Under consideration for publication in Knowledge and Information Systems

Sequential Pattern Sampling with Norm-based Utility

Lamine Diop[1,2], Cheikh Talibouya Diop[1], Arnaud Giacometti[2], Dominique Li[2] and Arnaud Soulet[2]

[1]University of Gaston Berger of Saint-Louis, Senegal
Email: {diop.lamine3, cheikh-talibouya.diop}@ugb.edu.sn
[2]University of Tours, France
Email: {arnaud.giacometti, dominique.li, arnaud.soulet}@univ-tours.fr

Abstract. Sequential pattern mining has been introduced by [1] two decades ago and its usefulness has been widely proved for different mining tasks and application fields such as web usage mining, text mining, bioinformatics, fraud detection and so on. Since 1995, despite numerous optimization proposals, sequential pattern mining remains a costly task that often generates too many patterns. This limit, also reached by itemset mining, was circumvented by pattern sampling. 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 different kinds of patterns including itemsets and subgraphs has not yet been applied to sequential patterns. In this paper, we propose the first method dedicated to sequential pattern sampling. In addition to address sequential data, the originality of our approach is to introduce a class of interestingness measures relying on the norm of the sequence, named *norm-based utilities*. In particular, it enables to add constraints on the norm of sampled patterns 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 two-step random procedure integrating this class of measures, named NUSSampling, that randomly draws sequential patterns according to frequency weighted by a norm-based utility. We demonstrate that this method performs an exact sampling according to the underlying measure. Moreover, despite the use of rejection sampling, the experimental study shows that NUSSampling remains efficient. We especially focus on the interest of norm constraints and exponential decays that help 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 sequential pattern sampling with norm-based utility.

Keywords: Pattern Mining, Pattern Sampling, Sequential Data

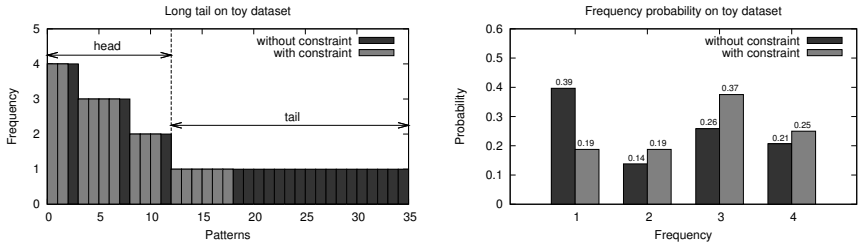

Fig. 1. Impact of the long tail on frequent sequential pattern sampling

1. Introduction

Sequential pattern mining has been introduced by [1] two decades ago and its usefulness has been proved for a wide range of mining tasks and application fields [2] such as web usage mining, text mining, fraud detection and so on. Since 1995, many methods have optimized the mining of sequential patterns [3, 4, 5] and have introduced variants with constraints [6, 7] or condensed representations [8, 9]. 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 languages (e.g., itemset or subgraph), was circumvented by Monte Carlo tree search [10] or pattern sampling [11]. Introduced in [11], 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. This kind of instantaneous methods is at the core of many approaches that make data mining more interactive [12, 13, 14, 15, 16]. To the best of our knowledge, there is no work addressing pattern sampling in sequential data.

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 [17]. 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 3. 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 5). 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. Other relevant interestingness measures like the area (which is the frequency of a pattern times its size) suffer from the same difficulties. Of course, we could use another interestingness measure such as a power of the frequency to mitigate the long tail issue, but this kind of measures are more expensive due to preprocessing time (as discussed in Sections 2 and 6). Is it possible to simply keep the frequency or the area to assess the interest of a pattern while cutting this long tail?

To circumvent the pitfall of the long tail, we propose in [18] 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 interestingly, it still allows 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 [15]. 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). Rather than using a binary constraint on the norm to limit the effect of the long tail, it is also possible to use a fuzzy constraint by using an exponential decay to mitigate the long tail. Therefore, we show in this paper how to consider a larger class of interestingness measures taking into account the norm of sequences. To achieve this goal, we propose to use the two-step random procedure [19] 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 a class of measures using utility functions that requires to count subsequences of a certain length.

The main contributions of the paper are as follows:

- We introduce a new class of interestingness measures based on *norm-based utility* functions. This class of measures enables us to consider classical measures like the frequency or the area. But, it also allows us to consider constraints (i.e., function whose range is 0 or 1) on the norm to remove too long patterns and to prevent the long tail issue.
- We propose a new algorithm named NUSSampling (Norm-based Utility Subsequence Sampling) that samples sequential patterns proportionally to frequency weighted by a norm-based utility. It relies on a two-step random procedure that requires solving two sub-problems: (i) counting the number of distinct subsequences for a given norm and (ii) uniformly drawing subsequences. We demonstrate that NUSSampling performs an exact sequential pattern sampling according to the underlying norm-based utility, and we analyze its complexity on average.
- We present a large set of experimental results for analyzing the behavior of NUSSampling. We show on several datasets that our approach is efficient enough to return hundreds of sequential patterns per second whatever the norm-based utility. We also highlight the practical interest of norm constraints

or exponential decays to better control the quality of the returned patterns and to avoid the curse of the long tail (for the frequency or the area).
– Sequence classification is a crucial data mining task useful in a wide range of applications. We investigate how sequential pattern sampling leads 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.

This paper is an extended version of [18] that generalizes the notion of frequent sequential pattern sampling under constraint to any norm-based utility. Consequently, we extend our algorithm to this class and perform additional experiments. There is also a comparison of the class of norm-based utility functions with other existing classes in the literature and several improvements are discussed.

The outline of this paper is as follows. Section 2 reviews some related work about pattern sampling methods. Section 3 introduces basic definitions, the class of norm-based utilities and the formal problem statement. We present our constrained two-step random procedure for sequential pattern sampling in Section 4. We evaluate our approach in Section 5 by carrying out a study on real-world and benchmark datasets, and by comparing the accuracy of a sample-based classifier with the state of the art methods. Then, Section 6 explores some limits and possible improvements of the proposed approach, in particular for dealing with discriminative measures. Finally, we conclude and introduce some perspectives in Section 7.

2. Related Work

After discussing in Section 2.1 the usefulness of pattern sampling to develop user-centered pattern mining methods, we present in Section 2.2 a brief overview of the different classes of output space pattern sampling methods proposed in the literature. In particular, we underline the advantages of multi-step methods compared with the methods using a random walk or the SAT framework.

2.1. Interest of output space pattern sampling

In recent years, the field of pattern mining has shifted to user-centered methods [29]. Typically, the idea is 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 [30, 31] or within a short response time of only a few seconds. Pattern sampling is an efficient approach that instantly returns patterns, which enables to design interactive systems [12, 13, 14] or to produce pattern-based models at any time [15]. For instance, [15] shows how to approximate the Frequent Pattern Outlier Factor [32] with guarantees by using frequent itemset sampling. Importantly, it is necessary to distinguish between input and output space sampling. The *input space* sampling [33] consists in generating from a sample of data all the patterns that would have been mined

Table 1. Typology of outspace pattern sampling methods

Type	Guarantees	Reference	Language	Measure	Constraint
Random walk: MCMC	convergence	[11, 12]	graphs	frequency	no
		[20]	itemsets	any strictly positive measure	closed
		[21]	itemsets	frequency	minimal
Random walk: Heuristics	no	[22]	itemsets	frequency	maximal
		[23]	graphs	weighted relative accuracy	valid
SAT	bounds	[24, 25]	itemsets	any strictly positive measure	any constraint
Multi-step	exact	[19]	itemsets	frequency, area, discriminative, power of frequency	no
		[26]	itemsets	frequency, area, discriminative, power of frequency, rare	no
		[27]	itemsets	exceptional model	no
		[28]	numerical	density	no

from the complete dataset. The *output space* sampling [11] consists in generating a sample of patterns among the patterns that would have been mined from the complete dataset. More formally, pattern sampling [11, 19] 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 [34, 35], 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 [28].

2.2. Pattern sampling techniques

Three main classes of methods have been proposed for the output space sampling of patterns (see Table 1 for an overview).

The first class of methods uses *random walks* in the search space to sample interesting patterns. In this class, most of the methods [11, 12, 20, 21] are based on Markov Chain Monte Carlo algorithms (MCMC). The idea is to construct a Markov chain that has the desired probability distribution as its equilibrium distribution. This means that a sampled pattern according to the desired distribution is obtained by observing the Markov chain after a number of iterations. The problem is that MCMC methods often take a large number of iterations before converging to the desired stationary distribution, i.e. the convergence is guaranteed but may be very slow. Therefore, *heuristic methods* [23, 22] have been proposed to accelerate the convergence and favor the patterns that are the most relevant (according to a given interestingness measure). In practice, it has been shown that these methods are more efficient in terms of computation time.

However, as they use heuristics to simulate a random walk and explore the pattern language, they offer no guarantee on the quality of the outputted sample. Note finally that [22, 20] consider constraints to restrict the sampling space and return only maximal and closed itemsets, respectively.

The second class of methods [24] benefits from the SAT framework (Boolean SATisfiability problem framework) to restrict the sampling using any kind of constraints. For example, it has been used to sample only closed frequent itemsets, frequent itemsets with a minimum length and minimum support, etc. Based on the SAT framework, [24] requires to have a solver integrating efficiently XOR constraints and in practice, it has been implemented only for itemsets. Compared to the class of methods based on MCMC, it is important to note that the performed sampling is not exact. Nevertheless, the class of methods based on SAT offers interesting theoretical guarantees. More precisely, the probability that the method samples a random pattern that satisfies a given constraint lies within a bounded range determined by the quality of the pattern (according to an interestingness measure) and an error tolerance. Finally, the authors [24] emphasize that the efficiency of their generic approach will hardly compete with approaches dedicated to a single language and/or class of constraints. Recently, [25] has improved the performance of this approach by executing it on an FPGA.

The third class of methods [19, 28, 27] consists in drawing an instance of the dataset and then drawing a pattern contained in this instance. By judiciously selecting the two sampling distributions, it is possible to obtain an exact sampling according to the desired final distribution. Recently, [28] adds a third step for taking into account numerical data where the pattern language is infinite. In this paper, we opted for such a multi-step random procedure for its speed and accuracy. Section 4.1 underlines specific challenges for achieving this goal in the case of sequences. Note that one of the shortcomings of multi-step random procedures is the preprocessing stage, which is quadratic with the size of the dataset for discriminative measures (and even worse with other specific interestingness measures). However, it is possible to use a stochastic technique, named coupling from the past, to accelerate this preprocessing [26]. In this paper, we will not benefit from this technique because we focus on the class of norm-based utilities whose preprocessing remains linear (for both frequency and area). In [26], the authors show how to handle a very large set of measures to evaluate the interestingness of an itemset X, i.e. the set of measures of the form $f(X, \mathcal{S}) = u_\star(X) \times \pi_{i=1}^{K} q_i(X, \mathcal{S}_i)$ where $u_\star(X) = \bigstar_{x \in X} b(x)$ with $\bigstar \in \{\Pi, \sum\}$ and q_i is either the positive or negative frequency of X in a specific portion $\mathcal{S}_i$ of the input dataset $\mathcal{S}$.

Besides the inherent difficulty of addressing sequences rather than itemsets, using a two-step procedure, we already showed in [18] how to sample subsequences under a norm constraint with a probability proportional to their frequency. In this paper, we show how to handle a larger set of measures to evaluate the interestingness of a sequence s. Compared to [26], these measures have the form $f(s, \mathcal{S}) = u(s) \times \mathit{freq}(s, \mathcal{S})$ where u is a norm-based utility, i.e. a utility function that depends only on the norm of the sequence s, and $\mathit{freq}(s, \mathcal{S})$ is the frequency of s in the input dataset $\mathcal{S}$. This class of measures contains interestingness measures such as frequency or area as already done [19]. However, we cannot consider discriminative measures [19] or individual utilities on items [20]. Nevertheless, this class of measures allows to incorporate constraints on the norm of sampled sequences, which is crucial to reduce the long tail effect. Finally, in

Table 2. A sequential dataset $\mathcal{S}$ with 4 sequences

Sid	Sequence	#occurrences	$\Phi(s_i)$	$w_{freq}(s_i)$	$w_{area}(s_i)$	$w_{\in[1..2]}(s_i)$	$w_{decay}(s_i)$
s_1	$\langle(ab)c\rangle$	8	8	8	12	6	3.38
s_2	$\langle(ab)c(ac)\rangle$	32	25	25	65	11	5.72
s_3	$\langle c(ac)\rangle$	8	7	7	11	5	2.88
s_4	$\langle(ab)(cd)\rangle$	16	16	16	32	10	5.06

Section 6, we compare more precisely our class of measures with the class of measures introduced in [26], and we discuss its possible extensions.

3. Problem Statement

This section formalizes the problem of sequential pattern sampling with norm-based utilities. Before, we recall some preliminary definitions about sequences and we introduce the class of norm-based utilities.

3.1. 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 now recall the definitions of *subsequences* and *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 2 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. $\|s_1\| = |(ab)| + |(c)| = 2 + 1 = 3$. 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 $\Phi(s_1) = |\phi(s_1)| = 8$. The number of subsequences $\Phi(s_i)$ of all sequences $s_i \in \mathcal{S}$ is detailed in Table 2. The definition and the purpose of the four weights $w_{freq}(s_i)$, $w_{area}(s_i)$, $w_{\in[1,2]}(s_i)$ and $w_{decay}(s_i)$ are explained in Section 4.

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:

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 2, 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).

3.2. Norm-based utilities

Pattern discovery is based on interestingness measures that evaluate the quality of a pattern. One of the most popular interestingness measures is 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). The frequency of a subsequence $s \in \mathbb{S}$ in the sequential dataset $\mathcal{S}$, denoted by $\mathit{freq}(s, \mathcal{S})$, is defined by: $\mathit{freq}(s, \mathcal{S}) = |\{s' \in \mathcal{S} : s \sqsubseteq s'\}|$. It is also common to associate a utility according to the pattern norm as it is the case with the area measure $\mathit{area}(s, \mathcal{S}) = \mathit{freq}(s, \mathcal{S}) \times \|s\|$. For this reason, we are interested in the class of interestingness measures of the form $\mathit{freq}(s, \mathcal{S}) \times u(s)$ where u is a norm-based utility:

Definition 3 (Norm-based utility). A utility function u is a norm-based utility iff there exists a function $f_u : \mathbb{N} \to \mathbb{R}$ such that for every pattern $s \in \mathbb{S}$, one has $u(s) = f_u(\|s\|)$.

The set of all norm-based utilities is denoted by $\mathcal{U}$. In the remainder of the paper, for sake of simplicity, any utility implicitly refers to a norm-based utiliy. For instance, the utility $u_{area}(s) = \|s\|$ allows us to consider the area measure $\mathit{area}(s, \mathcal{S}) = \mathit{freq}(s, \mathcal{S}) \times \|s\|$ and in that case, one has $f_{u_{area}}(\ell) = \ell$. Obviously, let us notice that the norm-based utility $u_{\mathit{freq}}(s) = 1$ enables us to consider the frequency as interestingness measure. Besides, the utility $u_{\leq M}$ (resp. $u_{\geq m}$) defined as 1 if $\|s\| \leq M$ (resp. $\|s\| \geq m$) and 0 otherwise simulates a maximum (resp. minimum) norm constraint. Indeed, with the induced interestingness measure $\mathit{freq}(s, \mathcal{S}) \times u_{\leq M}(s)$ (resp. $\mathit{freq}(s, \mathcal{S}) \times u_{\geq m}(s)$), a pattern with a norm strictly greater than M (resp. lower than m), is judged useless (whatever its frequency). $\geq m$ and $\leq M$ are said to be norm-based utility constraints (where 1 means *true* and 0 means *false*). Finally, the utility $u_{decay}(s) = \alpha^{\|s\|}$, named exponential decay, is useful for penalizing long sequences but in a smooth way in comparison with $u_{\leq M}$.

Interestingly, it is possible to combine norm-based utilities thanks to arithmetical operations:

Property 1 (Arithmetical closure). The class of norm-based utilities is closed under arithmetical operations i.e., $u_1(s) \star u_2(s)$ is a norm-based utility if $u_1 \in \mathcal{U}$, $u_2 \in \mathcal{U}$ and $\star \in \{+, -, \times, /\}$.

This straightforward property enable us to combine several norm-based utilities for building complex ones. For instance, if q_1 and q_2 are two norm-based

Table 3. Examples of subsequences in $\mathbb{S}$ considering sequences in $\mathcal{S}$

Pattern s	$u_{\in[1..2]}(s)$	$freq(s,\mathcal{S})$	$u_{area}(s)$	**Pattern** s	$u_{\in[1..2]}(s)$	$freq(s,\mathcal{S})$	$u_{area}(s)$
$\langle a\rangle$	1	4	1	$\langle ac\rangle$	1	3	2
$\langle b\rangle$	1	3	1	$\langle ad\rangle$	1	1	2
$\langle c\rangle$	1	4	1	$\langle ba\rangle$	1	1	2
$\langle d\rangle$	1	1	1	$\langle bc\rangle$	1	3	2
$\langle(ab)\rangle$	1	3	2	$\langle bd\rangle$	1	1	2
$\langle(ac)\rangle$	1	2	2	$\langle ca\rangle$	1	2	2
$\langle(cd)\rangle$	1	1	2	$\langle cc\rangle$	1	2	2
$\langle aa\rangle$	1	1	2	$\langle c(ac)\rangle$	0	2	3
$\langle\rangle$	0	4	0	$\langle(ab)c\rangle$	0	3	3

utility constraints (i.e., norm-based utilities with $\{0,1\}$ as range), then $q_1 \wedge q_2$ and $q_1 \vee q_2$ are also norm-based utility constraints because $q_1 \wedge q_2 = q_1 \times q_2$ and $q_1 \vee q_2 = q_1 + q_2 - q_1 \times q_2$. Consequently, in the following, we consider the norm-based utility $u_{\in[m..M]} = u_{\geq m} \times u_{\leq M}$ for focusing the pattern discovery on patterns having a norm between m and M.

3.3. Problem of frequent sequential pattern sampling with a norm-based utility

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 $\mathcal{L}$ (with Z as nomalizing constant). Our goal is to randomly draw sequential patterns according to frequency under a norm-based utility:

Given a sequential dataset $\mathcal{S}$, a norm-based utility $u \in \mathcal{U}$, we aim at randomly drawing a subsequence $s \in \mathbb{S}$ with a probability distribution $\pi(s)$ proportional to its frequency in $\mathcal{S}$ weighted by u i.e.,

$$\pi(s) = \frac{freq(s,\mathcal{S}) \times u(s)}{\sum_{s' \in \mathbb{S}} freq(s',\mathcal{S}) \times u(s')}$$

One of the advantages of frequent pattern sampling [19] is to remove the minimum frequency threshold (always difficult to set) while our problem introduces two thresholds in the case of $u_{\in[m..M]}$: 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 3 provides a set of subsequences considering the sequential dataset $\mathcal{S}$, and gives the frequencies in $\mathcal{S}$ of all these subsequences. Additionally, we also give the norm-based utilies $u_{\in[1..2]}(s)$ and $u_{area}(s)$ for each subsequence s. For instance, because our problem is to draw a subsequence proportionally to its frequency times a norm-based utility, 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 $\langle ba\rangle$ (because $\langle ac\rangle$ and $\langle ba\rangle$ has the same norm and $u(\langle ac\rangle) = u(\langle ba\rangle)$ whatever the norm-based utility u). But, even if the subsequence $\langle(ab)c\rangle$ has a frequency of 3, it will not be drawn if we consider the utility $u_{\in[1..2]}$ (because its norm is 3 and greater than $M = 2$).

4. Two-Step Random Procedure With a Norm-based Utility

In this section, we present our algorithm called NUSSampling (for Norm-based Utility Subsequence Sampling) that samples sequential patterns This algorithm relies on a two-step random procedure presented in Section 4.1. We explain in Section 4.2 how to count the number of distinct subsequences for a given norm, and we show in Section 4.3 how this result can be used to sample a sequence with a probability proportional to its weight. Then, we present in Section 4.4 a rejection method to uniformly draw a subsequence from a sequence. Finally, in Section 4.5, we demonstrate that NUSSampling performs an exact sequential pattern sampling, and we analyze its complexity on average.

4.1. 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 [19] for sampling itemsets proportionally to their support. But, we adapt this random procedure to consider a norm-based utility on each pattern at both step.

Algorithm 1 NUSSampling

Input: A sequential dataset $\mathcal{S}$ and a norm-based utility u
Output: A sequence $s \in \mathbb{S}$ randomly drawn w.r.t $freq(s, \mathcal{S}) \times u(s)$, i.e. $s \sim \pi(\mathbb{S})$ where $\pi(s) = freq(s, \mathcal{S}) \times u(s)$
// Step 1: Sampling a sequence
1: Compute the weight w defined by $w(s) := \sum_{s' \sqsubseteq s} u(s')$ for all $s \in \mathcal{S}$
2: Draw a sequence s from $\mathcal{S}$ proportionally to w: $s \sim w(\mathcal{S})$
// Step 2: Sampling a subsequence
3: Compute the weight defined by $w_\ell(s) := \sum_{s' \sqsubseteq s \wedge \|s'\| = \ell} u(s')$ for all $\ell \in [0..\|s\|]$
4: Draw an integer ℓ proportionally to $w_\ell(s)$: $\ell \sim w_{[0..\|s\|]}(s)$
5: **return** A subsequence s' of norm ℓ randomly drawn from s:
$s' \sim unif(\{s' \sqsubseteq s : \|s'\| = \ell\})$ where $unif$ is the uniform distribution

Given a dataset $\mathcal{S}$ and a norm-based utility u, NUSSampling (Norm-based Utility Subsequence Sampling) returns a sequential pattern drawn proportionally to $freq(s, \mathcal{S}) \times u(s)$:

Step 1: Sampling a sequence In the first step (lines 1 and 2 of Algorithm 1), we start by summing for each sequence $s \in \mathcal{S}$ the norm-based utility of each subsequence, i.e. $w(s) = \sum_{s' \sqsubseteq s} u(s')$. We show in Section 4.3 that the calculation of $w(s)$ requires to count the number of subsequences in s having a norm ℓ. For this reason, we show in Section 4.2 how to extend the formula given in [36]. Then, this first step continues with the drawing of a sequence s from $\mathcal{S}$ proportionally to its weight $w(s)$. For instance, Table 2 provides the weights $w_{\in[1,2]}(s_i)$ and $w_{area}(s_i)$ 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.

Step 2: Sampling a subsequence In the second step, we randomly draw the norm ℓ of the subsequence of s that will be returned (line 4 of Algorithm 1). This number ℓ is randomly drawn with a probability proportional to the sum of norm-based utilities of subsequences in s having exactly ℓ as norm, i.e. according to the probability distribution $w_\ell(s)$ defined for all $\ell \in [0..\|s\|]$ by: $w_\ell(s) = \sum_{s' \sqsubseteq s \wedge \|s'\|=\ell} u(s')$. Finally, Algorithm 1 returns at line 5 a subsequence s' in s of norm ℓ according to a uniform distribution, meaning that each subsequence s' from s of norm ℓ will be drawn with the same probability. We show in Section 4.4 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 4.5.

4.2. Counting the number of distinct subsequences

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 [36] 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 that have a norm less than j and that 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 [36]). 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 [36] by setting $j = \infty$. Note that $\Phi_{\ell}(s)$ denotes $\Phi_{\leq \ell}(s) - \Phi_{\leq \ell-1}(s)$ if $\ell > 0$ and 1, otherwise.

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 $i \leq \lceil s \rceil$ such that $T_m \subseteq s[i]$. Let $l = max\{i \leq \lceil s \rceil : T_m \subseteq s[i]\}$. 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 $R_{\leq j}(s, Y) = |\bigcup_{l \in L(s,Y)} (\cup_{k=1}^{j} \phi_{\leq j-k}(s^{l-1}) \circ \mathcal{P}_{=k}(s[l] \cap Y))|$.

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

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 $R^K_{\leq j}(s,Y) = |\bigcap_{l \in K} (\cup_{k=1}^{j} \phi_{\leq j-k}(s^{l-1}) \circ \mathcal{P}_{=k}(s[l] \cap Y))|$. 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. Let $s = \langle (ab)c(ac) \rangle$ be a sequence. The set $\phi_{\leq 2}(s^1)$ of subsequences of $s^1 = \langle (ab) \rangle$ with a norm less than 2 is defined by $\phi_{\leq 2}(s^1) = \{\langle\ \rangle, \langle a \rangle, \langle b \rangle, \langle (ab) \rangle\}$. Thus, we have $\Phi_{\leq 2}(s^1) = 4$. It is also easy to see that $\Phi_{\leq 1}(s^1) = 3$ (the number of subsequences of $\langle (ab) \rangle$ having a norm less than 1). As $L(s^1, s[2]) = \emptyset$ (see Example 4), 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^2 obtained by concatenating the empty set to the subsequences of s^1, while the second term corresponds to 3 subsequences in s^2 obtained by concatenating the itemset (c) to each subsequence of s^1 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 row by row 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

Table 4. 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

to note that $m = min(K) \leq i - 1 < i$. Thus, at line 11, it ensures that only previously calculated terms $T[m-1][j-k]$ of T are used to calculate $R[i][j]$. 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 4 for the 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: $T[0][j] := 1$ and $T[1][j] := T[1][j-1] + \binom{|s[1]|}{j}$
4: **end for**
5: **for** $i = 2$ to $|s|$ **do**
6: **for** $j = 1$ to M **do**
7: $R[i][j] := T[i][j] := 0$
8: **for all** $K \in \mathcal{P}_{\geq 1}(L(s^{i-1}, s[i]))$ **do**
9: $m := min(K)$ and $k_{max} := |s[K] \cap s[i]|$
10: **for** $k = 1$ to j **do**
11: $R[i][j] \mathrel{+}= (-1)^{|K|+1} T[m-1][j-k] \times \binom{k_{max}}{k}$
12: **end for**
13: **end for**
14: **for** $k = 0$ to $min\{j, |s[i]|\}$ **do**
15: $T[i][j] \mathrel{+}= T[i-1][j-k] \times \binom{|s[i]|}{k}$
16: **end for**
17: $T[i][j] := T[i][j] - R[i][j]$
18: **end for**
19: **end for**
20: **return**(T)

4.3. Step 1: Sequence sampling

This first step aims to draw a sequence with a probability proportional to its weight $w(s)$. Of course, it is not possible to calculate this weight for each sequence s by naively summing one by one the weight of each subsequence $s' \sqsubseteq s$ as suggested by line 1 of Algorithm 1. In fact, it is more efficient to break down this weight according to the norm of the subsequences by observing that $w(s) = \sum_{\ell=0}^{\|s\|} w_\ell(s)$. Indeed, as all subsequences of the same norm have the same utility, it is sufficient to count the number of subsequences of norm ℓ and to multiply this quantity by the utility $f_u(\ell)$. The following property formalizes this intuition:

Property 2 (Computing $w_\ell(s)$). Given a sequence s and a norm-based utilitly u, the weight of subsequences having ℓ as norm, in s is defined by:

$$w_\ell(s) = \sum_{s' \sqsubseteq s \wedge \|s'\| = \ell} u(s') = \Phi_\ell(s) \times f_u(\ell)$$

Proof. Using Definition 3, we have $\sum_{s' \sqsubseteq s \wedge \|s'\|=\ell} u(s') = \sum_{s' \sqsubseteq s \wedge \|s'\|=\ell} f_u(\|s'\|)$ $= \sum_{s' \sqsubseteq s \wedge \|s'\|=\ell} f_u(\ell) = \left(\sum_{s' \sqsubseteq s \wedge \|s'\|=\ell} 1\right) \times f_u(\ell)$. We conclude that Property 2 is correct because $\sum_{s' \sqsubseteq s \wedge \|s'\|=\ell} 1$ is exactly the number of distinct subsequences of norm ℓ in s. □

This property strongly relies on Theorem 1 providing the number of subsequences of norm ℓ i.e., $\Phi_\ell(s) = \Phi_{\leq \ell}(s) - \Phi_{\leq \ell-1}(s)$. Thus, the formula of Property 2 makes it possible to calculate the initial weight $w(s) = \sum_{\ell=0}^{\|s\|} w_\ell(s)$ for each sequence s of the sequential database $\mathcal{S}$ (see line 1 of Algorithm 1).

Example 6. Let us come back on the example of Table 2. As $f_{u_{freq}}(\ell) = 1$ for all $l \in \mathbb{N}$, the weight $w_{freq}(s_i)$ equals to the number of distinct subsequences $\Phi_\ell(s_i)$. We now detail the calculation for the first sequence s_1 with three other utilities:

- With $f_{u_{area}}(\ell) = l$, we have $w_{area}(s_1) = \Phi_0(s_1) \times 0 + \Phi_1(s_1) \times 1 + \Phi_2(s_1) \times 2 + \Phi_0(s_3) \times 3 = 1 \cdot 0 + 3 \cdot 1 + 3 \cdot 2 + 1 \cdot 3 = 12$.
- With $f_{u_{\in[1..2]}}(\ell) = 1$ iff $l \in [1..2]$ (0 otherwise), we have $w_{\in[1..2]}(s_1) = \Phi_0(s_1) \times 0 + \Phi_1(s_1) \times 1 + \Phi_2(s_1) \times 1 + \Phi_0(s_3) \times 0 = 1 \cdot 0 + 3 \cdot 1 + 3 \cdot 2 + 1 \cdot 0 = 6$.
- With $f_{u_{decay}}(\ell) = \alpha^l$ and $\alpha = 0.5$, we have $w_{decay}(s_1) = \Phi_0(s_1) \times 0.5^0 + \Phi_1(s_1) \times 0.5^1 + \Phi_2(s_1) \times 0.5^2 + \Phi_0(s_3) \times 0.5^3 = 1 \cdot 1 + 3 \cdot 0.5 + 3 \cdot 0.25 + 1 \cdot 0.075 = 3.375$.

Computing the number of distinct subsequences smaller than a given length ℓ is expensive as we will see in Section 4.5. Fortunately, with the norm-based utility $u_{\leq M}$ (that we recommend to use for dealing with the long tail), we have $f_{u_{\leq M}}(\ell) = 0$ for all $\ell > M$ and consequently, $w(s) = \sum_{\ell=0}^{\|s\|} w_\ell(s) = \sum_{\ell=0}^{M} w_\ell(s)$.

4.4. Step 2: 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 ℓ between 0 and $\|s\|$ according to the distribution $w_\ell(s)$ (line 4 of Algorithm 1), NUSSAMPLING aims at returning a subsequence of norm ℓ drawn uniformly from the sequence s (line 5 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 7. 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 3:

Property 3. 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 with 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. $o[i_k] \subseteq s[l]$. $\square$

Thanks to Property 3, it is finally easy to draw uniformly a subsequence of norm ℓ in a sequence s. By randomly drawing ℓ distinct item positions between 1 and $\|s\|$, we start by uniformly drawing an occurrence containing ℓ 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 NUSSAMPLING 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 8. 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.

4.5. Theoretical analysis of the method

This property states that NUSSAMPLING returns an exact sample of subsequences with norm-based utility:

Property 4 (Soundness). Let $\mathcal{S}$ be a sequential dataset and u be a norm-based utility, NUSSAMPLING draws a subsequence of $\mathcal{S}$ according to a distribution proportional to frequency times its norm-based utility.

Proof. Let Z be the normalizing constant defined by $Z = \sum_{s\in\mathcal{S}} w(s)$. Let t be a subsequence in $\mathbb{S}$ 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 $P(s) = \frac{w(s)}{Z}$. Then, considering the third and fourth lines of Algorithm 1, if t is a subsequence of norm k, we have $P(t/s) = P(k/s) \times P(t/k, s) = \frac{\Phi_k(s)\times f_u(s)}{w(s)} \times \frac{1}{\Phi_k(s)} = \frac{f_u(s)}{w(s)}$. Thus, we have $P(t) = \sum_{s\in\mathcal{S},t\sqsubseteq s} P(s) \times P(t/s) = \sum_{s\in\mathcal{S},t\sqsubseteq s} \frac{w(s)}{Z} \times \frac{f_u(s)}{w(s)} = \frac{freq(s,\mathcal{S})\times f_u(s)}{Z}$, which shows that t is drawn proportionnaly to its frequency times its norm-based utility 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.

Preprocessing complexity The preprocessing is performed in time $O(|\mathcal{S}| \cdot L \cdot M^2 \cdot 2^P \cdot T)$ where L is the maximum length of a sequence, M is the maximum norm of drawn subsequences (if there is no norm constraint M equal to the maximum norm of the sequences in $\mathcal{S}$), 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 easy to see 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 $O(|\mathcal{S}| \cdot L \cdot M^2 \cdot T)$.

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 $\mu_u(\mathcal{S})$, required to derive a subsequence of $\mathcal{S}$ having a norm-based utility u. Intuitively, $\mu_u(\mathcal{S})$ depends both on the probability that a sequence $s \in \mathcal{S}$ is drawn and the average number of draws, denoted by $\mu_u(s)$,

required to find a first occurrence of a subsequence of s. The following property shows how these terms can be calculated:

Property 5 (Average number of draws). Given a norm-based utility u, the average number of draws for the acceptance of a subsequence in the sequential dataset $\mathcal{S}$ is defined by: $\mu_u(\mathcal{S}) = \sum_{s \in \mathcal{S}} \frac{w(s)}{\sum_{s' \in \mathcal{S}} w(s')} \times \mu_u(s)$ where $\mu_u(s) = \frac{\sum_{k=0}^{\|s\|} \binom{\|s\|}{k} \times f_u(k)}{w(s)}$ and $w(s) = \sum_{k=0}^{\|s\|} \Phi_k(s) \times f_u(k)$.

Proof. Using Algorithm 1, it is clear that $\mu_u(\mathcal{S}) = \sum_{s \in \mathcal{S}} P(s) \times \mu_u(s)$ with $P(s) = \frac{w(s)}{\sum_{s' \in \mathcal{S}} w(s')}$. Then, we have $\mu_u(s) = \sum_{k \in [0..\|s\|]} P(k/s) \times N_k(s)$ 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 $P_a^k(s) = \frac{\Phi_k(s)}{\binom{\|s\|}{k}}$. 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 $\mu_u(s) = \sum_{k \in [0..\|s\|]} P(k/s) \times N_k(s) = \sum_{k \in [0..\|s\|]} \frac{w_k(s)}{w(s)} \times \frac{\binom{\|s\|}{k}}{\Phi_k(s)}$. As $w_k(s) = \Phi_k(s) \times f_u(k)$, we finally obtain $\mu_u(s) = \frac{\sum_{k \in [0..\|s\|]} \binom{\|s\|}{k} \times f_u(k)}{w(s)}$. □

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, meaning that 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 in a sequence s is $O(\|s\|^2)$ in the worst case, the average complexity of drawing N subsequences from a dataset $\mathcal{S}$ (after the preprocessing phase) is in $O(N \cdot M^2 \cdot \mu_u(\mathcal{S}))$.

5. Experimental Study

This experimental study aims to evaluate the efficiency of our approach and the interest of the sampled subsequences using different norm-based utilities. More precisely, we consider the three below interestingness measures involving norm-based utilities:

- M-**frequency** combines the frequency measure with the norm-based utility constraint defined for every sequence s by $u_{\geq 1}(s) \times u_{\leq M}(s)$,
- M-**area** combines the frequency measure with the utility function defined for every sequence s by $u_{area}(s) \times u_{\geq 1}(s) \times u_{\leq M}(s)$, and
- α-**frequency** combines the frequency measure with the exponential decay utility $u_{decay}(s) = \alpha^{\|s\|}$.

Section 5.1 focuses on the speed of NUSSAMPLING and its ability to draw patterns that do not belong to the long tail. In particular, we compare the impact of using norm-based utility contraint $u_{\leq M}$ or exponential decay utility u_{decay}. Section 5.2 compares the sampled sequential patterns returned by NUSSAMPLING in the context of sequence classification, where the accuracy performs an objective measure.

Table 5. Statistics of benchmark datasets

Dataset	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{I}\rvert$	$\lVert S\rVert_{max}$	$\lVert S\rVert_{mean}$	P	T	$\lvert\mathcal{C}\rvert$
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	–
aslbu	441	132	27	7.5	1	1	7
aslgt	3,493	87	88	22.8	1	1	40
auslan	200	12	24	10.0	1	1	10
blocks	210	8	12	6.7	1	1	8
context	240	48	123	45.2	1	1	5
pioneer	160	92	50	21.1	1	1	3
skater	530	41	120	25.1	1	1	6
speed	530	41	260	64.5	1	1	7
reuters	5,459	14,577	533	67.3	1	1	8

Note that 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 used experimental datasets, as well as source code, are available at `https://github.com/LDIOPBSF/NUSSampling` under the GPLv3 license.

5.1. Analysis of NUSSampling method

This experimental section evaluates the speed of our method and the impact of different norm-based utilities on the sampled patterns. For this purpose, we use 15 datasets including 11 real life datasets [2] and 4 synthetic datasets generated by IBM data generator [3]. The main interest of using synthetic datasets is to have examples of sequences with itemsets instead of sequences containing only items (i.e., with $T > 1$). Table 5 lists basic statistics of all datasets: the number of sequences, the number of items, the maximum/mean sequence norm, the maximum size of position sets P, the maximum size of itemsets T and the number of classes. Table 6 compares the average number of draws per subsequence required to extract a pattern when there is a contraint utility $u_{\geq 1} \times u_{\leq M}$ with $M \in \{1, 2, 3, 5, 7\}$. Table 6 is obtained using Property 5. It shows that `bms` and `sign` datasets do not contain multiple occurrences within a same subsequence, while the number of multiple occurrences increases with M for the other datasets.

5.1.1. Preprocessing and sampling speed

In this section, we analyze the preprocessing and sampling speed of NUSSampling considering three different interestingness measures M-frequency, M-area and α-frequency.

Norm constraint Figure 2 plots the average execution time of our method by distinguishing the preprocessing time (left-hand side) and the average number of draws of a sequential pattern (right-hand side) using M-frequency measure with $M \in \{1, 2, 3, 5, 7\}$. Note that we do not report these results for M-area measure

[2] The datasets `bms` and `sign` are available at `http://www.philippe-fournier-viger.com/spmf` and other ones at `http://www.mybytes.de/#data`

[3] `https://github.com/zakimjz/IBMGenerator`

Table 6. 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
aslbu	1.2	1.3	1.3	1.4	1.5
aslgt	1.4	1.9	2.8	5.8	12.6
auslan	2.6	3.5	4.8	8.2	10.4
blocks	1.4	1.9	2.4	3.7	4.5
context	2.5	6.3	17.5	159.5	1,652.7
pioneer	1.1	1.3	1.5	2.0	2.6
skater	1.7	3.3	8.3	81.3	837.0
speed	3.3	7.0	15.9	99.4	554.2
reuters	1.6	2.4	3.4	5.9	8.6

because it has exactly the same behavior as M-frequency for all experiments. 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 (less than 80 seconds). Regarding the sampling phase, whatever the dataset, the measure and the maximum norm M, the execution time is always under 1.5 millisecond, despite an average number of draws $\mu_{[m,M]}(\mathcal{S})$ greater than 1 (and hence, rejection).

Exponential decay Figure 3 plots the preprocessing (left-hand side) and sampling (right-hand side) time of our method using the α-frequency measure with different values of $\alpha \in \{0.05, 0.1, 0.2, 0.3, 0.4, 0.5\}$. Given a dataset, we first see that the preprocessing time remains constant with α. Indeed, the computation of the weight of a sequence in a dataset is clearly independent of α, i.e. without constraint, for each sequence $s \in \mathcal{S}$, we just have to compute the number of distinct subsequences of s. Conversely, the right-hand side of Figure 3 shows that the sampling speed varies with α. When α increases, the probability to draw a subsequence with a higher norm increases, and the computation time to draw an occurrence of a subsequence increases with its norm. However, when we draw an occurrence, the probability that it is not a first occurrence (i.e., rejection) decreases with its norm. This explains why the sampling time starts to increase with α, and then, decreases. Finally, it is interesting to see that both the preprocessing and sampling time are lower when we use a norm constraint. More precisely, by comparing Figure 2 with Figure 3, we observe that they are at least two times lower when a maximum norm constraint $u_{\leq M}$ is used with $M \in [1..7]$.

5.1.2. Distribution of sampled patterns

We now consider the benefit of using a norm-based utility function to limit the effect of the long tail.

Norm constraint We first compare the distribution of the sampled pattern using the frequency and area measures with or without using a maximum norm constraint $u_{\leq M}$ (with $M \in \{4, 7\}$). Figures 4 and 5 represent the distribution of $10,000$ sequential patterns sampled according to the M-frequency and M-

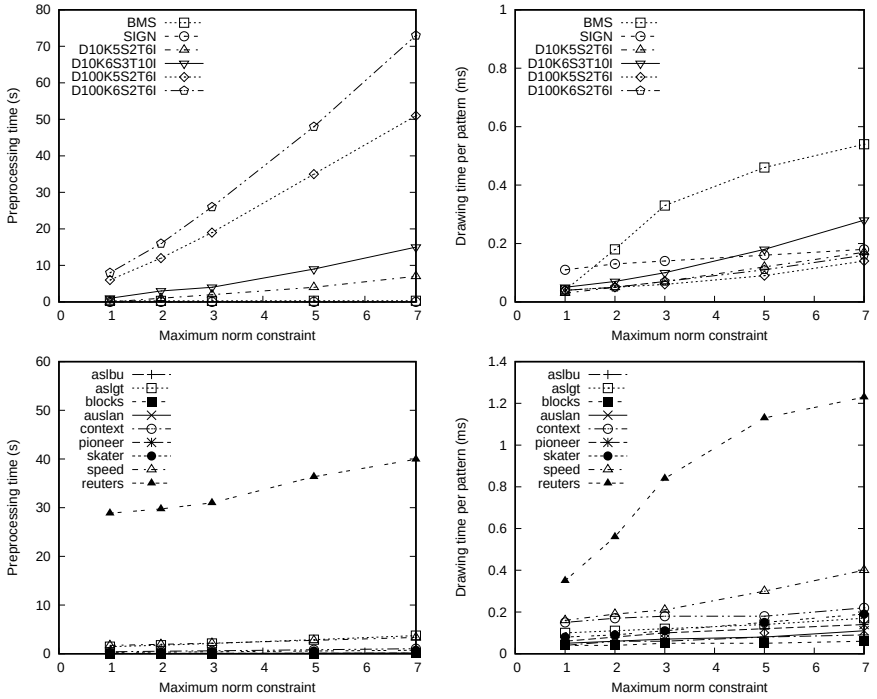

Fig. 2. Execution time for sequential pattern sampling with M-frequency

area measures for different datasets, with or without constraint. For all datasets, without constraint, the sampling method returns only patterns with very low interestingness. With the frequency measure, we notice in Figure 4 that without constraint, most of the sampled sequences have a frequency equal to 1. More precisely, with the area measure, we also see that in most of the cases, without constraint, the sampling method returns sequences with very low areas. Note that this is not the case with `bms` dataset, because this dataset contains a very long sequence, and that we mainly sample subsequences of this sequence. Thus, even if the frequency of these subsequences is very low, the mean of their norm is large. Conversely, with a maximum norm constraint, the sampling method returns sequential patterns with significantly higher frequency or areas, which shows the importance of introducing constraints on the norm to avoid the problem of the long tail. 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.

Exponential decay Figure 6 shows the distribution of $10,000$ sequential patterns sampled according to the α-frequency measure with different values of α. When α equals to 1.0, it means that there is no exponential decay filtering. Consequently, the results are as bad as those obtained without maximum norm constraint. Besides, it is easy to see that when α decreases, the frequency of the drawn sequential patterns increases as it was the case when M increases. However, the drawn subsequences do not have exactly the same form as those

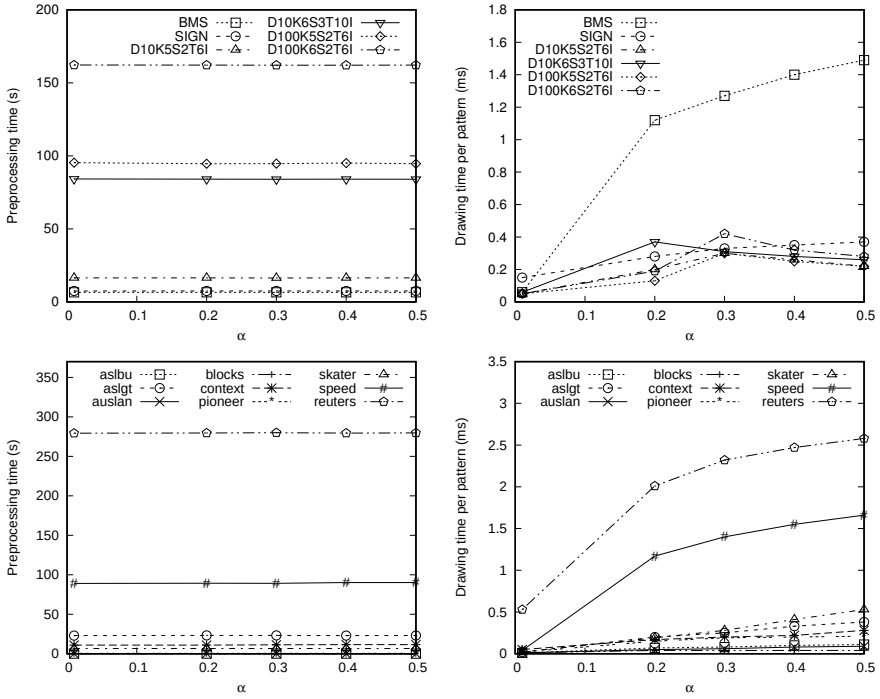

Fig. 3. Execution time for sequential pattern sampling with α-frequency

obtained with norm constraints. Indeed, Figure 7 compares the distribution of 10,000 sampled patterns with respect to their norm considering M-frequency, M-area ($M \in \{4, 7\}$) and α-frequency ($\alpha \in \{0.10, 0.05, 0.01\}$) as interestingness measures for two sequential datasets (`sign` and `D10K6S3T10I`). First, we see on the charts of the first two rows that with M-frequency and M-area, most of the sampled patterns have a norm equals to the maximum norm constraint M. By comparison, we observe in Figure 7 (last row) that with the α-frequency measure, the norm diversity of the sampled subsequences is higher when α is not too low (i.e., 0.10 or 0.05). If the value of α is low (i.e., 0.01 in our experiments), we only obtain subsequences with very low norms between 1 and 4.

5.2. Accuracy of sampling-based classification

This section shows how sampled subsequences can be used to build associative classifiers dedicated to sequences and the benefit to use norm constraints or exponential decays to obtain better classification models. Our classification method, called NUSSAMPLING+SVM, is a standard approach relying on two phases as done in [37]:

1. **Feature extraction:** In a first phase, using NUSSAMPLING, we build a sample $F = \{f_1, \ldots, f_K\}$ of K subsequences. Then, this sample is used to transform a training dataset $\mathcal{S}$ that is a labeled sequential dataset into a binary dataset $\mathcal{D}$. More precisely, for each sequence $s_i \in \mathcal{S}$ labeled by a class c_i, $\mathcal{D}$

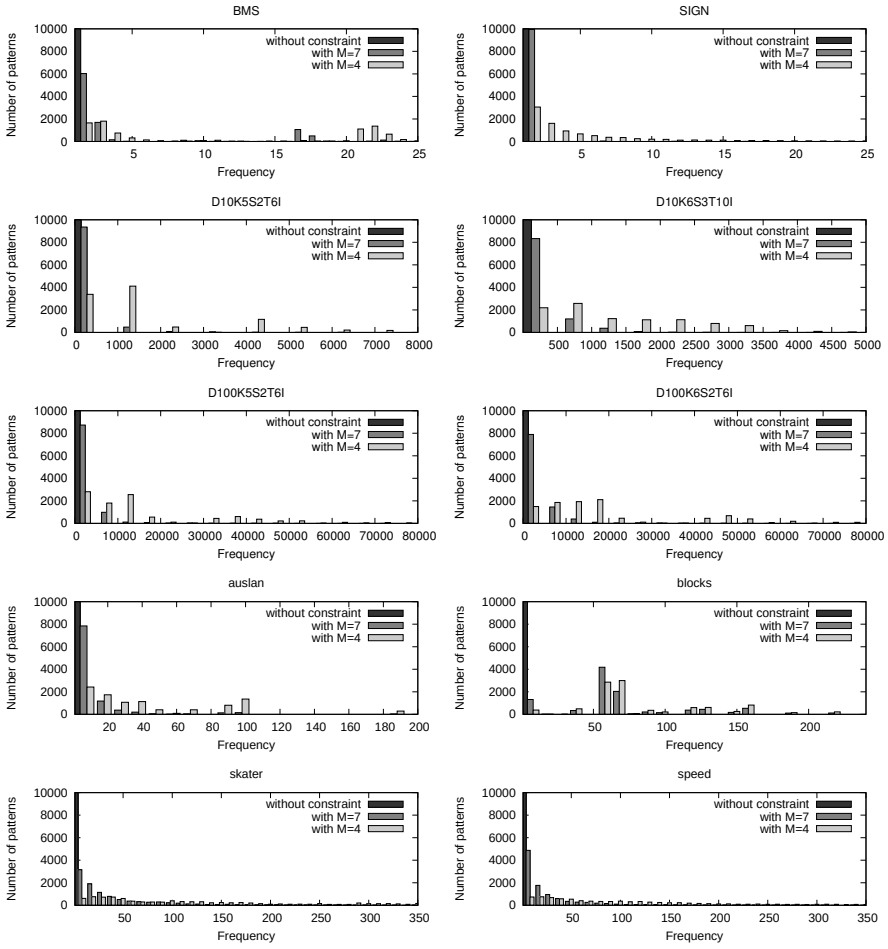

Fig. 4. Distribution of 10,000 sequential patterns according to M-frequency

contains a tuple t_i of $K+1$ values where $t_i[j] = 1$ if $f_j \sqsubseteq s_i$ (0 otherwise) for $j \in [1..K]$, and $t_i[K+1] = c_i$. Figure 8 illustrates this principle where a labeled sequential dataset with 4 sequences in class A or B is transformed into a binary dataset using a sample of $K = 3$ subsequences. For example, considering the third sampled subsequence $\langle ac \rangle$, we have $t_1[3] = 1$ because the sequence s_1 contains the subsequence $\langle ac \rangle$, whereas $t_3[3] = 0$ because the sequence s_3 does not contain the subsequence $\langle ac \rangle$.

2. **Model construction:** Then, in a second phase, we create a SVM classifier C from the transformed training dataset. Note that in our experiments, we use the SMO algorithm provided by Weka 3.8 and its default options to build SVM classifiers.

After the model construction, in order to predict the class of an unlabeled sequence, we first transform this sequence into a binary vector using the sample

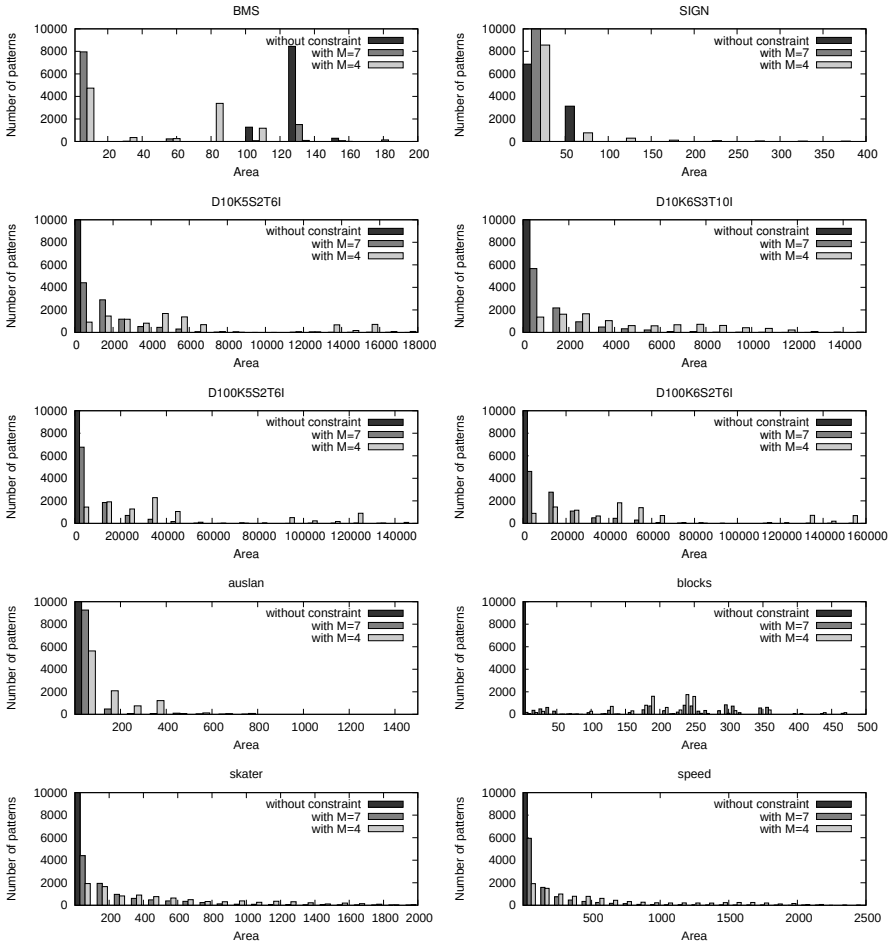

Fig. 5. Distribution of 10,000 sequential patterns according to M-area

F of K subsequences. Then, we use the SVM classifier C to predict the class of the sequence previously transformed into a binary vector.

We evaluate the efficiency of NUSSAMPLING+SVM on real-world datasets [37][4] that have a wide variety in the number of sequences, items, sequence lengths and classes as well as application domains (see Table 5). For each dataset, we calculate the accuracy of NUSSAMPLING+SVM with respect to various sample sizes and norm constraints, by performing a 10-fold cross-validation.

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

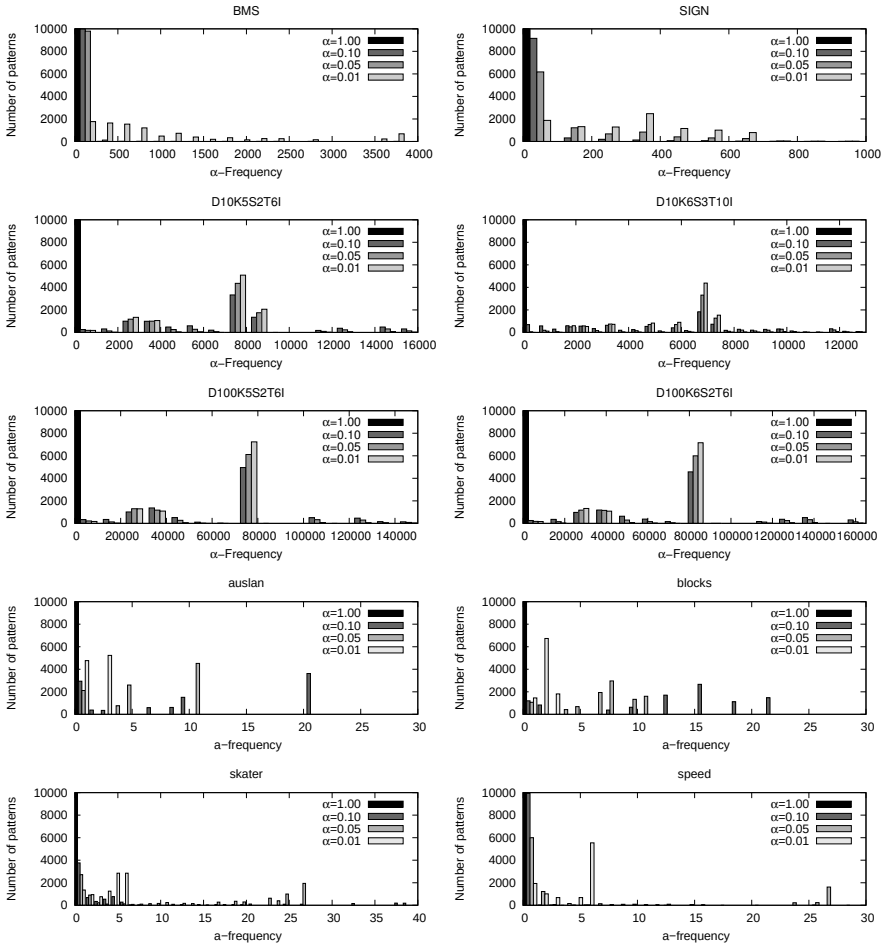

Fig. 6. Distribution of 10,000 sequential patterns according to α-frequency

5.2.1. Importance of the norm constraint and exponential decay

As described in the previous sections, the norm-based utility constraint $u_{\leq M}$ and u_{decay} can be used to limit the maximal length of sampled subsequences. In this section, we evaluate the impact of using these norm-based utilities to improve the performance of our classifiers. More precisely, we first build classifiers using subsequences sampled with M-frequency or M-area measures considering different values of $M \in [1..10]$. Then, we also build classifiers using subsequences sampled with the α-frequency measure considering values of $\alpha \in \{0.01, 0.025, 0.05, 0.075, 0.1, 0.5\}$. Note that we reduce the preprocessing time for the exponential decay utility by adding a maximum norm constraint $u_{\leq 10}$. This norm constraint has no effect on the sampled subsequences because even if no constraint was considered, the probability of drawing a subsequence with a norm greater than 10 would be almost zero with $\alpha \leq 0.5$.

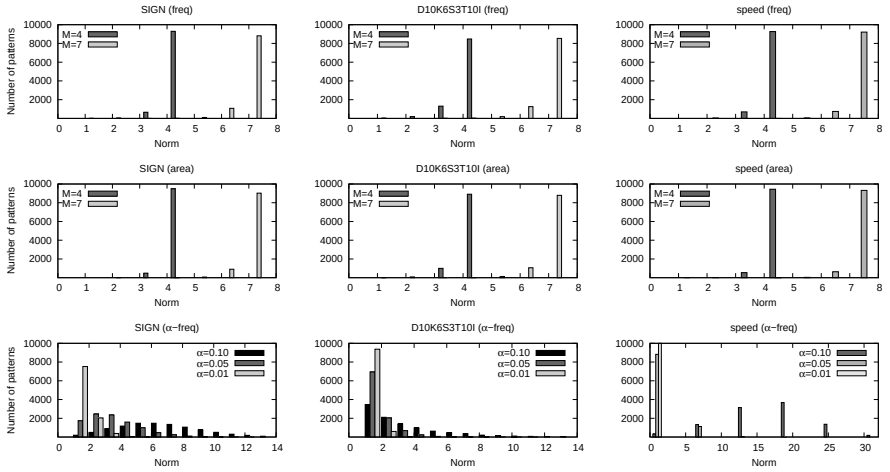

Fig. 7. Distribution of 10,000 sequential patterns according to norm with different norm-based utilities

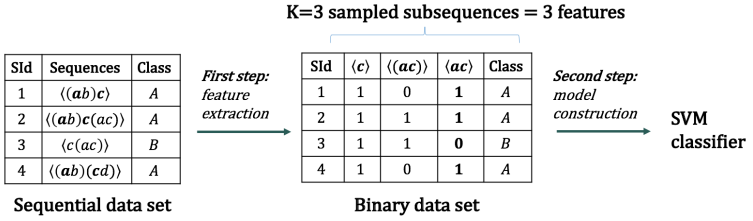

Fig. 8. NUSSampling+SVM classification method

For all experiments and each dataset, we use a different paired Student's t-test (with a confidence level $\beta = 95\%$) to evaluate if the best mean accuracy (obtained with the optimal value of M or α) is significantly different from the other mean accuracies (obtained with non-optimal values of parameters M or α). Indeed, the optimal values of parameters M or α are not necessarily the same for each dataset.

Table 7. Impact of the norm constraint using M-frequency

Dataset	M=1	M=2	M=3	M=5	M=7	M=10	**Best**
aslbu	**0.649***	0.601	**0.608**	0.539	0.396	0.373	0.649
aslgt	0.668	**0.688***	**0.680**	0.634	0.505	0.364	0.688
auslan	0.230	0.250	**0.320**	**0.320**	**0.330***	**0.330***	0.330
blocks	0.857	**1.000***	**0.995**	**0.995**	**0.995**	**0.995**	1.000
context	**0.984***	**0.984**	**0.971**	**0.975**	**0.967**	**0.959**	0.984
pioneer	**1.000***	**0.975**	**0.969**	0.858	0.691	0.656	1.000
skater	0.883	**0.930**	**0.944***	0.919	0.889	0.874	0.944
speed	0.257	0.281	0.306	**0.366***	**0.326**	0.301	0.366
reuters	**0.949***	0.901	0.765	0.531	0.523	0.519	0.949
Average	0.720	**0.734**	0.729	0.682	0.625	0.597	0.734

Table 8. Impact of the norm constraint using M-area

Dataset	M=1	M=2	M=3	M=5	M=7	M=10	**Best**
`aslbu`	**0.649***	**0.623**	0.588	0.452	0.370	0.375	0.649
`aslgt`	0.668	**0.688***	0.680	0.634	0.505	0.359	0.688
`auslan`	0.250	0.255	**0.355***	**0.350**	**0.345**	**0.325**	0.355
`blocks`	0.857	**1.000***	**0.995**	**0.990**	**0.995**	**0.991**	1.000
`context`	**0.964**	**0.984***	**0.966**	**0.983**	**0.971**	**0.971**	0.984
`pioneer`	**0.994***	**0.975**	0.962	0.801	0.701	0.645	0.994
`skater`	0.887	0.930	**0.945***	**0.925**	0.866	0.805	0.945
`speed`	0.266	0.273	0.339	**0.371***	**0.329**	0.272	0.371
`reuters`	**0.952***	0.904	0.545	0.522	0.520	0.518	0.952
Average	0.721	**0.737**	0.708	0.670	0.622	0.584	0.737

Norm constraint Considering the M-frequency or M-area measures,Tables 7 and 8 show that the accuracy of NUSSampling+SVM clearly depends on the norm constraint. The best mean accuracy is marked for each dataset with a star, whereas the comparable accuracies (identified with a paired Student's t-test) are in bold. While the total size of sample is fixed (here, 10,000 patterns), we can see that the best classification performance is generally obtained when the maximum norm threshold is strictly larger than 1 and lower that 10. However, this result is less significant when the classification problem to be solved is rather simple (for accuracy greater than 95%), i.e. for the datasets `blocks`, `context`, `pioneer` and `reuters`. For these datasets, a good performance can generally be obtained with a maximum norm constraint M equal to 1 (i.e., the features are subsequences with only one item). For the other datasets, the performance of classifiers decreases with M when M is greater than its optimal value. For this reason, it is important 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.

Exponential decay For the α-frequency measure, Table 9 also shows that the performance of NUSSampling+SVM clearly depends on the value of the parameter α. First, we can observe that if the value of α is too high (i.e., $\alpha = 0.5$ or above), then the performance of the classifiers is generally not satisfactory. This phenomenon can be easily explained because when the value of α is too high, we mainly sample large subsequences with very low frequencies. Then, compared with the search of the optimal maximum norm constraint M, we might think that finding the optimal value of α is more difficult because it is a real parameter. However, we can observe in Table 9 that the accuracy is not very sensitive to the value of α between 0.01 and 0.1.

5.2.2. Pattern Sampling vs Top-k Pattern Mining method with norm constraints

This section compares NUSSampling+SVM with M-frequency with the TKS algorithm [38], which returns the k most frequent sequential patterns having a norm smaller than M. Interestingly, these two approaches use a parameter k to control the number of mined patterns and a parameter M to limit their size. To build a classifier, we use TKS for extracting features from the same dataset as NUSSampling+SVM. Then, as with our approach, we use the SMO algorithm

Table 9. Impact of the exponential decay using α-frequency

Dataset	α=0.010	α=0.025	α=0.050	α=0.075	α=0.100	α=0.500	**Best**
aslbu	**0.635***	0.602	**0.618**	**0.618**	**0.611**	**0.592**	0.635
aslgt	**0.683***	0.681	**0.683**	**0.679**	**0.682**	0.471	0.683
auslan	0.265	**0.335**	**0.335**	**0.355***	**0.345**	**0.325**	0.355
blocks	**0.995**	**0.995**	**0.990**	**0.995**	**1.000***	**0.995**	1.000
context	**0.987***	**0.983**	**0.975**	**0.975**	**0.975**	**0.975**	0.987
pioneer	**0.994***	**0.994***	**0.981**	**0.975**	**0.987**	0.688	0.994
skater	0.925	**0.934**	**0.947***	**0.940**	**0.943**	0.843	0.947
speed	0.289	**0.319**	**0.345**	**0.347***	**0.328**	0.260	0.347
reuters	**0.950***	0.899	0.544	0.520	0.519	0.519	0.950
Average	0.747	**0.749**	0.713	0.712	0.710	0.630	0.749

Table 10. Comparison of accuracy between NUSSampling (M-freq) and TKS

	$NUSSampling(M_{freq})$		TKS	
Dataset	Optimal M	Best accuracy	Optimal M	Best accuracy
aslbu	M=1	**0.649***	M=1	**0.623**
aslgt	M=2	**0.688***	M=2	0.659
auslan	M=7	**0.330***	M=1	0.230
blocks	M=2	**1.0***	M=1	**0.976**
context	M=1	**0.984***	M=1	**0.980**
pioneer	M=1	**1.0***	M=1	**0.994**
skater	M=3	**0.944***	M=1	0.870
speed	M=5	**0.366***	M=1	0.249
reuters	M=1	**0.950***	M=1	**0.950***

to build a SVM classifier. The classification approaches are therefore extremely comparable because only the mined features differ.

Table 10 reports the accuracy of our approach compared with those of TKS on all classification datasets. More precisely, for both approaches, using cross-validation, we select the optimal value of the parameter M (M varying between 1 and 7) and report the best obtained accuracy. Note that for TKS the optimal value of the parameter M is always 1 except on aslgt. Then, using a paired Student's t-test (with a confidence level $\beta = 95\%$), we evaluate if the best accuracies obtained with NUSSampling+SVM and TKS are comparable or not.

First, we note that NUSSampling+SVM always has a better accuracy than TKS (except on reuters where the best accuracies are equal). Then, we observe that the accuracy gap between the two methods is generally higher when the best accuracy of NUSSampling+SVM is for a value M greater than 1 (except on blocks). In this case, a good classification requires complex features based on combinations of items. This feature space then becomes larger and pattern sampling better covers this feature space than TKS. For this reason, in 4 datasets where the best accuracy requires to set $M > 1$ (aslgt, auslan, skater and speed), the accuracy of NUSSampling+SVM is significantly better than the accuracy obtained by TKS.

5.2.3. Comparison with pattern-based sequence classification methods

We finally compare the accuracy of NUSSampling+SVM with the results of 7 state-of-the-art sequence classification methods reported in [37] as baselines with respect to the same datasets: MiSeRe, SQS, GoKrimp, cSPADE, SCII and DeFFeD. Using the three measures M-frequency, M-area and α-frequency, Figure 9 shows that the best accuracies obtained by NUSSampling+SVM (col-

umn **Best** of Tables 7, 8 and 9) are comparable to other pattern-based sequence classification methods reported in [37], even better for datasets `auslan`, `context`, `speed` and `skater`.

To compare more precisely the different methods, we apply the Friedman test and a post-hoc Nemenyi test as suggested by [39] for comparisons of classifiers over multiple datasets (with a confidence level $\beta = 95\%$ for all tests). First, the Friedman test is used to evaluate if the measured average ranks of the different classifiers are significantly different from the mean rank expected under the null-hypothesis. In our case, since $F_F = 11.06$ is greater than the critical value 2.01 (obtained for $\beta = 95\%$), we reject the null-hypothesis that all methods are comparable. Then, we use the Nemenyi test for pairwise comparisons. With $p = 0.05$, the Critical Difference CD is equal to 4.52. Thus, we can distinguish in Figure 10 two groups of methods: NUSSAMPLING+SVM, MISERE, CSPADE and SQS build better classifiers than SCII, GOKRIMP and DEFFED.

Thus, even if the goal of this paper is not to propose a new sequence classification method, these experiments show how subsequence sampling can be used to build classifiers, and that our NUSSAMPLING+SVM method is competitive with the best methods of the literature. Finally, it is important to recall that the complexity of our sampling method is very low (only linear with $|\mathcal{S}|$ during the preprocessing step, and logarithmic with $|\mathcal{S}|$ during the drawing phase), which means that our method could be used with larger datasets than the datasets in our benchmark.

5.2.4. 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. In this section, in order to evaluate the impact of the sample size, we only perform experiments using the M-frequency measure. In this context, Figure 11 shows the classification 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 7. It is easy to observe that the classification performance increases continuously when 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.

6. Discussions and Possible Extensions

This section compares the expressivity of our class of measures with that of the literature in Section 6.1. Then, we extend our class of measures by considering the squared frequency and the discriminativity introduced in Section 6.2. For this purpose, Section 6.3 shows how to calculate the number of common distinct subsequences between two sequences and Section 6.4 analyzes its complexity.

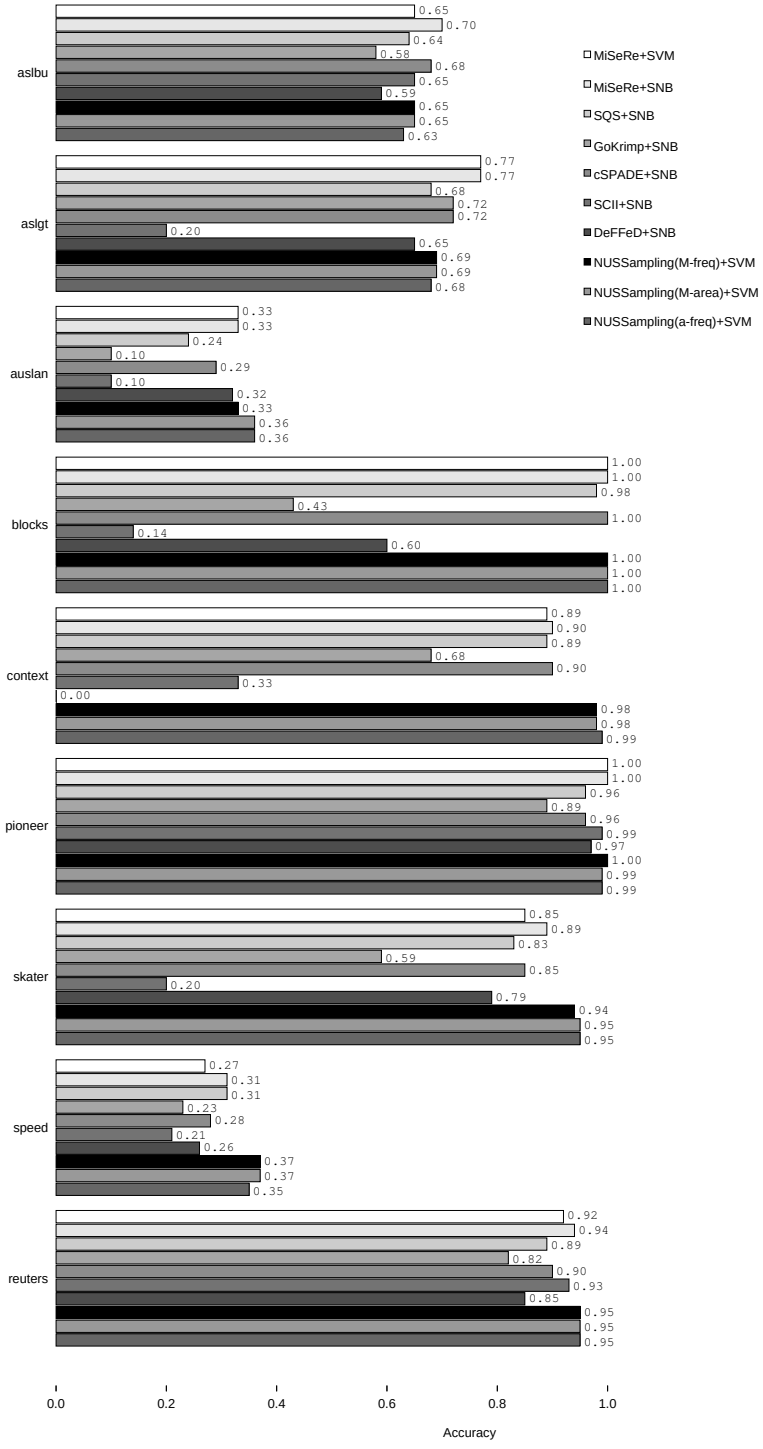

Fig. 9. Comparison of accuracy results between NUSSAMPLING with SVM and state-of-the-art sequence classification methods.

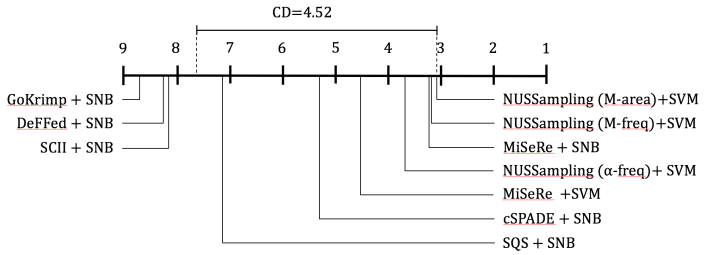

Fig. 10. Critical difference of performance between different classifiers.

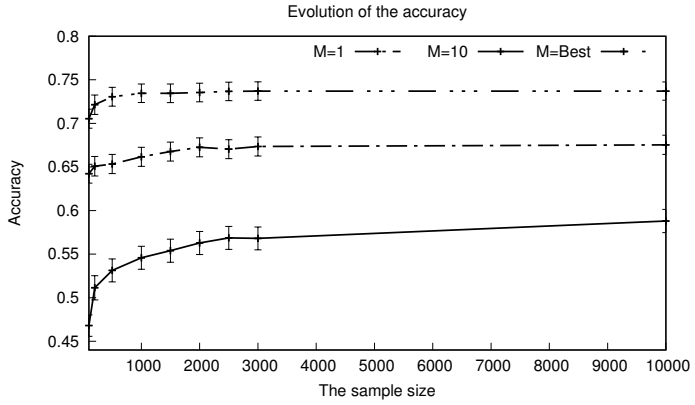

Fig. 11. Impact of the sample size on classification performance with M-frequency

6.1. Different classes of measures

In [19, 26], the authors show how to consider a very large set of interestingness measures to draw itemsets in a transactional dataset. More precisely, given an itemset X and a transactional dataset $\mathcal{S}$, they consider the set of all measures of the form $f(X, \mathcal{S}) = u_\star(X) \times \pi_{i=1}^{K} q_i(X, \mathcal{S}_i)$ where $u_\star(X) = \bigstar_{x \in X} b(x)$ with $\bigstar \in \{\Pi, \sum\}$ and q_i is either the positive or negative frequency of X in a specific portion $\mathcal{S}_i$ of the input dataset $\mathcal{S}$.

In our approach, we only consider utility functions u that are norm-based, i.e. such that for all sequence s, $u(s) = f_u(\|s\|)$. Thus, by comparison with [19, 26], we cannot consider utility functions $u_\star(s)$ defined by $u_\star(s) = \Pi_{x \in s} b(x)$ where b is not a constant function (for any item $x \in \mathcal{I}$). On the other hand, it is easy to see that our utility functions u_{area} and u_{decay} can be considered by [19, 26]. Indeed, given an itemset X, we have $u_{area}(X) = \sum_{x \in X} b(x) = |X|$ if $b(X) = 1$ for every item $x \in \mathcal{I}$, and $u_{decay}(X) = \Pi_{x \in X} b(x) = \alpha^{|X|}$ if $b(X) = \alpha$ for every item $x \in \mathcal{I}$. However, the class of utility functions defined in [19, 26] cannot model a size constraint using a utility function u_M defined by $u_M(X) = 1$ if $|X| \leq M$ (0 otherwise). However, we saw in Section 5, the importance and the benefit of considering contraints on the norm of the drawn subsequences to limit the effect of the long tail.

Now, it is important to note that in our work, we only consider *linear* interestingness measures, i.e. that depends linearly on the frequency of the sequences. For example, compared to [19, 26], we do not consider *quadratic* measures such as the *squared frequency* and *discriminativity* measures. Nevertheless, we show in the next sections how our approach could be extended to this class of measures and the cost to consider this extensions (in term of complexity with respect to the size of the datasets).

6.2. Squared frequency and discriminativity

In our framework, given a sequential dataset $\mathcal{S}$, we can define the *weighted squared frequency* of a sequence s by $q_{sqf}(s) = u(s) \times freq(s, \mathcal{S})^2$ where u is a utility function. Similarly, given a positive sequential dataset $\mathcal{S}^+$ and a negative sequential dataset $\mathcal{S}^-$, the discriminativity of a sequence s can be defined by $q_{disc}(s) = u(s) \times freq(s, \mathcal{S}^+) \times (|\mathcal{S}^-| - freq(s, \mathcal{S}^-))$. Following the procedure proposed in [19, 26] for transactional datasets, it is rather direct to extend our approach to draw a sequence s with a probability proportional to its weighted square frequency $q_{sqf}(s)$ or weighted discriminativity $q_{disc}(s)$. For example, given a positive sequential dataset $\mathcal{S}^+$ and a negative sequential dataset $\mathcal{S}^-$, we can draw a subsequence w.r.t. its weighted discriminativity as follows:

1. First, we sample a pair of sequences (s^+, s^-) with a probability proportional to the weight $w_{disc}(s^+, s^-) = \sum_{k=0}^{\|s^+\|} w^k_{disc}(s^+, s^-)$ where $w^k_{disc}(s^+, s^-) = f_u(k) \times (\Phi_k(s^+) - \Phi_k(s^+, s^-))$ and $\Phi_k(s^+, s^-)$ is the number of common subsequences between s^+ and s^- with a norm equal to k.
2. Then, after drawing the norm k of the subsequence s that will be returned, we *uniformly* draw a subsequence of s^+ that is not a subsequence of s^- (with a norm equal to k). This second step can be done with a rejection method similar to the rejection method proposed in Section 4.4.

In order to implement this procedure, the main challenge of the first step is to count the number of common subsequences between two sequences. We show in Section 6.3 how this computation can be done by generalizing a formula presented in [36]. Considering the second step, it is interesting to evaluate the average number of draws necessary to return a subsequence. Given a pair of sequences (s^+, s^-), it is easy to see that the probability $P^k_a(s^+, s^-)$ that a subsequence s of s^+ is accepted is defined by: $P^k_a(s^+, s^-) = \frac{\Phi_k(s^+) - \Phi_k(s^+, s^-)}{\binom{\|s^+\|}{k}}$. Indeed, s will be accepted if it is a first occurrence in s^+ and not a subsequence in s^-. Thus, given a pair of sequences (s^+, s^-), the average number of draws to accept a subsequence of s^+ is $\mu_{disc}(s^+, s^-) = \sum_{k=0}^{\|s^+\|} \frac{w^k_{disc}(s^+, s^-)}{w_{disc}(s^+, s^-)} \times P^k_a(s^+, s^-)^{-1}$ $= \sum_{k=0}^{\|s^+\|} \frac{f_u(k) \times \binom{\|s^+\|}{k}}{w_{disc}(s^+, s^-)}$. It is clear that this number will be more important if the number of common subsequences between s^+ and s^- increases. Indeed, $w_{disc}(s^+, s^-)$ is decreasing with $\Phi_k(s^+, s^-)$. However, the probability to draw a pair of sequences (s^+, s^-), defined by $P(s^+, s-) = \frac{w_{disc}(s^+, s^-)}{Z_{disc}}$ where $Z_{disc} = \sum_{(s^+, s^-) \in \mathcal{S}^+ \times \mathcal{S}^-} w_{disc}(s^+, s^-)$, is decreasing with the number of common subsequences between s^+ and s^-.

6.3. Counting the common distinct subsequences between two sequences

In this Section 6.3, we show how to extend the results presented in Section 4.2 to count the number of common distinct subsequences between two sequences s_1 and s_2 under a norm constraint. This result is a generalization of the formula presented in [36] to count the number of common subsequences between two sequences *without* constraint on the norm. In order to count the number of common subsequences between two sequences $s_1 \circ Y$ and s_2 under a maximal norm constraint j, this new formula contains three terms. The first term $\Phi_{\leq j}(s_1, s_2)$ is simply the number of common subsequences already existing between s_1 and s_2. The term $A_{\leq j}(s_1, s_2, Y)$ represents the number of extra common subsequences that are added when a new itemset Y is concatenated to the sequence s_1. Finally, the term $R_{\leq j}(s_1, s_2, Y)$ is a correction term in order to take into account the repetitions that may occur (as for the counting method of the distinct subsequences of a sequence).

Theorem 2 (Number of common subsequences with a maximal norm). Let s_1 and s_2 be two sequences, Y be an itemset and j be an integer, the number of distinct common subsequences having a norm less or equal to j between $s_1 \circ Y$ and s_2, denoted by $\Phi_{\leq j}(s_1 \circ Y, s_2)$, is defined as follows:

$$\Phi_{\leq j}(s_1 \circ Y, s_2) = \Phi_{\leq j}(s_1, s_2) + A_{\leq j}(s_1, s_2, Y) - R_{\leq j}(s_1, s_2, Y)$$

where $A_{\leq j}(s_1, s_2, Y)$ and $R_{\leq j}(s_1, s_2, Y)$ are the terms defined by:

$$A_{\leq j}(s_1, s_2, Y) = \sum_{\emptyset \subset K \subseteq L(s_2, Y)} (-1)^{|K|+1} A^K_{\leq j}(s_1, s_2, Y)$$

with $A^K_{\leq j}(s_1, s_2, Y) = \sum_{k=1}^{j} \Phi_{\leq j-k}(s_1, s_2^{min(K)-1}) \times \binom{|s_2[K] \cap Y|}{k}$ and

$$R_{\leq j}(s_1, s_2, Y) = \sum_{\emptyset \subset K \subseteq L(s_1, Y)} (-1)^{|K|+1} \left(\sum_{\emptyset \subset K' \subseteq L(s_2, Y)} (-1)^{|K'|+1} R^{K,K'}_{\leq j}(s_1, s_2, Y) \right)$$

with $R^{K,K'}_{\leq j}(s, Y) = \sum_{k=1}^{j} \Phi_{\leq j-k}(s_1^{min(K)-1}, s_2^{min(K')-1}) \times \binom{|s_1[K] \cap s_2[K'] \cap Y|}{k}$ [5].

The following example illustrates the principles of the formula given by Theorem 2.

Example 9. In this example, we show how to compute the number of common subsequences between sequences $s_2 = \langle (ab)d(ac) \rangle$ and $s_4 = \langle (ab)(cd) \rangle$ under a maximum norm constraint $j = 2$. First, it is easy to see that $\Phi_{\leq 2}(s_2^0, s_4) = 1$ since the empty subsequence is the only common subsequence between $s_2^0 = \langle\rangle$ and s_4. Now, we show how to compute: $\Phi_{\leq 2}(s_2^1, s_4) = \Phi_{\leq 2}(s_2^0 \circ (ab), s_4) = \Phi_{\leq 2}(s_2^0, s_4) + A_{\leq 2}(s_2^0, s_4, (ab)) + R_{\leq 2}(s_2^0, s_4, (ab))$. Because $L(s_4, (ab)) = \{1\}$ and $s_4[K] = (ab)$ with $K = \{1\}$, we have $A_{\leq 2}(s_2^0, s_4, (ab)) = A^{\{1\}}_{\leq 2}(s_2^0, s_4, (ab)) = \sum_{k=1}^{2} \Phi_{\leq 2-k}(s_2^0, s_4^0) \times \binom{2}{k} = 1 + 2 = 3$. Indeed, when we concatenate (ab) to the empty subsequence, we add 3 extra common subsequences to $\Phi_{\leq 2}(s_2^0, s_4)$, i.e. the subsequences $\langle a \rangle$, $\langle b \rangle$ and $\langle (ab) \rangle$. Then, because $L(s_2^0, (ab)) = \emptyset$, we have

[5] We recall that for every sequence s, we define $s[K]$ as: $s[K] = \cap_{k \in K} s[k]$

$R_{\leq 2}(s_2^0, s_4, (ab)) = 0$. Thus, we finally obtain $\Phi_{\leq 2}(s_2^1, s_4) = 1 + 3 - 0 = 4$. Using Theorem 2, it is also possible to check that $\Phi_{\leq 2}(s_2^2, s_4) = \Phi_{\leq 2}(s_2^1 \circ (d), s_4) = \Phi_{\leq 2}(s_2^1, s_4) + A_{\leq 2}(s_2^1, s_4, (d)) + R_{\leq 2}(s_2^1, s_4, (d)) = 4 + 3 - 0 = 7$.

Now, we give an example where the correction term is not equal to zero. Let us consider the computation of $\Phi_{\leq 2}(s_2, s_4)$. Using Theorem 2, we have $\Phi_{\leq 2}(s_2, s_4) = \Phi_{\leq 2}(s_2^2 \circ (ac), s_4) = \Phi_{\leq 2}(s_2^2, s_4) + A_{\leq 2}(s_2^2, s_4, (ac)) + R_{\leq 2}(s_2^2, s_4, (ac))$. Because $L(s_4, (ac)) = \{1, 2\}$, we have $A_{\leq 2}(s_2^1, s_4, (d)) = A_{\leq 2}^{\{1\}}(s_2^2, s_4, (ac)) + A_{\leq 2}^{\{2\}}(s_2^2, s_4, (ac)) - A_{\leq 2}^{\{1,2\}}(s_2^2, s_4, (ac))$. Then, with $Y = (ac)$, because $s_4[K] \cap Y = (a)$ with $K = \{1\}$, $s_4[K] \cap Y = (c)$ with $K = \{2\}$, and $s_4[K] \cap Y = \emptyset$ with $K = \{1, 2\}$, we have $A_{\leq 2}^{\{1\}}(s_2^2, s_4, (ac)) = \sum_{k=1}^{2} \Phi_{\leq 2-k}(s_2^2, s_4^0) \times \binom{1}{k} = 1$, $A_{\leq 2}^{\{2\}}(s_2^2, s_4, (ac)) = \sum_{k=1}^{2} \Phi_{\leq 2-k}(s_2^2, s_4^1) \times \binom{1}{k} = \Phi_{\leq 1}(s_2^2, s_4^1) = 3$ and $A_{\leq 2}^{\{1,2\}}(s_2^2, s_4, (ac)) = 0$. Thus, we obtain $A_{\leq 2}(s_2^1, s_4, (d)) = 1 + 3$, which means that we add 4 extra common subsequences to $\Phi_{\leq 2}(s_2^2, s_4)$, i.e. the subsequences $\langle a \rangle$ (which is a repetition), $\langle c \rangle$, $\langle ac \rangle$ and $\langle bc \rangle$. Moreover, because $L(s_2^2, (ac)) = \{1\}$ and $L(s_4, (ac)) = \{1, 2\}$ and only $s_2^2[K] \cap s_4[K'] \cap Y$ with $K = \{1\}$, $K' = \{1\}$ and $Y = (ac)$ is non empty, we have $R_{\leq 2}(s_2^2, s_4, (ac)) = R_{\leq 2}^{\{1\},\{1\}}(s_2^2, s_4, (ac)) = \sum_{k=1}^{2} \Phi_{\leq 2-k}(s_2^0, s_4^0) \times \binom{1}{k} = 1$. This correction term is useful to take into account that the common subsequence $\langle a \rangle$ has already been counted. It follows that we finally obtain $\Phi_{\leq 2}(s_2, s_4) = 7 + 4 - 1 = 10$.

6.4. Complexity analysis and discussion

The complexity for the computation of the number of common subsequences between two sequences s_1 and s_2 with norm lower than M using Theorem 2 is in time $O(|s_1| \cdot |s_2| \cdot M^2 \cdot 2^{P_1 \cdot P_2} \cdot T)$ where P_1 is the maximum size of position sets $L(s_1^{i-1}, s_1[i])$, P_2 is the maximum size of position sets $L(s_2^j, s_1[i])$ and T is the maximum size of an itemset in s_1 or s_2.

Thus, given a positive sequential dataset $\mathcal{S}^+$ and a negative sequential dataset $\mathcal{S}^-$, the preprocessing time to draw subsequences with a probability proportional to their discriminative measure will be in time $O(|\mathcal{S}|^+ \cdot |\mathcal{S}|^- \cdot L^+ \cdot L^- \cdot M^2 \cdot 2^{P^+ \cdot P^-} \cdot T)$ where L^+ (resp. L^-) is the maximum length of a sequence in $\mathcal{S}^+$ (resp. in $\mathcal{S}^-$), M is the maximum norm of drawn subsequences, P^+ is the maximum size of position sets $L(s_1^{i-1}, s_1[i])$ with $s_1 \in \mathcal{S}^+$, P^- is the maximum size of position sets $L(s_2^j, s_1[i])$ with $s_1 \in \mathcal{S}^+$ and $s_2 \in \mathcal{S}^-$, and T is the maximum size of an itemset in a sequence in $\mathcal{S}^+$ or $\mathcal{S}^-$.

Concerning this complexity, it is first important to note that $P^+ \leq L^+$ and $P^- \leq L^-$ may be very small in practice, and that if the datasets $\mathcal{S}^+$ and $\mathcal{S}^-$ contain only sequences of *items* (and not sequences of *itemsets*), then we have $P^+ = P^- = 1$. Thus, in that case, the preprocessing can be performed in polynomial time $O(|\mathcal{S}|^+ \cdot |\mathcal{S}|^- \cdot L^+ \cdot L^- \cdot M^2 \cdot T)$. Nevertheless, it should be pointed out that this complexity is quadratic with the size of the datasets, which represents a significant bottleneck with large datasets. For that reason, it is clear that for quadratic measures, an *enumerative implementation* of the first step of the sampling procedure is certainly not a good solution, and that an indirect approach based on the *coupling from the past* exact sampling method (as proposed in [26])

is certainly a better solution. However, such an approach is out of the scope of this paper.

7. Conclusion

This paper proposes the first output space sampling method for sequential patterns with norm-based utility. This class of interestingness measures includes frequency and area. 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. Besides, the experiments show that as well with frequency as area, the addition of constraints on the norm or exponential decays 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. Whatever the measure (frequency or area), these models still have an accuracy comparable to some methods achieving a complete enumeration of the pattern search space.

We have already presented in Section 6 an important direction to extend our work to other interestingness measures involving several times the frequency. Of course, we can envisage many other future directions including:

- **Language extent:** We would especially like to extend our approach to more complex languages. A first advance would be to introduce a gap constraint to avoid having significant gaps between two itemsets of the same sequence [40]. More ambitiously, it would be interesting to consider any set system [41], which is a flexible framework of pattern languages. Indeed, the uniform drawing within complex structures made possible by a canonical form (here the first occurrence) can be envisaged with other structured languages.
- **Sequential pattern modeling:** As it 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 [15] or for designing interactive systems dedicated to sequential pattern discovery [12].

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

References

[1] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Proc. of ICDE 95*, 1995, pp. 3–14.
[2] 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.
[3] R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements," in *Proc. of EDBT 96*, 1996, pp. 3–17.
[4] M. J. Zaki, "SPADE: An efficient algorithm for mining frequent sequences," *Machine Learning*, vol. 42, no. 1-2, pp. 31–60, 2001.
[5] 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.

[6] M. N. Garofalakis, R. Rastogi, and K. Shim, "Spirit: Sequential pattern mining with regular expression constraints," in *VLDB*, vol. 99, 1999, pp. 7–10.
[7] J. Pei, J. Han, and L. V. Lakshmanan, "Mining frequent itemsets with convertible constraints," in *Proc. of ICDE 2001*. IEEE, 2001, pp. 433–442.
[8] J. Wang and J. Han, "Bide: Efficient mining of frequent closed sequences," in *Proc. of ICDE 2004*. IEEE, 2004, pp. 79–90.
[9] 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.
[10] 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.
[11] 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.
[12] 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.
[13] A. Giacometti and A. Soulet, "Interactive pattern sampling for characterizing unlabeled data," in *Proc. of IDA 2017*, 2017, pp. 99–111.
[14] 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.
[15] 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.
[16] V. Dzyuba and M. van Leeuwen, "Learning what matters–sampling interesting patterns," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2017, pp. 534–546.
[17] C. Anderson, "The long tail," *Wired magazine*, vol. 12, no. 10, pp. 170–177, 2004.
[18] L. Diop, C. T. Diop, A. Giacometti, D. Li, and A. Soulet, "Sequential pattern sampling with norm constraints," in *IEEE International Conference on Data Mining (ICDM)*, 2018.
[19] 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.
[20] 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.
[21] G. Li and M. J. Zaki, "Sampling minimal frequent boolean (DNF) patterns," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 87–95.
[22] S. Moens and B. Goethals, "Randomly sampling maximal itemsets," in *Proc. of IDEA Workshop 2013*, 2013, pp. 79–86.
[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] 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.
[25] M. Gueguen, O. Sentieys, and A. Termier, "Accelerating itemset sampling using satisfiability constraints on FPGA," in *IEEE/ACM Design, Automation and Test in Europe (DATE)*, 2019.
[26] M. Boley, S. Moens, and T. Gärtner, "Linear space direct pattern sampling using coupling from the past," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 69–77.
[27] S. Moens and M. Boley, "Instant exceptional model mining using weighted controlled pattern sampling," in *International Symposium on Intelligent Data Analysis*. Springer, 2014, pp. 203–214.
[28] A. Giacometti and A. Soulet, "Dense neighborhood pattern sampling in numerical data," in *Proc. of SDM 2018*, 2018, pp. 756–764.
[29] M. van Leeuwen, "Interactive data exploration using pattern mining," in *Interactive Knowledge Discovery and Data Mining in Biomedical Informatics*, 2014, pp. 169–182.
[30] S. Zilberstein, "Using anytime algorithms in intelligent systems," *AI magazine*, vol. 17, no. 3, p. 73, 1996.
[31] Q. Hu and T. Imielinski, "Alpine: Progressive itemset mining with definite guarantees," in *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM, 2017, pp. 63–71.
[32] Z. He, X. Xu, Z. J. Huang, and S. Deng, "Fp-outlier: Frequent pattern based outlier detection," *Computer Science and Information Systems*, vol. 2, no. 1, pp. 103–118, 2005.

[33] H. Toivonen *et al.*, "Sampling large databases for association rules," in *Proc. of VLDB 96*, vol. 96, 1996, pp. 134–145.
[34] 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.
[35] 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.
[36] 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.
[37] 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.
[38] 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*. Springer, 2013, pp. 109–120.
[39] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Dec. 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1248547.1248548
[40] R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements," in *International Conference on Extending Database Technology*. Springer, 1996, pp. 1–17.
[41] H. Arimura and T. Uno, "Polynomial-delay and polynomial-space algorithms for mining closed sequences, graphs, and pictures in accessible set systems," in *Proceedings of the 2009 SIAM International Conference on Data Mining*. SIAM, 2009, pp. 1088–1099.