle$	3
$\langle b\rangle$	3	$\langle ad\rangle$	1
$\langle c\rangle$	4	$\langle ba\rangle$	1
$\langle d\rangle$	1	$\langle bc\rangle$	3
$\langle (ab)\rangle$	3	$\langle bd\rangle$	1
$\langle (ac)\rangle$	2	$\langle ca\rangle$	2
$\langle (cd)\rangle$	1	$\langle cc\rangle$	2
$\langle aa\rangle$	1		

For instance, because our problem is to draw a subsequence proportionally to its frequency, and $freq(\langle ac\rangle,\mathcal{S}) = 3 \times freq(\langle ba\rangle,\mathcal{S})$, our objective is to develop a sampling method such that the probability to draw the subsequence $\langle ac\rangle$ is three times greater than the probability to draw the subsequence $\langle ba\rangle$. But, even if the subsequence $\langle (ab)c\rangle$ has a frequency of 3, it will not be drawn because its norm is 3 ($> M$).

IV. CONSTRAINED TWO-STEP RANDOM PROCEDURE

A. Overview of the algorithm

To address the problem stated in the previous section, we propose to benefit from a two-step random procedure as done in [11] for sampling itemsets proportionally to their support. But, we constrain this random procedure to consider only the patterns whose norm is satisfactory at both step.

Given a dataset $\mathcal{S}$ and two integers m and M such that $m \leq M$, CSSAMPLING (Constrained Subsequence Sampling) returns a sequential pattern having a norm between m and M:

a) Step 1: Sampling a sequence: In the first step (lines 1 and 2 of Algorithm 1), we start by counting for each sequence $s \in \mathcal{S}$ the number of subsequences having a norm between m and M, i.e. $\Phi_{[m,M]}(s) = |\phi_{[m,M]}(s)|$ where $\phi_{[m,M]}(s) = \{s' \sqsubseteq s : m \leq \|s'\| \leq M\}$. To do this, we show in Section IV-B how to extend the formula given in [29]. Then, this first step continues with the drawing of a sequence s from $\mathcal{S}$ proportionally to its weight $w(s) = \Phi_{[m,M]}(s)$. For instance, Table I provides the weight $\Phi_{[1,2]}(s_i)$ of each sequence s_i. It is clear that this weight is different from the number of occurrences $2^{\|s_i\|}$ or the number of distinct subsequences $\Phi(s_i)$ and shows the importance of this calculation so as not to bias the drawing of the subsequence.

b) Step 2: Sampling a subsequence: In the second step, we randomly draw the norm k of the subsequence of s which will be returned (line 3 of Algorithm 1). This number k is randomly drawn with a probability proportional to the number of subsequences in s having exactly k as norm, i.e. according to the probability distribution $P_{[m,M]}$ defined for all $k \in [m..M]$ by: $P_{[m,M]}(k) = \frac{\Phi_{[k,k]}(s)}{\Phi_{[m,M]}(s)}$. Finally, Algorithm 1 returns at line 4 a subsequence s' in s of norm k according to a uniform distribution, meaning that each subsequence s' from s of norm k will be drawn with the same probability $\frac{1}{\Phi_{[k,k]}(s)}$. We show in Section IV-C how to perform such a uniform drawing thanks to a rejection sampling. The main challenge is to avoid to pick more often subsequences that have multiple occurrences within the sequence s. Typically, even if $\langle (a)(c)\rangle$ has two occurrences in s_2, its drawing probability must be the same as that of $\langle (a)(a)\rangle$ (that appears once within s_2).

Note that the theoretical study of these two steps (soundness and complexity) will be done in Section IV-D.

Algorithm 1 CSSAMPLING

Input: A sequential dataset $\mathcal{S}$, and two integers m and M such that $m \leq M$

Output: A sequence $s \in \mathbb{S}_{[m,M]}$ randomly drawn, i.e. $s \sim freq(\mathbb{S}_{[m,M]},\mathcal{S})$

// Step 1: Sampling a sequence

1: Compute for all $s \in \mathcal{S}$, a weight w defined by $w(s) = \Phi_{[m,M]}(s)$

2: Draw a sequence s from $\mathcal{S}$ proportionally to w: $s \sim w(\mathcal{S})$

// Step 2: Sampling a subsequence

3: Draw an integer k from m to M according to the distribution $P_{[m,M]}(k)$

4: **return** A subsequence s' of norm k randomly drawn from s: $s' \sim u(\phi_{[k,k]}(s))$ where u is the uniform distribution

B. Subsequence counting for drawing a sequence

In this section, we show how to compute the number of distinct subsequences of a sequence with an interval constraint on the norm. We benefit from [29] where a formula counts the number of distinct subsequences in a sequence *without* constraint on the norm. The main difficulty is to avoid to count the same subsequence several times, even if it has several occurrences within the sequence.

To compute the number of distinct subsequences having a norm less than or equal to j contained in a sequence $s = \langle X_1 \dots X_n\rangle$, we start with the empty sequence and then, we concatenate all itemsets X_i one by one. $s \circ Y$ denotes the concatenation of s and Y: $s \circ Y = \langle X_1 \dots X_n Y\rangle$. For each new itemset Y concatenated to s, we count only subsequences which have a norm less than j and which have not already occurred previously in s. For instance, if we add the itemset ac to $\langle (ab)c\rangle$ to count the number of subsequences having a norm less than 2 in $\langle (ab)c(\mathbf{ac})\rangle$, then we avoid counting $\langle (ab)\mathbf{a}\rangle$ whose norm (i.e., 3) is too large and we avoid counting $\langle (a)\mathbf{c}\rangle$ which has already been counted previously (for $\langle (ab)\rangle \circ c$). It is easy to see that the duplicates (here, only $\langle (a)c\rangle$) result from previous occurrences of items in (ac) within sequences $\langle (ab)c\rangle$ (here, c occurs previously at position 2). For this reason, we need the notion of position set:

Definition 4 (Position set [29]): Let s be a sequence and Y be an itemset. $L(s,Y) = \{i \in \mathbb{N} : i \leq |s| \wedge s[i] \cap Y \neq 0 \wedge (\forall j > i)(s[i] \cap Y \not\subseteq s[j] \cap Y)\}$ is the position set where Y has a maximal intersection with the different itemsets of s.

Example 4: Let $s = \langle (ab)c(ac)\rangle$ be a sequence. We have $s^1 = \langle (ab)\rangle$, $s[2] = (c)$ and $L(s^1, s[2]) = \emptyset$ because $s[2]$ intersects no itemset of s^1. Now, we are going to compute $L(s^2, s[3])$. $s[3] = (ac)$ intersects at the same time the first itemset $s[1] = (ab)$ of s ($s[1] \cap s[3] = (a)$) and the second itemset $s[2] = (c)$ of s ($s[2] \cap s[3] = (c)$). As these two intersections are disjoint, we obtain $L(s^2, s[3]) = \{1, 2\}$. This means that by concatenating subsets of $s[3]$ to the subsequences in s^2, some subsequences of s^2 might been counted twice as items of $s[3]$ are also present at positions 1 and 2 in s^2.

Using the notion of position set and the inclusion-exclusion principle, we propose a new recursive formula to count the number of distinct subsequences in a sequence s considering a maximum norm as constraint. Intuitively, to construct a

subsequence of $s \circ Y$ having a norm less than j, we can concatenate any subset of size k of Y to a subsequence of s having a norm less than $j-k$. Indeed, we are sure to obtain a subsequence of $s \circ Y$ having a norm less than $k+(j-k)=j$, and this principle is repeated for any possible size of a subset of Y. Thus, we have: $\phi_{\leq j}(s \circ Y) = \cup_{k=0}^{j} \phi_{\leq j-k}(s) \circ \mathcal{P}_{=k}(Y)$ where $\mathcal{P}_{=k}(Y) = \{X \subseteq Y : |X| = k\}$, which explains the first term of the formula given by Theorem 1. The difficulty is that a subsequence obtained by the concatenation of a subset of Y to a subsequence of s may also occur in $\phi_{\leq j}(s)$. Therefore, we have to take into account these possible redundancies to count the exact number of distinct subsequences of s with a norm less than j. This remark explains the correction term $R_{\leq j}(s,Y)$ of the formula given by Theorem 1:

Theorem 1 (Subsequence number with a maximum norm): Let s be a sequence, Y be an itemset and j be an integer, the number of distinct subsequences having a norm less or equal to j in $s \circ Y$, denoted by $\Phi_{\leq j}(s \circ Y)$, is defined as follows[1]:

$$\Phi_{\leq j}(s \circ Y) = \left(\sum_{k=0}^{j} \Phi_{\leq j-k}(s) \times \binom{|Y|}{k} \right) - R_{\leq j}(s,Y)$$

where $R_{\leq j}(s,Y)$ is the correction term defined by:

$$R_{\leq j}(s,Y) = \sum_{\emptyset \subset K \subseteq L(s,Y)} (-1)^{|K|+1} R^K_{\leq j}(s,Y)$$

with $R^K_{\leq j}(s,Y) = \sum_{k=1}^{j} \Phi_{\leq j-k}(s^{min(K)-1}) \times \binom{|s[K] \cap Y|}{k}$ where $s[K] = \cap_{k \in K} s[k]$.

This Theorem 1 extends the proposal [29] by setting $j = \infty$.

Proof: Let s be a sequence and Y be an itemset. We already explain that to construct a subsequence of $s \circ Y$ having a norm less than j, we can concatenate any subset of size k of Y to a subsequence of s having a norm less than $j-k$. Indeed, we are sure to obtain a subsequence of $s \circ Y$ having a norm less than $k+(j-k)=j$. Thus, we have $\phi_{\leq j}(s \circ Y) = \cup_{k=0}^{j} \phi_{\leq j-k}(s) \circ \mathcal{P}_{=k}(Y)$ and $\Phi_{\leq j}(s \circ Y) = \sum_{k=0}^{j} \Phi_{\leq j-k}(s) \times \binom{|Y|}{k} - R_{\leq j}(s,Y)$ where $R_{\leq j}(s,Y)$ is a correction term (to count the number of *distinct* subsequences).

Let $t = \langle T_1 \dots T_m \rangle$ with $|T_m| = k$ be a sequence that is counted multiple times, i.e. $t \in \phi_{\leq j}(s) \cap (\phi_{\leq j}(s) \circ \mathcal{P}_{\geq 1}(Y))$ where $\mathcal{P}_{\geq 1}(Y) = \{X \subseteq Y : |X| \geq 1\}$. Because $t \in (\phi_{\leq j}(s) \circ \mathcal{P}_{\geq 1}(Y))$, we necessarily have $T_m \in \mathcal{P}_{\geq 1}(Y)$, i.e. $T_m \subseteq Y$. Moreover, because $t \in \phi_{\leq j}(s)$, there exists an integer $i \leq |s|$ such that $T_m \subseteq s[i]$. Let $l = max\{i \leq |s| : T_m \subseteq s[i]\}$. Since $T_m \subseteq Y$, we also have $l = max\{i \leq |s| : T_m \subseteq (s[i] \cap Y)\}$. We show now that $l \in L(s,Y)$. First, because $T_m \neq \emptyset$, we have $s[l] \cap Y \neq \emptyset$. Now, assume that there exists $l' > l$ such that $s[l] \cap Y \subseteq s[l'] \cap Y$. Then, we would have $T_m \subseteq s[l'] \cap Y$, which contradicts that l is maximal, and completes the proof that $l \in L(s,Y)$. At this point, we proved that $T \in \phi_{\leq j-k}(s^{l-1}) \circ \mathcal{P}_{=k}(s[l] \cap Y)$ for an integer $l \in L(s,Y)$. Thus, we have $R_{\leq j}(s,Y) = |\bigcup_{l \in L(s,Y)} (\cup_{k=1}^{j} \phi_{\leq j-k}(s^{l-1}) \circ \mathcal{P}_{=k}(s[l] \cap Y))|$.

Using the inclusion-exclusion principle, we rewrite $R_{\leq j}(s,Y)$ as $\sum_{\emptyset \subset K \subseteq L(s,Y)} (-1)^{|K|+1} R^K_{\leq j}(s,Y)$ with $R^K_{\leq j}(s,Y) = |\bigcap_{l \in K} (\cup_{k=1}^{j} \phi_{\leq j-k}(s^{l-1}) \circ \mathcal{P}_{=k}(s[l] \cap Y))|$.

[1]By convention, we consider that $\binom{n}{p} = 0$ if $p > n$.

Now, let $t = \langle T_1 \dots T_m \rangle$ be a sequence in the set $\bigcap_{l \in K} (\cup_{k=1}^{j} \phi_{\leq j-k}(s^{l-1}) \circ \mathcal{P}_{=k}(s[l] \cap Y))$. We necessarily have $t^{m-1} \in \phi_{\leq j-k}(s^{min(K)-1})$ and $T_m \in \cap_{l \in K} \mathcal{P}_{=k}(s[l] \cap Y)$, i.e. $T_m \in \mathcal{P}_{=k}(s[K] \cap Y)$ with $s[K] = \cap_{l \in K} s[l]$. It follows that $R^K_{\leq j}(s,Y) = |\cup_{k=1}^{j} \phi_{\leq j-k}(s^{min(K)-1}) \circ \mathcal{P}_{=k}(s[K] \cap Y))|$. Finally, because the sets $\phi_{\leq j-k}(s^{min(K)-1}) \circ \mathcal{P}_{=k}(s[K] \cap Y))$ are disjoints, we have $R^K_{\leq j}(s,Y) = \sum_{k=1}^{j} \Phi_{\leq j-k}(s^{min(K)-1}) \times \binom{|s[K] \cap Y|}{k}$, which completes the proof of Theorem 1. ■

By continuing Example 4 with the sequence $s = \langle (ab)c(ac) \rangle$, the following example illustrates the principle of the formula given by Theorem 1.

Example 5: The set $\phi_{\leq 2}(s^1)$ of subsequences of $s^1 = \langle (ab) \rangle$ with a norm less than 2 is defined by $\phi_{\leq 2}(s^1) = \{\langle\ \rangle, \langle a \rangle, \langle b \rangle, \langle (ab) \rangle\}$. We have $\Phi_{\leq 2}(s^1) = 4$, and it is easy to see that $\Phi_{\leq 1}(s^1) = 3$ (the subsequence $\langle (ab) \rangle$ having a norm strictly greater than 1). As $L(s^1, s[2]) = \emptyset$, we have $R_{\leq 2}(s^1, s[2]) = 0$ and $\Phi_{\leq 2}(s^2) = \sum_{k=0}^{|(c)|} \Phi_{\leq 2-k}(s^1) \times \binom{|(c)|}{k} = \Phi_{\leq 2}(s^1) \times \binom{1}{0} + \Phi_{\leq 1}(s^1) \times \binom{1}{1} = 4 + 3 = 7$. The first term of the sum corresponds to 4 subsequences in s^3 obtained by concatenating the empty set to subsequences of s^2, while the second term corresponds to 3 subsequences in s^3 obtained by concatenating the itemset (c) to each subsequence of s^2 having a norm less than 1. Let us detail the calculation of $\Phi_{\leq 2}(s^3) = \sum_{k=0}^{|(ac)|} \Phi_{\leq 2-k}(s^2) \times \binom{|(ac)|}{k} - R_{\leq 2}(s^2, s[3]) = \Phi_{\leq 2}(s^2) + \Phi_{\leq 1}(s^2) \times 2 + \Phi_{\leq 0}(s^2) - R_{\leq 2}(s^2, s[3]) = 7 + 4 \times 2 + 1 - R_{\leq 2}(s^2, s[3])$. For instance, the second term of $\Phi_{\leq 2}(s^3)$, that equals to 4×2, refers to the number of subsequences in s^3 that are obtained by concatenating the two subsets of size 1 of (ab) with a subsequence in s^2 having a norm less that 1. Finally, the calculation of the correction term $R_{\leq 2}(s^2, s[3])$ is as follows: $R_{\leq 2}(s^2, s[3]) = (-1)^2 \Phi_{\leq 1}(s^0) \times \binom{|(a)|}{1} + (-1)^2 \Phi_{\leq 1}(s^1) \times \binom{|(c)|}{1} = 1 + 3 = 4$. Thereby, we deduce that $\Phi_{\leq 2}(s^3) = 7 + 4 \times 2 + 1 - 4 = 12$.

The formula given by Theorem 1 is recursive. Nevertheless, given a sequence s and a maximum norm M, this recursion can easily be removed by calculating line by line the matrices T and R defined by:

- $T[i][j] = \Phi_{\leq j}(s^i)$ for $i \in [0..|s|]$ and $j \in [0..M]$. $T[i][j]$ is the number of subsequences with a norm less than or equal to j in the sequence s^i.
- $R[i][j] = R_{\leq j}(s^{i-1}, s[i])$ for $i \in [2..|s|]$ and $j \in [0..M]$. This correction term is the term required to correct the number of subsequences with a norm less than j of $s^i = s^{i-1} \circ s[i]$ using the number of subsequences with a norm less than j of s^i by concatenating the subsets of $s[i]$.

Algorithm 2 details how the matrices T and R can be computed for a sequence s and a maximum norm M. At each iteration of the main loop (lines 5 to 19 of Algorithm 2), it computes the number $T[i][j]$ of subsequences s^i of s with a norm less than or equal to j (for all $j \in [1..M]$) using the previous lines of matrices T and R. For each $i \in [2..|s|]$ and $j \in [1..M]$, Algorithm 2 first computes the correction term $R[i][j]$ (lines 7-13). Because $K \subseteq L(s^{i-1}, s[i])$, it is important to note that $m = min(K) \leq i-1 < i$. Thus, at line 11, it ensures that to calculate $R[i][j]$, only previously calculated

terms $T[m-1][j-k]$ of T are used. Then, Algorithm 2 computes (lines 14-17) the value of $T[i][j]$ using only the previous line $i-1$ of matrix T (line 15) and the correction term $R[i][j]$ (line 17). Examples of the matrices T and R are provided by Table III for a sequence $s = \langle (ab)c(ac) \rangle$. In particular, we find the values $R[3][2] = R_{\leq 2}(s^2, s[3])$ and $T[3][2] = \Phi_{\leq 2}(s^3)$ computed in Example 5.

Algorithm 2 Number of subsequences with a maximum norm

Input: A sequence s and a maximal norm $M \leq \|s\|$
Output: A matrix T such that $T[i][j] = \Phi_{\leq j}(s^i)$

1: $T[0][0] := T[1][0] = 1$
2: **for** $j = 1$ to M **do**
3: $\quad T[0][j] := 1$ and $T[1][j] := T[1][j-1] + \binom{|s[1]|}{j}$
4: **end for**
5: **for** $i = 2$ to $|s|$ **do**
6: $\quad$ **for** $j = 1$ to M **do**
7: $\quad\quad R[i][j] := T[i][j] = 0$
8: $\quad\quad$ **for all** $K \in \mathcal{P}_{\geq 1}(L(s^{i-1}, s[i]))$ **do**
9: $\quad\quad\quad m := min(K)$ and $k_{max} := |s[K] \cap s[i]|$
10: $\quad\quad\quad$ **for** $k = 1$ to k_{max} **do**
11: $\quad\quad\quad\quad R[i][j] \mathrel{+}= (-1)^{|K|+1} T[m-1][j-k] \times \binom{k_{max}}{k}$
12: $\quad\quad\quad$ **end for**
13: $\quad\quad$ **end for**
14: $\quad\quad$ **for** $k = 0$ to $min\{j, |s[i]|\}$ **do**
15: $\quad\quad\quad T[i][j] \mathrel{+}= T[i-1][j-k] \times \binom{|s[i]|}{k}$
16: $\quad\quad$ **end for**
17: $\quad\quad T[i][j] := T[i][j] - R[i][j]$
18: $\quad$ **end for**
19: **end for**
20: **return**(T)

To conclude this section, using Theorem 1, note that we calculate the number of distinct subsequences in a sequence s having a norm between m and M as follows: $\Phi_{[m,M]}(s) = \Phi_{\leq M}(s) - \Phi_{\leq m-1}(s)$. In Algorithm 1, this formula makes it possible to calculate the initial weight $w(s)$ for each sequence s of the sequential database $\mathcal{S}$ (see line 1 of Algorithm 1).

TABLE III. EXAMPLES OF MATRICES T AND R

T[i][j]	$\leq \mathbf{0}$	$\leq \mathbf{1}$	$\leq \mathbf{2}$	$\leq \mathbf{3}$
$s^{\mathbf{0}} = \langle\rangle$	1	1	1	1
$s^{\mathbf{1}} = \langle (ab) \rangle$	1	3	4	4
$s^{\mathbf{2}} = \langle (ab)c \rangle$	1	4	7	8
$s^{\mathbf{3}} = \langle (ab)c(ac) \rangle$	1	4	12	21

R[i][j]	$\leq \mathbf{0}$	$\leq \mathbf{1}$	$\leq \mathbf{2}$
$s^1, s[\mathbf{2}] = c$	0	0	0
$s^2, s[\mathbf{3}] = (ac)$	2	4	5

C. Subsequence sampling by rejection

After randomly drawing a sequence $s \in \mathcal{S}$ proportionally to its weight $w(s)$ (line 2 of Algorithm 1) and an integer k between m and M according to the distribution $P_{[m,M]}(k)$ (line 3 of Algorithm 1), CSSAMPLING aims at returning a subsequence of norm k drawn uniformly from the sequence s (line 4 of Algorithm 1). The difficulty is not to favor the subsequences that have multiple occurrences within the sequence.

To cope with this difficulty, we use a rejection method by uniformly drawing an occurrence of the sequence s and rejecting it if this occurrence is not the first one. As each subsequence has a unique first occurrence, this approach ensures a uniform draw of subsequences. We start by formalizing the notion of first occurrence:

Definition 5 (First occurrence): Given a sequence s, let o_1 and o_2 be two occurrences of a subsequence s' within s, whose signatures are $\langle i_1^1, i_2^1, \ldots, i_m^1 \rangle$ and $\langle i_1^2, i_2^2, \ldots, i_m^2 \rangle$ respectively. o_1 is less than o_2, denoted by $o_1 < o_2$, if there exists an index $k \in [1..m]$ such that for all $j \in [1..k-1]$, one has $i_j^1 = i_j^2$, and $i_k^1 < i_k^2$. Finally, we call the *first occurrence* of s' in s its smallest occurrence w.r.t. the order defined previously.

Example 6: Let us continue Example 2 where $\langle 1, 2 \rangle$ and $\langle 1, 3 \rangle$ are the signatures of occurrences $o_1 = \langle (a)(c)\emptyset \rangle$ and $o_2 = \langle (a)\emptyset(c) \rangle$ of the subsequence $s' = \langle (a)(c) \rangle$ in $s = \langle (ab)(cd)(ce) \rangle$. As $\langle 1, 2 \rangle$ is less than $\langle 1, 3 \rangle$, we obtain that $o_1 < o_2$. Finally, as o_1 and o_2 are the only two occurrences of s' in s, it means that o_1 is the first occurrence of s' in s.

In practice, we especially check if an occurrence of the subsequence $s' \sqsubseteq s$ is the first occurrence of s' within the sequence s. This can be done efficiently by using Property 1:

Property 1: Given an occurrence o of the subsequence $s' \sqsubseteq s$ whose signature is $\sigma = \langle i_1, i_2, \ldots, i_m \rangle$, o is the first occurrence of s' if and only if for all $i_j \in \sigma$, there is no index $l \in [i_{j-1} + 1..i_j - 1]$ such that $o[i_j] \subseteq s[l]$ (with $i_0 = 0$).

Proof: Let $\sigma = \langle i_1, \ldots, i_m \rangle$ be the signature of an occurrence o of $s' \sqsubseteq s$. We first show that if there exist $i_j \in \sigma$ and $l \in [i_{j-1}+1..i_j-1]$ such that $o[i_j] \subseteq s[l]$, then o is not the first occurrence of s'. Let $1 \leq i'_1 < i'_2 < \cdots < i'_m \leq n$ be the index sequence defined by $i'_j = l$ and for all $k \in [1..m] \setminus \{j\}$, $i'_k = i_k$. Consider now the ordered list o' of n itemsets defined by $o'[l] = o[i_j]$, $o'[i_j] = \emptyset$ and for all $k \in [1..n] \setminus \{l, i_j\}$, $o'[k] = o[k]$. As o' is an occurrence of $s' \sqsubseteq s$ and $o' < o$, it proves that o is not the first occurrence of s'. Conversely, we show that if o of signature σ is not the first occurence of $s' \sqsubseteq s$, then there exist $i_j \in \sigma$ and $l \in [i_{j-1} + 1..i_j - 1]$ such that $o[i_j] \subseteq s[l]$. By definition, if o is not the first occurrence of s, then there exists another occurrence o' of s' such that $o' < o$. So, we know that there exists $k \in [1..n]$ such that $i'_k < i_k$ and for all $j \in [1..k-1]$, $i'_j = i_j$. Thus, there exist indexes $i_k \in \sigma$ and $l = i'_k \in [i'_{k-1} + 1..i_k - 1] = [i_{k-1} + 1..i_k - 1]$ such that $o[i_k] = o[i'_k] \subseteq s[i'_k]$, i.e. $o[i_k] \subseteq s[l]$. ■

Thanks to Property 1, it is finally easy to draw uniformly a subsequence of norm k in a sequence s. By randomly drawing k distinct item positions between 1 and $\|s\|$, we start by uniformly drawing an occurrence containing k items from s. If this occurrence is a first occurrence, it is accepted and returned. Otherwise we reject it and perform another random draw of a new occurrence of s. Although CSSAMPLING relies on a rejection sampling technique, we show in the next section that the average number of draws before acceptance is computable. The experimental section also shows that this average number of draws may be extremely low for real-world datasets.

Example 7: In Example 2, assume that we have drawn item positions 1 and 5 within the sequence $s = \langle (\mathbf{a}b)(cd)(\mathbf{c}e) \rangle$ in order to build an occurence of a subsequence of s of norm $k = 2$. In this way, we obtain the occurrence $o = \langle (a)\emptyset(c) \rangle$ of signature $\langle 1, 3 \rangle$ of the subsequence $s' = \langle (a)(c) \rangle$ in s. In that case, as there exists $l = 2$ in $[1 + 1..3 - 1]$ such that

$o[3] = (c) \subseteq s[2] = (cd)$, we are sure that o is not the first occurrence of s' and this occurrence is rejected.

D. Theoretical analysis of the method

This property states that CSSAMPLING returns an exact sample of subsequences with norm constraints:

Property 2 (Soundness): Let $\mathcal{S}$ be a sequential dataset, m be a minimum norm and M a maximum norm, CSSAMPLING draws a subsequence of $\mathcal{S}$ having a norm between m and M according to a distribution proportional to frequency.

Proof: Let Z be the normalizing constant defined by $Z = \sum_{s \in \mathcal{S}} w(s) = \sum_{s \in \mathcal{S}} \Phi_{[m,M]}(s)$. Let t be a subsequence in $\mathbb{S}_{[m,M]}$ and $P(t)$ be the probability to draw subsequence t using Algorithm 1. We have: $P(t) = \sum_{s \in \mathcal{S}} P(t, s) = \sum_{s \in \mathcal{S}, t \sqsubseteq s} P(s) \times P(t/s)$. Considering the second line of Algorithm 1, we have $P(s) = \frac{w(s)}{Z} = \frac{\Phi_{[m,M]}(s)}{Z}$. Then, considering the third and fourth lines of Algorithm 1, if t is a subsequence of norm k, we have $P(t/s) = P(k/s) \times P(t/k, s) = \frac{\Phi_{[k,k]}(s)}{\Phi_{[m,M]}(s)} \times \frac{1}{\Phi_{[k,k]}(s)} = \frac{1}{\Phi_{[m,M]}(s)}$. Thus, we have $P(t) = \sum_{s \in \mathcal{S}, t \sqsubseteq s} P(s) \times P(t/s) = \sum_{s \in \mathcal{S}, t \sqsubseteq s} \frac{\Phi_{[m,M]}(s)}{Z} \times \frac{1}{\Phi_{[m,M]}(s)} = \frac{freq(s,\mathcal{S})}{Z}$, which shows that t is drawn proportionnaly to its frequency and completes the proof. ■

We now study the complexity of our method by distinguishing two main phases: the preprocessing (where the distribution of subsequences according to the norm is calculated for each sequence) and the drawing of subsequences.

a) Preprocessing complexity: The preprocessing is performed in time $O(|\mathcal{S}| \cdot L \cdot M^2 \cdot 2^P \cdot T^2)$ where L is the maximum length of a sequence, M is the maximum norm of drawn subsequences, P is the maximum size of position sets $L(s^{i-1}, s[i])$ and T is the maximum size of an itemset in a sequence. It is important to note that $P \leq L$ may be very small in practice (see the next section) and that this preprocessing (line 1 of Algorithm 1) is achieved only once before the drawing phase (where a large number of subsequences are drawn from $\mathcal{S}$). Moreover, it is important to note that if the dataset $\mathcal{S}$ contains only sequences of *items* (and not sequences of *itemsets*), then we have $P = 1$. Thus, in that case, the preprocessing can be performed in polynomial time $O(|\mathcal{S}| \cdot L \cdot M^2 \cdot T^2)$.

b) Drawing complexity: The draw of subsequences is less expensive. First, the draw of a sequence (line 2 of Algorithm 1) is realized in $O(\ln |\mathcal{S}|)$. It is more difficult to estimate the complexity in the worst case for the draw of a subsequence because the number of rejections is not bounded. Nevertheless, a good way to measure the effectiveness of the approach is to calculate the average number of draws, denoted by $\mu_{[m,M]}(\mathcal{S})$, required to derive a subsequence of $\mathcal{S}$ having a norm between m and M. Intuitively, $\mu_{[m,M]}(\mathcal{S})$ depends both on the probability that a sequence $s \in \mathcal{S}$ is drawn and the average number of draws, denoted by $\mu_{[m,M]}(s)$, required to find a first occurrence of a subsequence of s. The following property shows how these terms can be calculated:

Property 3 (Average number of draws): The average number of draws for the acceptance of a subsequence having a norm between m and M in the sequential dataset $\mathcal{S}$ is defined by: $\mu_{[m,M]}(\mathcal{S}) = \sum_{s \in \mathcal{S}} \frac{\Phi_{[m,M]}(s)}{\sum_{s' \in \mathcal{S}} \Phi_{[m,M]}(s')} \times \mu_{[m,M]}(s)$ where $\mu_{[m,M]}(s) = \frac{\sum_{k=m}^{M} \binom{\|s\|}{k}}{\Phi_{[m,M]}(s)}$.

Proof: Using Algorithm 1, it is clear that $\mu_{[m,M]}(\mathcal{S}) = \sum_{s \in \mathcal{S}} P(s) \times \mu_{[m,M]}(s)$ with $P(s) = \frac{\Phi_{[m,M]}(s)}{\sum_{s' \in \mathcal{S}} \Phi_{[m,M]}(s')}$. Then, we have $\mu_{[m,M]}(s) = \sum_{k \in [m..M]} P(k/s) \times N_k(s)$ where $N_k(s)$ is the average number of draws necessary to obtain a subsequence s' of s such that $\|s'\| = k$. When we draw a subsequence s' of norm k, the probability that this subsequence is accepted (because it is a first occurrence) is $P_a^k(s) = \frac{\Phi_{[k,k]}(s)}{\binom{\|s\|}{k}}$. Thus, we have $N_k(s) = \sum_{i=1}^{\infty} i \times (1 - P_a^k(s))^{i-1} \times P_a^k(s) = P_a^k(s) \times \sum_{i=1}^{\infty} i \times (1 - P_a^k(s))^{i-1} = P_a^k(s) \times \frac{1}{P_a^k(s)^2} = \frac{1}{P_a^k(s)}$. It follows that $\mu_{[m,M]}(s) = \sum_{k \in [m..M]} P(k/s) \times N_k(s) = \sum_{k \in [m..M]} \frac{\Phi_{[k,k]}(s)}{\Phi_{[m,M]}(s)} \times \frac{\binom{\|s\|}{k}}{\Phi_{[k,k]}(s)} = \frac{\sum_{k \in [m..M]} \binom{\|s\|}{k}}{\Phi_{[m,M]}(s)}$. ■

When the average number of draws is close to 1, it means that the draw of a subsequence is achieved without rejection. For a given sequence, there is no rejection if each occurrence is the first occurrence i.e., there is no duplicate within the sequence. In practice, the average number of draws measured on real-world datasets is often very low. Finally, as the temporal complexity of the draw of an occurrence having a norm equal to $k \in [m..M]$ in a sequence s is in the worst case in $O(M^2)$, the average complexity of drawing N subsequences from a dataset $\mathcal{S}$ (after the preprocessing phase) is in $O(N \cdot M^2 \cdot \mu_{[m,M]}(\mathcal{S}))$.

V. EXPERIMENTAL STUDY

In the previous section, we proved that our sampling algorithm CSSAMPLING is exact, and studied its complexity. In this section, we evaluate the efficiency of the approach and the interest of the sampled subsequences. More precisely, Section V-A focuses on the speed of CSSAMPLING and its ability to draw patterns that do not belong to the long tail. In Section V-B, in order to illustrate the usefulness of sampled patterns, we show how these patterns can be used to build associative classifiers dedicated to sequences and that our approach rivals state of the art proposal.

A. Analysis of CSSAMPLING method

This experimental section evaluates the speed of our method and the impact of the norm constraint on the sampled patterns. For this, we use 6 datasets including 2 real life datasets `bms` and `sign`[2] and 4 synthetic datasets generated by IBM data generator[3]. One of the interests of using synthetic datasets is to have examples where the average number of draws $\mu_{[m,M]}(\mathcal{S})$ is ensured to be greater than 1 by adding multiple occurrences within a same sequences. Table V lists basic statistics of all datasets and Table VI compares the average number of draws per subsequence required to extract a pattern with $M \in \{1, 2, 3, 5, 7\}$ (while m is always fixed to 1 in all of our experiments). The prototype of our method is implemented in Python and all experiments are performed on a 2.71 GHz 2 Core CPU with 12 GB of RAM. All experimental datasets used, as well as source code, are available at `https://github.com/LDIOPBSF/CSSampling`.

[2] `http://www.philippe-fournier-viger.com/spmf`
[3] `https://github.com/zakimjz/IBMGenerator`

TABLE IV. EXECUTION TIME FOR SEQUENTIAL PATTERN SAMPLING (AVERAGE AND STANDARD DEVIATION)

Dataset	Preprocessing time (s)					Drawing time per pattern (ms)				
	M=1	M=2	M=3	M=5	M=7	M=1	M=2	M=3	M=5	M=7
bms	0.22±0.00	0.29±0.01	0.30±0.00	0.30±0.01	0.31±0.02	0.07±0.01	0.25±0.01	0.43±0.02	0.59±0.00	0.7±0.01
sign	0.01±0.00	0.02±0.00	0.02±0.00	0.02±0.00	0.03±0.00	0.15±0.01	0.17±0.01	0.19±0.01	0.22±0.00	0.24±0.00
D10K5S2T6I	0.81±0.33	2.02±0.02	2.92±0.03	5.02±0.05	7.36±0.06	0.05±0.00	0.07±0.00	0.09±0.00	0.17±0.00	0.24±0.01
D10K6S3T10I	1.38±0.04	3.11±0.10	5.16±0.04	9.33±0.18	14.56±0.10	0.06±0.01	0.09±0.01	0.14±0.01	0.24±0.02	0.37±0.01
D100K5S2T6I	5.86±0.26	12.34±0.11	18.8±0.21	32.88±0.27	49.24±0.29	0.05±0.01	0.07±0.01	0.09±0.00	0.13±0.01	0.20±0.02
D100K6S2T6I	8.44±0.55	17.48±0.26	27.33±0.74	49.48±0.90	74.89±0.56	0.06±0.01	0.07±0.02	0.10±0.01	0.15±0.01	0.21±0.01

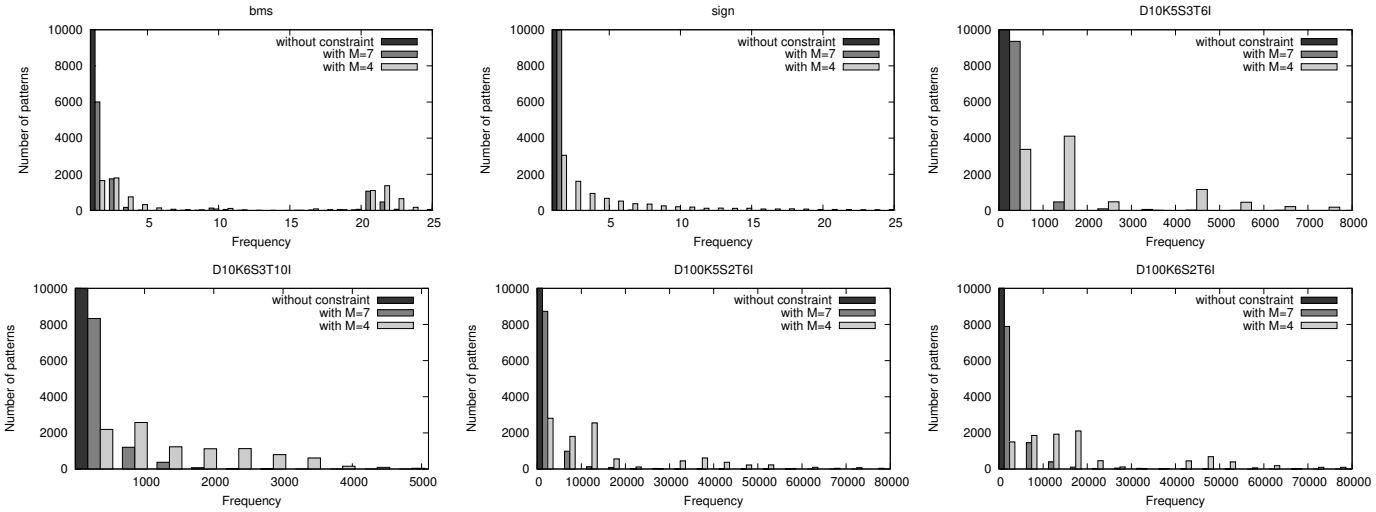

Fig. 2. Distribution of 10,000 sequential patterns according to frequency

TABLE V. STATISTICS OF BENCHMARK DATASETS

Dataset	$\|\mathcal{S}\|$	$\|\mathcal{I}\|$	$\|\|S\|\|_{max}$	$\|\|S\|\|_{mean}$	P	T
bms	59,601	497	267	2.5	1	1
sign	730	267	94	52.0	1	1
D10K5S2T6I	10,000	6	70	10.3	7	6
D10K6S3T10I	10,000	10	92	15.9	10	6
D100K5S2T6I	100,000	6	72	8.5	7	6
D100K6S2T6I	100,000	6	83	10.4	8	9

TABLE VI. AVERAGE NUMBER OF DRAWS PER SUBSEQUENCE

Dataset	M=1	M=2	M=3	M=5	M=7
bms	1.0	1.0	1.0	1.0	1.0
sign	1.0	1.0	1.0	1.0	1.0
D10K5S2T6I	4.0	7.0	11.4	23.5	38.4
D10K6S3T10I	3.9	6.7	10.4	18.5	25.7
D100K5S2T6I	3.6	5.8	8.5	14.9	23.9
D100K6S2T6I	4.0	7.0	11.1	21.4	32.4

1) Pre-processing and sampling speed: Table IV indicates the execution time of our method by distinguishing the preprocessing time and the average number of draws of a sequential pattern with $M \in \{1, 2, 3, 5, 7\}$. As expected, the preprocessing time increases with the size of the dataset, the maximum size P of position sets, the maximum size T of an itemset in a sequence, and the maximum norm M of drawn subsequences. However, even for D100K6S2T6I which is large, the execution time of the preprocessing (which can be prepared off-line) is quite reasonable. Regarding the sampling phase, whatever the dataset and the maximum norm M, the execution time is always under 1 millisecond. Despite an average number of draws $\mu_{[m,M]}(\mathcal{S})$ greater than 1 (and hence, rejection), performances on synthetic datasets are good.

2) Impact of norm constraints: Figure 2 depicts the distribution of 10,000 sequential patterns sampled according to frequency with a maximum norm constraint of 4, 7, and without constraint for different datasets. In all cases, the unconstrained method returns only very low frequent patterns and in particular, with 1 as frequency on real-world datasets. Conversely, the constrained sampling method returns sequential patterns with significantly higher frequency, which shows the importance of introducing constraints on the norm to avoid the problem of the long tail. More precisely, we can see that the lower the value of the M constraint is, the more the method allows to draw patterns with high frequency values. For instance, for D100K6S2T6I, the mean frequency of sampled patterns is equal to $3,770$ using $M = 7$, whereas it is equal to $19,683$ using $M = 4$. Note that for sign, the maximum norm of 7 is not sufficient to return sampled patterns with frequency greater than 1. A norm of at most 4 is necessary so that the frequencies of the subsequences of the sample increase. In that case, the mean frequency of sample patterns is equal to 8.65.

B. Accuracy of sampling-based classification

This section shows how sampled subsequences can be used to build associative classifiers dedicated to sequences. Our classification method, called CSSAMPLING+SVM, is a standard two-step approach. In a first step, using a sample $F = \{f_1, \ldots, f_k\}$ of k subsequences obtained using CSSAMPLING, a labeled sequential dataset $\mathcal{S}$ is recoded into a numerical dataset $\mathcal{D}$. More precisely, for each sequence $s \in \mathcal{S}$ labeled by a class c, $\mathcal{D}$ contains a tuple of $k+1$ values where $t[j] = 1$ if $f_j \sqsubseteq s$ (0 otherwise) for $j \in [1..k]$, and $t[k+1] = c$. Then, in a second step, using dataset $\mathcal{D}$, we propose to use a SVM as

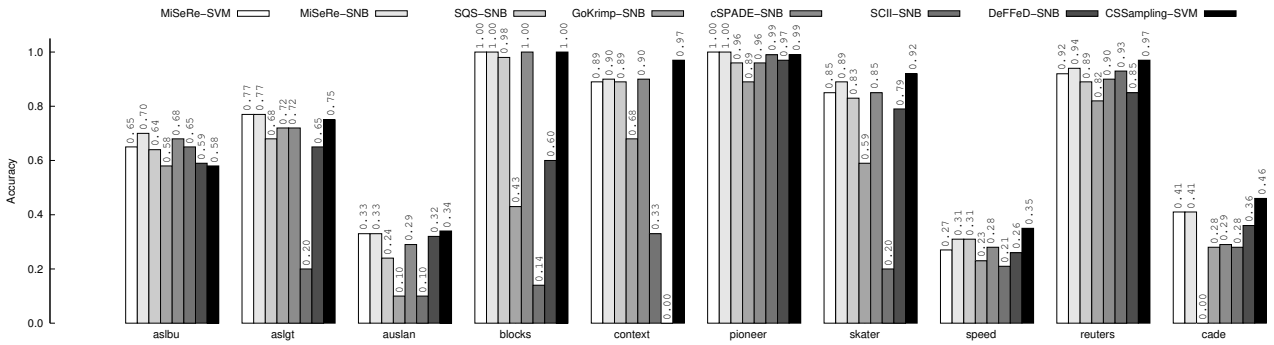

Fig. 3. Comparison of accuracy results between CSSAMPLING with SVM and state-of-the-art sequence classification methods.

TABLE VII. STATISTICS OF BENCHMARK DATASETS

Dataset	$\|\mathcal{S}\|$	$\|\mathcal{I}\|$	$\|S\|_{max}$	$\|S\|_{mean}$	$\|\mathcal{C}\|$
aslbu	441	132	27	7.52	7
aslgt	3,493	87	88	22.83	40
auslan	200	12	24	10.00	10
blocks	210	8	12	6.75	8
context	240	48	123	45.20	5
pioneer	160	92	50	21.07	3
skater	530	41	120	25.06	6
speed	530	41	260	64.50	7
reuters	5,459	14,577	533	67.32	8
cade	15,000	100,197	15,318	112.70	12

classifier for predicting the class of new sequences. Note that in our experiments, we use the SMO algorithm provided by Weka 3.8 and its default options to build SVM classifiers.

In order to evaluate the efficiency of CSSAMPLING+SVM, we use a set of real-world datasets [30][4] that have a wide variety in the number of sequences, items, sequence lengths and classes as well as application domains (see Table VII). For each dataset, we calculate the accuracy of CSSAMPLING+SVM with respect to varied sample sizes and norm constraints, by performing a 10-fold cross-validation.

TABLE VIII. IMPACT OF THE NORM CONSTRAINT ON CLASSIFICATION

Dataset	$M=1$	$M=2$	$M=3$	$M=5$	$M=7$	$M=10$	**Best**
aslbu	0.57	**0.58**	0.56	0.55	0.42	0.38	0.58
aslgt	0.73	**0.75**	**0.75**	0.72	0.59	0.43	0.75
auslan	0.24	0.24	**0.34**	0.32	0.32	0.32	0.34
blocks	0.86	**1.00**	0.99	0.99	0.99	0.99	1.00
context	0.94	0.96	**0.97**	**0.97**	0.96	0.95	0.97
pioneer	**0.99**	**0.99**	0.98	0.87	0.74	0.66	0.99
skater	0.84	0.90	**0.92**	**0.92**	0.88	0.73	0.92
speed	0.24	0.29	**0.35**	**0.35**	**0.35**	0.23	0.35
reuters	**0.97**	0.95	0.85	0.56	0.52	0.52	0.97
cade	**0.46**	0.33	0.25	0.22	0.22	0.21	0.46
Average	0.68	**0.70**	**0.70**	0.69	0.64	0.58	0.76

1) Importance of the norm constraint: As described in previous sections, the norm constraint M is introduced to limit the maximal length of sampled subsequences since too long patterns have been proved less useful in pattern discovery. Table VIII shows that the accuracy of CSSAMPLING+SVM clearly depends on the norm constraint. While the total size of sample is fixed (here, 10,000 patterns), the best classification performance is generally obtained when the maximum norm threshold is strictly larger than 1 (except for datasets reuters and cade, as observed in [30]) and lower that 10. Given a dataset, the optimal value of M (**Best** column in Table VIII) can be easily identified using cross-validation (evaluating the performance of CSSAMPLING+SVM for $M \in [1..10]$). Finally, note that the performance of classifiers decreases with M when M is greater that its optimal value, which shows the importance to consider maximum norm thresholds to build efficient classifiers. In particular, the performance of classifiers that would be obtained without considering norm constraints (i.e., $M \to \infty$) would therefore be very low.

2) Comparison with pattern-based sequence classification methods: We finally compare the accuracy of CSSAMPLING+SVM with the results of 7 state-of-the-art sequence classification methods reported in [30] as baselines with respect to the same datasets: MISERE, SQS, GOKRIMP, CSPADE, SCII and DEFFED. Figure 3 shows that the best accuracies obtained by CSSAMPLING+SVM (column **Best** of Table VIII) are comparable, even better according to datasets, to other pattern-based sequence classification methods reported in [30]. Notice that the goal of this paper is not to propose a new sequence classification method, we just want to illustrate that subsequence sampling is useful in some applications.

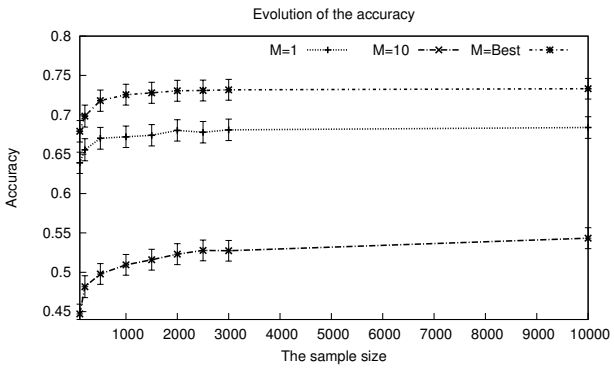

Fig. 4. Impact of the sample size on classification performance.

3) Impact of the sample size: Depending on applications, in particular to classification tasks, the impact of sample size shall not be ignored with our classification method. Obviously, the accuracy of the classification increases with the sample size because the sequences are more likely to be covered by at least one subsequence. Figure 4 shows the classification

[4]The datasets reuters and cade are available at ana.cachopo.org/datasets-for-single-label-text-categorization and other ones, at www.mybytes.de/#data.

performance, considered as average accuracy values over all datasets, obtained by different sample sizes with respect to norm constraint values 1, 10 and **Best** mentioned in Table VIII. It is easy to observe that the classification performance increases while more sampled sequential patterns are involved (which is useful for developing an anytime approach). Interestingly, the accuracy increases very quickly with the sample size. Thus a classifier built in a short response time considering only 1,000 subsequences competes with methods of the state of the art where all the pattern search space is explored.

VI. Conclusion

This paper proposes the first output space sampling method for sequential patterns. It also allows to specify an interval constraint on the norm of sequential patterns to better control the returned patterns. We have demonstrated that our sampling algorithm is exact and we have estimated its efficiency with respect to the average number of rejections which increases with the number of occurrences within a sequence. The experimental study shows that the approach is very efficient on real-world datasets where the number of repetitions is low. Moreover, the experiments show that the addition of constraints on the norm avoids returning too many patterns too rare and focuses the sampling on the patterns of the "head" as desired. Finally, we illustrated how to build a classifier in a very short response time by just drawing a sample containing 1,000 patterns. These models still have an accuracy comparable to some methods achieving a complete enumeration of the pattern search space.

We would like to extend our approach to other interestingness measures and to any set system. First, the draw weight of a sequence could be calculated for interestingness measures $u(s) \times freq(s, \mathcal{S})$ (where the utility u depends only on the sequence norm) because the utility can be integrated into the subsequence counting formulas. Second, the uniform drawing within complex structures made possible by a canonical form (here the first occurrence) can be envisaged with other structured languages. As was the case with the itemsets, we think that the results about associative classification are promising for addressing other data mining tasks like detecting outliers in sequential data [6] or for designing interactive systems dedicated to sequential pattern discovery [3].

Acknowledgements. This work has been partly supported by the CEA-MITIC (Centre d'Excellence Africain en Mathématiques, Informatique et TIC).

References

[1] M. van Leeuwen, "Interactive data exploration using pattern mining," in *Interactive Knowledge Discovery and Data Mining in Biomedical Informatics*, 2014, pp. 169–182.

[2] S. Zilberstein, "Using anytime algorithms in intelligent systems," *AI magazine*, vol. 17, no. 3, p. 73, 1996.

[3] M. Bhuiyan, S. Mukhopadhyay, and M. A. Hasan, "Interactive pattern mining on hidden data: a sampling-based solution," in *Proc. of CIKM 2012*, 2012, pp. 95–104.

[4] A. Giacometti and A. Soulet, "Interactive pattern sampling for characterizing unlabeled data," in *Proc. of IDA 2017*, 2017, pp. 99–111.

[5] V. Dzyuba, M. v. Leeuwen, S. Nijssen, and L. De Raedt, "Interactive learning of pattern rankings," *Int. Journal on Artificial Intelligence Tools*, vol. 23, no. 06, p. 32 pages, 2014.

[6] A. Giacometti and A. Soulet, "Anytime algorithm for frequent pattern outlier detection," *International Journal of Data Science and Analytics*, vol. 2, no. 3-4, pp. 119–130, 2016.

[7] M. Al Hasan and M. J. Zaki, "Output space sampling for graph patterns," *Proc. of the VLDB*, vol. 2, no. 1, pp. 730–741, 2009.

[8] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Proc. of ICDE 95*, 1995, pp. 3–14.

[9] J. Han, H. Cheng, D. Xin, and X. Yan, "Frequent pattern mining: current status and future directions," *Data mining and knowledge discovery*, vol. 15, no. 1, pp. 55–86, 2007.

[10] C. Anderson, "The long tail," *Wired magazine*, vol. 12, no. 10, pp. 170–177, 2004.

[11] M. Boley, C. Lucchese, D. Paurat, and T. Gärtner, "Direct local pattern sampling by efficient two-step random procedures," in *Proc. of SIGKDD 2011*, 2011, pp. 582–590.

[12] R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements," in *Proc. of EDBT 96*, 1996, pp. 3–17.

[13] M. J. Zaki, "SPADE: An efficient algorithm for mining frequent sequences," *Machine Learning*, vol. 42, no. 1-2, pp. 31–60, 2001.

[14] J. Pei, J. Han, B. Mortazavi-Asl, and H. Pinto, "PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth," in *Proc. of ICDE 2001*, 2001, pp. 215–224.

[15] M. N. Garofalakis, R. Rastogi, and K. Shim, "Spirit: Sequential pattern mining with regular expression constraints," in *VLDB*, vol. 99, 1999, pp. 7–10.

[16] J. Pei, J. Han, and L. V. Lakshmanan, "Mining frequent itemsets with convertible constraints," in *Proc. of ICDE 2001*. IEEE, 2001, pp. 433–442.

[17] J. Wang and J. Han, "Bide: Efficient mining of frequent closed sequences," in *Proc. of ICDE 2004*. IEEE, 2004, pp. 79–90.

[18] X. Yan, J. Han, and R. Afshar, "Clospan: Mining: Closed sequential patterns in large datasets," in *Proc. of SDM 2003*. SIAM, 2003, pp. 166–177.

[19] G. Bosc, J.-F. Boulicaut, C. Raïssi, and M. Kaytoue, "Anytime discovery of a diverse set of patterns with monte carlo tree search," *Data Mining and Knowledge Discovery*, pp. 1–47, 2016.

[20] H. Toivonen *et al.*, "Sampling large databases for association rules," in *Proc. of VLDB 96*, vol. 96, 1996, pp. 134–145.

[21] C. Luo and S. M. Chung, "A scalable algorithm for mining maximal frequent sequences using sampling," in *Proc. of ICTAI 2004*. IEEE, 2004, pp. 156–165.

[22] C. Raissi and P. Poncelet, "Sampling for sequential pattern mining: From static databases to data streams," in *Proc. of ICDM 2007*, 2007, pp. 631–636.

[23] A. A. Bendimerad, M. Plantevit, and C. Robardet, "Unsupervised exceptional attributed sub-graph mining in urban data," in *Proc. of ICDM 2016*. IEEE, 2016, pp. 21–30.

[24] A. Giacometti and A. Soulet, "Dense neighborhood pattern sampling in numerical data," in *Proc. of SDM 2018*, 2018, pp. 756–764.

[25] S. Moens and B. Goethals, "Randomly sampling maximal itemsets," in *Proc. of IDEA Workshop 2013*, 2013, pp. 79–86.

[26] M. Boley, T. Gärtner, and H. Grosskreutz, "Formal concept sampling for counting and threshold-free local pattern mining," in *Proc. of SDM 2010*. SIAM, 2010, pp. 177–188.

[27] S. Moens and M. Boley, "Instant exceptional model mining using weighted controlled pattern sampling," in *Proc. of IDA 2014*. Springer, 2014, pp. 203–214.

[28] V. Dzyuba, M. van Leeuwen, and L. De Raedt, "Flexible constrained sampling with guarantees for pattern mining," *Data Mining and Knowledge Discovery*, vol. 31, no. 5, pp. 1266–1293, 2017.

[29] E. Egho, C. Raïssi, T. Calders, N. Jay, and A. Napoli, "On measuring similarity for sequences of itemsets," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 732–764, 2015.

[30] E. Egho, D. Gay, M. Boullé, N. Voisine, and F. Clérot, "A user parameter-free approach for mining robust sequential classification rules," *Knowl. Inf. Syst.*, vol. 52, no. 1, pp. 53–81, 2017.