Sequence Classification Based on Delta-Free Sequential Pattern

Pierre Holat, Marc Plantevit, Chedy Raïssi, Nadi Tomeh, Thierry Charnois, Bruno Crémilleux

Sequence Classification Based on Delta-Free Sequential Pattern

Pierre Holat*, Marc Plantevit†, Chedy Raïssi‡, Nadi Tomeh*, Thierry Charnois* and Bruno Crémilleux§
*Sorbonne Paris Cité, CNRS, Université Paris 13, LIPN UMR7030, F-93430, France
†Université de Lyon, CNRS, Université Lyon 1, LIRIS UMR5205, F-69622, France
‡INRIA Nancy Grand-Est, France
§Université de Caen Basse-Normandie, CNRS, GREYC UMR6072, F-14032, France

***Abstract*—Sequential pattern mining is one of the most studied and challenging tasks in data mining. However, the extension of well-known methods from many other classical patterns to sequences is not a trivial task. In this paper we study the notion of δ-freeness for sequences. While this notion has extensively been discussed for itemsets, this work is the first to extend it to sequences. We define an efficient algorithm devoted to the extraction of δ-free sequential patterns. Furthermore, we show the advantage of the δ-free sequences and highlight their importance when building sequence classifiers, and we show how they can be used to address the feature selection problem in statistical classifiers, as well as to build symbolic classifiers which optimizes both accuracy and earliness of predictions.**

***Keywords*—sequence mining; free patterns; text classification; feature selection; early classification**

I. Introduction

Sequence classification is an important component of many real-world applications where information is structured into sequences [1]. In biology, for instance, classifying DNA or protein sequences into various categories may help understanding their structure and function [2], while in medicine, classifying times series of heart rates may help identifying pathological cases [3]. Similarily, classifying textual documents into different topic categories is essential in many natural language processing (NLP) and information retrieval (IR) applications [4], [5]. However, sequence classification is a challenging task for several reasons, which we address in this paper.

First, the task of *feature selection* [6], which is an important step in many classification approaches that operate on a feature-based representation of the data, is not trivial. A simple approach would be to consider each item of the sequence as a feature. However, the sequential nature of the sequence and the dependencies between individual items cannot easily modeled. While more complete information can be captured by considering all possible sub-sequences instead of individual items, the exponential growth of the number of such subsequences is computationally prohibitive and results in sparsity issues. A simple middle-ground solution is to consider short segments of consecutive items, called n-grams as features [7], [8], however, the complete feature space is still not entirely explored.

Second, the classification accuracy may not be the only criterion we wish to optimize. In their key paper [9], Xing *et al.*, discuss the notion of *early prediction* for sequence classifiers. The authors note that: *"a reasonably accurate prediction using an as short as possible prefix [..] of a sequence is highly valuable"*. This is an important condition for critical applications that need to supervise and classify sequences as early as possible. For instance, in diagnosing a disease from a sequence of records in medical tests, or in network intrusion or failure detection systems, it is obviously better to detect and classify a sequence of to be abnormal at the onset of the disease or the beginning of the network attack or failure.

These issues can be addressed by exploiting sequential pattern mining techniques which can efficiently explore the complete feature space. Sequential pattern mining is one of the most studied and challenging tasks in data mining. Since its introduction by *Agrawal and Srikant* in [10], many researchers developed approaches to mine sequences in different and various fields such as bioinformatics, customer marketing, web log analysis and network telecommunications. While the extraction of sequential patterns can be seen as an end in itself, it has been shown useful as a first step to build global classification models [11]–[13]. The idea behind this process is that the extracted sequential patterns are easily manipulated, understood and used as *features* or *rules* by classification methods and models [14]–[18].

However, it is generally known that pattern mining typically yields an exponential number of patterns. Hence, many researchers focused on selecting a small subset of patterns with the same expressiveness power without jeopardizing the classification accuracy. Two of the most-used *concise representations*, the free and closed patterns, find their origin in Galois lattice theory and Formal Concept Analysis. A set of patterns is said to form an equivalence class if they are mapped to the same set of objects (or transactions) of a data set, and hence have the same *support*. The maximal element of an equivalence class is usually referred to as the closed pattern of the equivalence class. On the contrary, a free pattern (or generator) is a minimal element of the equivalence class. The authors in [19] studied and compared the efficiency of generators and closed patterns and concluded that *"generators are preferable in inductive inference and classification when using the Minimum Description Length principle"*. In the case of sequential patterns (as opposed to itemset patterns), no previous work tries to compare the different concise representations because the methods are not easy to transpose. However, depending on the support parameter, the number of free patterns may still be prohibitively large.

In this paper we solve this problem by introducing a new algorithm for the extraction of free patterns. We show the usefulness of these patterns in addressing the two issues of sequence classification mentioned above, namely: feature selection and earliness optimization. The contribution of this paper is thus three-fold.

First, in Section II we shed new light on the problem of concise representations for sequences by analyzing the usage of δ-free sequences, with δ being a parameter that allows to group equivalence classes with similar support values, and hence provide finer control on the number of extracted patterns. This is the first study to extend the notion of *freeness* to sequences. We introduce and discuss properties showing that δ-free sequences are indeed an efficient condensed representation of sequences and introduce a new algorithm to compute them. Second, in Section III, we describe a pipeline approach to textual document classification that uses δ-free patterns as features, and show that it outperforms other feature selection baseline methods while using smaller number of features. Third, in Section IV, we show that δ-free patterns are efficient to build a sequence classifier that optimizes both accuracy and earliness. This classifier is based on special rules called δ-strong sequence rules. We present a novel technique to select the *best* δ-strong rules from δ-free patterns.

II. Mining δ-free Sequential Patterns

In this section we present a novel algorithm to extract δ-free patterns from a database of sequences. We start by formalizing the problem and providing the necessary definitions in Sections II-A and II-B. We then describe the algorithm in Section II-C and analyze its performance Section II-D.

A. Definitions and problem description

Let $\mathcal{I} = \{i_1, i_2 \ldots i_m\}$ be the finite set of items. An itemset is a non-empty set of items. A sequence S over $\mathcal{I}$ is an ordered list $\langle it_1, \ldots, it_k \rangle$, with it_j an itemset over $\mathcal{I}$, $j = 1 \ldots k$. A k-sequence is a sequence of k items (i.e., of length k), $|S|$ denotes the length of sequence S and $S[0, l]$ denotes the l-sequence identified as a prefix of sequence S. $\mathbb{T}(\mathcal{I})$ will denote the (infinite) set of all possible sequences over $\mathcal{I}$ and $\mathbb{L}$ denotes the set of labels, or classes. A *labeled sequence database* $\mathcal{D}$ over $\mathcal{I}$ is a finite set of triples (SID, T, C), called transactions, with $SID \in \{1, 2, \ldots\}$ an identifier, $T \in \mathbb{T}(\mathcal{I})$ a sequence over $\mathcal{I}$ and $C \in \mathbb{L}$ is the class label associated to the sequence T. Let $s = \langle e_1, e_2, \ldots, e_n \rangle$ be a sequence. We denote by $s^{(i)} = \langle e_1, \ldots, e_{i-1}, e_{i+1}, \ldots, e_n \rangle$, the sequence s in which the i^{th} elements is deleted. '$\cdot$' denotes the itemset-append operation between 2 sequences and '$-\cdot$' denotes the item-append operation between 2 sequences. For instance, $\langle (a)(b) \rangle \cdot \langle (b)(a) \rangle = \langle (a)(b)(b)(a) \rangle$ and $\langle (a)(b) \rangle - \cdot \langle (c)(a) \rangle = \langle (a)(b, c)(a) \rangle$.

*Definition 2.1 (*Inclusion*):* A sequence $S' = \langle is'_1 \ is'_2 \ \ldots \ is'_n \rangle$ is a subsequence of another sequence $S = \langle is_1 \ is_2 \ \ldots \ is_m \rangle$, denoted $S' \preceq S$, if there exist $i_1 < i_2 < \ldots i_j \ldots < i_n$ such that $is'_1 \subseteq is_{i_1}, \ is'_2 \subseteq is_{i_2} \ldots is'_n \subseteq is_{i_n}$.

*Definition 2.2 (*Support*):* The support of a sequence S in a transaction database $\mathcal{D}$, denoted $Support(S, \mathcal{D})$, is defined as: $Support(S, \mathcal{D}) = |\{(SID, T) \in \mathcal{D} | S \preceq T\}|$. The frequency of S in $\mathcal{D}$, denoted $freq_S^{\mathcal{D}}$, is $freq_S^{\mathcal{D}} = \frac{Support(S,\mathcal{D})}{|\mathcal{D}|}$.

Given a user-defined minimal frequency threshold σ, the problem of sequential pattern mining is the extraction of all the sequences S in $\mathcal{D}$ such that $freq_S^{\mathcal{D}} \geq \sigma$. The set of all frequent sequences for a threshold σ in a database $\mathcal{D}$ is denoted $FSeqs(\mathcal{D}, \sigma)$[1],

$$FSeqs(\mathcal{D}, \sigma) = \{S \mid freq_S^{\mathcal{D}} \geq \sigma\}$$

TABLE I. The sequence database used as the running example.

S_1	$\langle (a)(b)(c)(d)(a)(b)(c) \rangle$	+
S_2	$\langle (a)(b)(c)(b)(c)(d)(a)(b)(c)(d) \rangle$	+
S_3	$\langle (a)(b)(b)(c)(d)(b)(c)(c)(d)(b)(c)(d) \rangle$	+
S_4	$\langle (b)(a)(c)(b)(c)(b)(b)(c)(d) \rangle$	+
S_5	$\langle (a)(c)(d)(c)(b)(c)(a) \rangle$	−
S_6	$\langle (a)(c)(d)(a)(b)(c)(a)(b)(c) \rangle$	−
S_7	$\langle (a)(c)(c)(a)(c)(b)(b)(a)(e)(d) \rangle$	−
S_8	$\langle (a)(c)(d)(b)(c)(b)(a)(b)(c) \rangle$	−

*Example 2.3 (*Running Example*):* In this paper, we use the sequence database $\mathcal{D}_{ex}$ in Table I containing 8 data sequences with $\mathcal{I} = \{a, b, c, d, e\}$ and $\mathbb{L} = \{+, -\}$ as the running example. Sequence $\langle (a)(b)(a) \rangle$ is included in $S_1 = \langle (a), (b), (c), (d), (a), (b), (c) \rangle$. Sequence S_1 is thus said to support $\langle (a)(b)(a) \rangle$. Notice, however, that S_5 does not support $\langle (b)(d) \rangle$ as $\langle (b)(d) \rangle \not\preceq S_5$. In addition $S_4[0, 3] = \langle (b)(a)(c) \rangle$ is the prefix of sequence S_4 of length 3.

For the sake of simplicity, we limit our examples and discussions to sequences of items, but all our propositions and theorems hold for the general case of sequences of itemsets.

Definition 2.4 (Projected database [20]): Let s_p be a sequential pattern in sequence database $\mathcal{D}$. The s_p-projected database, denoted as $\mathcal{D}_{|s_p}$, is the collection of suffixes of sequences in $\mathcal{D}$ having the prefix s_p.

Note that the prefix of a sequential pattern s_p within a data sequence S is equal to the subsequence of S starting at the beginning of S and ending strictly after the first *minimal occurrence* of s_p in S [21]. In the running example, $\mathcal{D}_{ex|\langle (a)(b)(a) \rangle} = \{\langle (b)(c) \rangle, \langle (b)(c)(d) \rangle, \langle \rangle, \langle \rangle, \langle \rangle, \langle (b)(c) \rangle, \langle (e)(d) \rangle, \langle (b)(c) \rangle\}$.

B. δ-free sequential patterns

The notion of minimal patterns according to a constraint has already been explored for more than a decade in the itemset framework. In this context, the free patterns (also called *minimal generators*) are the minimal patterns according to the frequency measure. In order to accept some few exceptions, this notion is generalized with the δ-free patterns introduced and studied in [22]. To the best of our knowledge, this notion of δ-freeness was never introduced or defined in the context of sequences. In the following, we extend this notion to sequences.

Definition 2.5 (δ-free sequential patterns): Given a sequence database $\mathcal{D}$, a sequence s is δ-free if:

$$\forall s' \prec s, \ Support(s', \mathcal{D}) > Support(s, \mathcal{D}) + \delta$$

[1] In the case that σ is an integer, $freq_S^{\mathcal{D}}$ is defined w.r.t. $Support(S, \mathcal{D})$. In the rest of the paper, σ is an integer if it is not specified.

The δ-free sequential patterns are especially interesting in real-world domains where few exceptions often appear and data sets often contain missing, incorrect or uncertain values. By using δ-free sequential patterns, one takes a more pragmatic approach to the extraction of sequential patterns towards the final goal of classification. Furthermore, this new type of pattern is also appealing from an implementation point of view as it helps maintaining very fast runtimes. Note also that a δ-free sequential pattern is a sequence that cannot be represented as a rule accepting less than δ errors.

Given the sequence database $\mathcal{D}_{ex}$ (Table I), Figure 1 represents all 1-free sequential patterns (**in bold**) having a support greater or equal to 3. For instance, $\langle(b)(d)(b)\rangle_3$ is a 1-free sequence whereas sequence $\langle(b)\rangle_8$ is not 1-free since it has the same support as $\langle\rangle_8$.

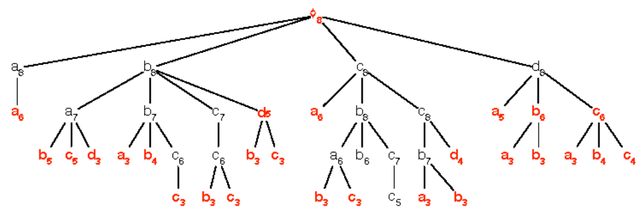


Fig. 1. The enumeration tree of the Frequent 1-Free sequential patterns on $\mathcal{D}$ ($\sigma = 3$)

The next subsection presents an algorithm that efficiently mines δ-free frequent sequences.

C. DeFFeD : *a new extraction algorithm*

To understand the underlying complexity gap between itemsets and sequences representations, one can notice that the set of frequent free patterns is a concise representation of the frequent itemsets that can be efficiently obtained thanks to an anti-monotonicity property. However, this is not true anymore for sequences. The next property highlights this fundamental difference and the complex algorithmic challenges that result.

Property 2.6: Anti-monotonicity property does not hold for δ-free sequences.

A simple illustration from our running example suffices to show that sequence $\langle(a)\rangle$ is not 1-free whereas sequence $\langle(a)(a)\rangle$ is 1-free. As a consequence, it is impossible to use counting inference for sequences with δ-free patterns. Note that this property meets the pessimistic results of [23]. Sequence generators [24], [25] are a particular case of δ-free sequence (i.e., $\delta = 0$). In [24], [25], the authors introduced a monotonous property for a subset of non-generator sequences. We extend this property to δ-free sequences. This generalization is based on the notion of δ-equivalence of projected databases.

Definition 2.7 (δ-equivalence of projected databases): Let s and s' be two sequences, Their respective projected database $\mathcal{D}_{|s}$ and $\mathcal{D}_{|s'}$ are said δ-equivalent (denoted by $\mathcal{D}_{|s} \equiv_\delta \mathcal{D}_{|s'}$) if they have at most δ different suffixes.

This definition can be exploited to produce a monotone property of some non δ-free sequences:

Property 2.8: Let s and s' be two sequences. If $s' \prec s$ and $\mathcal{D}_{|s} \equiv_\delta \mathcal{D}_{|s'}$, then no sequence with prefix s can be δ-free.

Proof: (By contradiction) Let s and s' be two sequences such that $s' \prec s$, $Support(s', \mathcal{D}) - Support(s, \mathcal{D}) \leq \delta$ and $\mathcal{D}_{|s} \equiv_\delta \mathcal{D}_{|s'}$. Assume that there exists a sequence $s_p = s \cdot s_c$ that is δ-free. Since $\mathcal{D}_{|s} \equiv_\delta \mathcal{D}_{|s'}$, there exists a sequence $s'' = s' \cdot s.c$ such that $Support(s'', \mathcal{D}) - Support(s_p, \mathcal{D}) \leq \delta$. This leads to a contradiction to the assumption that s_p is δ-free. ■

Property 2.8 is very interesting as it avoids the exploration of unpromising sequences. Furthermore, the verification of δ-equivalence of projected databases can be restricted only to subsequence of length $n-1$ as stated in Property 2.9:

Property 2.9 (Backward pruning): Let $s_p = \langle e_1, e_2, \ldots, e_n\rangle$ be a prefix sequence. If there exists an integer i ($1 \leq i < n-1$) such that $\mathcal{D}_{|s_p} \equiv_\delta \mathcal{D}_{|s_p^{(i)}}$, then the exploration of the sequence s_p can be stopped since there is no other δ-free sequential patterns in $\mathcal{S}$ with prefix s_p that can be discovered.

Property 2.9 enables the efficient pruning of *unpromising* sequences and can be trivially included in any algorithm mining free sequential patterns.

Property 2.10: Let $s_p = \langle e_1, e_2, \ldots, e_n\rangle$ be a prefix sequence. If s_p is δ-free then s_p cannot be pruned (unpromising).

Proof: If s_p is δ-free then there exists no integer i such that $Support(s_p, \mathcal{D}) + \delta < Support({s_p}^{(i)}, \mathcal{D})$. Hence, there exists no integer i such that $s_p \equiv_\delta {s_p}^{(i)}$ and the pruning of s_p cannot be applied. ■

While these properties enable the full exploitation of the monotonous property of some non δ-free sequences, one can also take benefit of the combination of two constraints: the δ-freeness and the frequency. In the case where sequences are within the neighborhood of the positive border of the frequent sequences, the combination of the two constraints can be used as stated in the following property.

Property 2.11: Let σ be the minimum support threshold. Let s_p be a sequence such that $\sigma \leq Support(s_p, \mathcal{D}) \leq \sigma + \delta$, then the exploration of the sequence s_p can be stopped.

Proof: It is easy to prove that sequences with prefix s_p cannot be both frequent and δ-free. ■

Both Properties 2.9, 2.10 and 2.11 are used as pruning techniques in Algorithm of Figure 2 called DeFFeD (DElta Free Frequent sEquence Discovery). In the same spirit as Bide algorithm for closed sequential patterns [26], DeFFeD mines frequent δ-free sequences without candidate maintenance. It adopts a *bi-directional* checking to prune the search space deeply. DeFFeD only stores a set of frequent sequences that are δ-free. This is a huge advantage compared to the generate-and-prune algorithms that would not otherwise handle the impressive number of non δ-free frequent sequences. In addition, it is important to note that δ-free sequences do not provide a condensed representation of frequent sequential patterns. They have to be combined with other patterns (maximal frequent sequential patterns) to exclude some infrequent patterns.

To discover the complete set of frequent δ-free sequential patterns in sequence database $\mathcal{D}$ (i.e., all the frequent δ-free sequence with prefix $\langle\rangle$), algorithm DeFFeD must be launched as follows: $\texttt{DeFFeD}(\sigma, \delta, \langle\rangle, \mathcal{D}, \{\langle\rangle_{|\mathcal{D}|}\})$. Indeed, $\langle\rangle_{|\mathcal{D}|}$ is, by definition, the smallest δ-free sequential pattern. Algorithm

DEFFED first scans the sequence database to find the frequent 1-sequences (Line 1). Then, it treats each frequent 1-sequence (Line 4) as a prefix and check if the prefix sequence is δ-free (Line 9). Finally, if the prefix sequence is worth being explored (tests in Lines 15 and 21), the algorithm is recursively called on the prefix sequence.

Data : σ, δ, prefix sequence s_p and its projected database $\mathcal{D}_{|s_p}$, FFS
Result : $FFS\cup$ The set of frequent δ-free sequences with prefix s_p

1: $LFI \leftarrow$ frequent 1-sequences($\mathcal{D}_{|s_p}$, σ);
2: $is\_free \leftarrow \bot$;
3: $unpromising \leftarrow \bot$;
4: **for all** item $e \in LFI$ **do**
5: $\quad s'_p = \langle s_p \cdot e \rangle$;
6: $\quad \mathcal{D}_{|s'_p} \leftarrow$ pseudo_projected_database($\mathcal{D}_{|s_p}$, s'_p);
7: $\quad$ **if** $Support(s'_p, \mathcal{D}) + \delta < Support(s_p, \mathcal{D})$ **then**
8: $\quad\quad$ //potentially δ-free
9: $\quad\quad$ **if** $\nexists$ integer i and $Support({s'_p}^{(i)}, \mathcal{D}) - \delta > Support(s'_p, \mathcal{D})$ **then**
10: $\quad\quad\quad FDS \leftarrow FDS \cup \{s'_p\}$;
11: $\quad\quad\quad is\_free \leftarrow \top$;
12: $\quad\quad$ **end if**
13: $\quad$ **end if**
14: $\quad$ **if** $\neg is\_free$ **then**
15: $\quad\quad$ **if** $\nexists$ integer i — $\mathcal{D}_{|s'_p} \equiv_\delta \mathcal{D}_{|{s'_p}^{(i)}}$ **then**
16: $\quad\quad\quad unpromising \leftarrow \top$;
17: $\quad\quad$ **end if**
18: $\quad$ **end if**
19: $\quad$ **if** $\neg unpromising$ **then**
20: $\quad\quad$ /* check if it is possible to find frequent δ-free sequences (property 2.11)*/
21: $\quad\quad$ **if** $Support(s'_p, \mathcal{D}) > \sigma + \delta$ **then**
22: $\quad\quad\quad$ **Call** DEFFED $(\sigma, \delta, s'_p, \mathcal{D}_{|s'_p}, FFS)$;
23: $\quad\quad$ **end if**
24: $\quad$ **end if**
25: **end for**

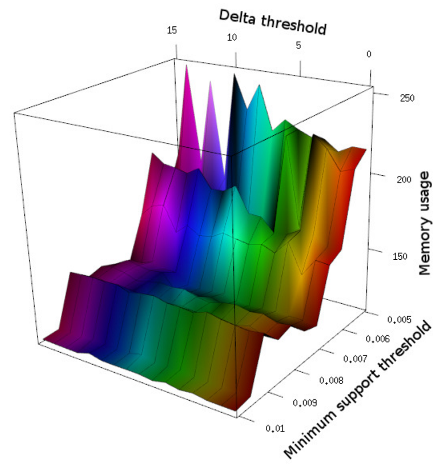
Fig. 2. Algorithm DeFFeD (DElta Free Frequent sEquence Discovery)

D. Performance analysis

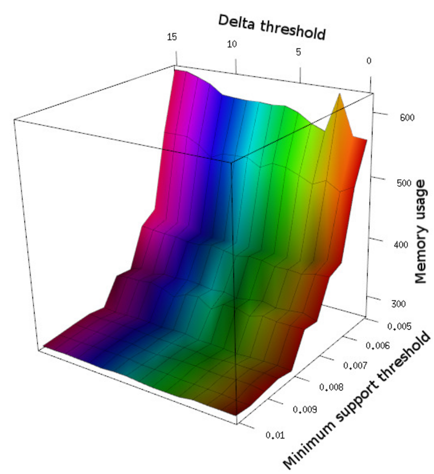
We report a performance evaluation of our algorithm on both synthetic and real datasets (source and data sets are publicly available[1]). The different data sets used for the experiments and their parameters are summarized in table II. The data sets $S50TR2SL10IT10K$ and $S100TR2SL10IT10K$ are generated with the QUEST[2] software. The $PremierLeague$ data set is a collection of sequences of football games played in England in the last 4 years. The version of the data sets used here is discretized to meet the classical sequential patterns needs.

We analyze the results of the experiments with regard to the two following questions: (a) How does the algorithm DEFFED behave with respect to the usual threshold parameters settings?

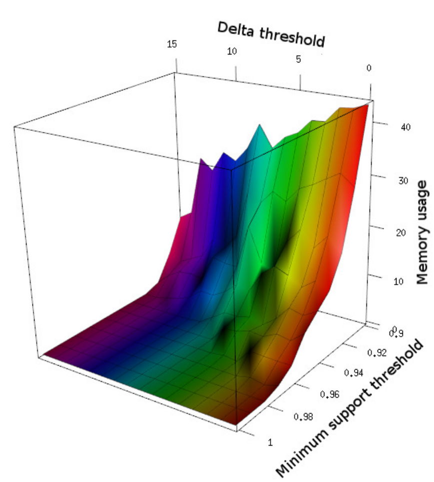
[1]http://lipn.univ-paris13.fr/∼holat/
[2]http://www.almaden.ibm.com/cs/projects/iis/hdb/Projects/data_mining/



(a) S50TR2SL10IT10K



(b) S100TR2SL10IT10K

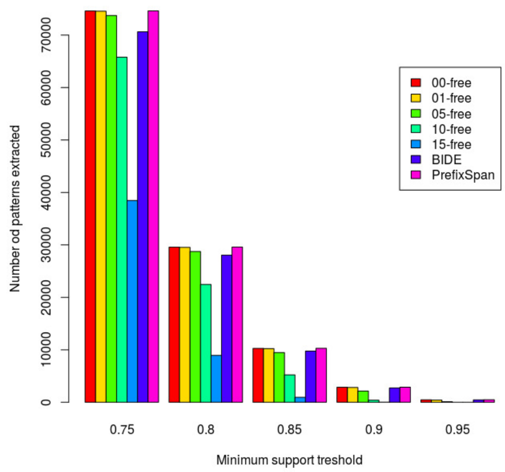


(c) PremierLeague

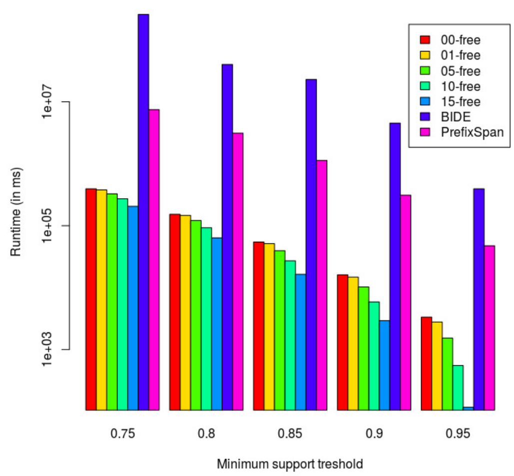
Fig. 3. The effects of varying δ w.r.t minimal support on memory usage (in Mbytes).

TABLE II. DIFFERENT DATA SETS USED FOR THE EXPERIMENTS.

Data Set	Items	Avg. size of itemsets	Avg. size of sequences	# of data sequences
S50TR2SL10IT10K	10000	2	10	50000
S100TR2SL10IT10K	10000	2	10	100000
PremierLeague	240	2	38	280



(a) PremierLeague patterns comparison



(b) PremierLeague runtime comparison

Fig. 4. Comparison between BIDE, PrefixSpan and DEFFED .

(b) How does DEFFED compare to state-of-the-art algorithms BIDE [27] and PrefixSpan [28]?

All the experiments were performed on a cluster and DEFFED , BIDE and PrefixSpan were implemented in Java. Nodes are equipped with 8 processors at 2.53GHz and 16Go of RAM under Debian operating system.

Figure 3 shows the impact of δ with respect to the minimal support threshold σ on the memory usage. As previously discussed with higher δ, the number of extracted sequential patterns tends to get very low. The consumption of memory follows this behavior in 3(c). But not with the generated data sets because the δ parameter is not high enough (in comparison to the number of sequences) to impact the number of extracted patterns. This space management via the δ parameter can be very useful in systems where memory is a core issue (i.e. embedded systems, sensors), again in these cases, a high parameter δ can help pushing the extraction process to very low supports.

One remaining issue is the comparison of DEFFED with well-known and efficient approaches (i.e., BIDE for closed frequent sequence and PrefixSpan for frequent sequences). From a theoretical point of view, nothing can be stated about the cardinality of the set of frequent $delta$-free sequences in comparison to the set of frequent closed sequences (the 0-free sequences are the minimal sequences in a support equivalence class, while closed sequences are the maximal ones). However, in Figure 4, one can notice the efficiency of the DEFFED in terms of runtime and in Figure 4(a), with a small error value $\delta = 5$ or $\delta = 10$, the number of extracted patterns is drastically lower than closed sequences. Notice that the $PremierLeague$ data set is very dense which explains the very high runtime values. For instance, for $\sigma = 0.75$, $BIDE$ takes more than 250 million milliseconds (69 hours) to complete the extraction process.

III. DEFFED FOR FEATURE SELECTION

In this section we investigate the utility of δ-free patterns as features in a supervised text classification task. We describe our classification approach in Section III-A and discuss our experiments in Section III-B.

A. A supervised classification approach

We follow [29] and employ a maximum entropy (MaxEnt) framework to model the probability of a category label c given a sequence $\mathbf{s}$ according to Equation 1.

$$P(c|\mathbf{s}) = \frac{1}{Z(\mathbf{s}, \boldsymbol{\theta})} \exp\{\sum_{k=1}^{|\boldsymbol{\theta}|} \theta_{k,c}\; g_k(\mathbf{s})\} \qquad (1)$$

The partition function Z acts as a normalizer; each g_k is a binary feature function which returns 1 if the feature $k \in \mathcal{K}$ is present in the sequence $\mathbf{s}$ and 0 otherwise; and the parameter $\theta_{k,c} \in \boldsymbol{\theta}$ associates a weight to each feature in a given category. The classification task amounts to searching for the most probable category $\hat{c} \in \mathcal{C}$ according to the rule $\hat{c} = \arg\max_{c \in \mathcal{C}} P(c|\mathbf{s})$. The parameters $\boldsymbol{\theta}$ of the MaxEnt model are learned during training on a corpus of n labeled sequences $\mathcal{D} = \{(\mathbf{s}_i, c_i)\}_{i=1}^{n}$.

The DEFFED algorithm intervene in this approach when computing the set of features $\mathcal{K}$ used by the classifier. During training, we divide the training corpus by categories into distinct subsets such that $\mathcal{D} = \cup_c\{\mathcal{D}_c\}$. We run the extraction algorithm on each subset $\mathcal{D}_c$ independently, and construct the set of δ-free patterns which we call $\mathcal{K}_c$. We aggregate all such sets to construct the set of features to be used by the classifier $\mathcal{K} = \cup_c\{\mathcal{K}_c\}$. The ability to produce an accurate estimation of the model parameters $\boldsymbol{\theta}$ depends heavily on their number and the sparsity of the data, which is directly related to the number of patterns produced by $DeFFeD$. We compare the δ-free based approach to building $\mathcal{K}$ with several selection approaches, including using individual items (bag of word) or contiguous short segments (n-grams) as features. The classification performance is evaluated using the well-known F-measure.

B. Experiments

We report an experimental evaluation of our approach by doing text classification using a real data set proposed by the French Laboratory LIMSI during the DEFT'2008 evaluation campaign[1]. The corpus statistics are given in Table III. Each document is modeled as a sequence and the set of possible categories for each document is $\mathcal{C} = \{$sport, economy, television, art$\}$. The sources of those documents are articles from the French newspapers *"Le Monde"* and the online free encyclopedia *"Wikipedia"*. We use the Wapiti[2] [30] implementation of the MaxEnt classifier in its default settings.

TABLE III. DETAILS OF THE DEFT DATA SET.

Data Set	# of documents	# of words	# of distincts words
Training set	15,223	3,375,888	161,622
Test set	10,596	2,306,471	128,377

Table IV presents the results of our text classification experiments. With the baseline approaches to feature selection, the best performance we were able to achieve is using all contiguous patterns, without gaps between the individual items in the source sequence and with a maximum size of 7, as features. The baseline approach did not scale up to include patterns with gaps due to memory limitations. We also compared our method with the VOGUE method [13]. VOGUE is a variable order and gapped Hidden Markov Model (HMM) with duration. It uses sequence mining to extract frequent patterns in the data. It then uses the mined patterns to build a variable order HMM with explicit duration on the gap states for sequence modeling and classification. The implementation (available on the author's website[3]) uses the python extension module Psyco[4] to speed up the computation, which is unmaintained and only available for 32bit systems. Therefore it is limited to 4Go of RAM, hence the really bad F-measure on a large data set like DEFT. The baseline experiments also include some basic features selection: frequents patterns extraction with an absolute minimal support $\sigma = 5$. We can see that this features selection reduces the number of features but at the cost of some accuracy. However the use of a delta threshold of $\delta = 10\%$ (relative to the number of documents) results in comparable performances to the best baseline approach in terms of F-measure, while reducing dramatically the number of features. We can see on Figure 5 that with higher values of the minimum support σ the classification is really inefficient, as expected when using only patterns that are *"too"* frequent. But we can also see that the threshold δ enhances the classification. The experiments were done with different minimal support values: 0.01%, 0.025%, 0.05%, 0.1%, 0.2%, 0.4%, 0.8%, 1.6%, 3.2%, 6.4%, 12.8% and 25.6% and δ values of 0%, 0.025%, 0.05%, 0.1%, 0.2%, 0.4%, 0.8%, 1.6%, 3.2%, 6.4%, 12.8%, 25.6% and 51,2% w.r.t to the number of documents of the corpus. Figure 6 highlights the effect of of low thresholds on the F-measure. The noticeable drop of the F-measure at $\delta = 0$, $\sigma = 0.01\%$ and 0.025% is because the extraction returns too many patterns to be handled as features by the classifier. However with $\delta > 0$, DEFFED is able to produce a low number of features to be processed by the classifier even with a low minimal support of $\sigma = 0.01\%$. Figure 7 is a more complete view of the effects of δ on the F-measure (δ from 0 to 1 with 0.02 steps, $\sigma = 0.005\%$). With a $\delta = 1$ (100%), the only δ-free sequence is the empty pattern $\langle\rangle$. There are no features to process, hence the $F - measure = 0$.

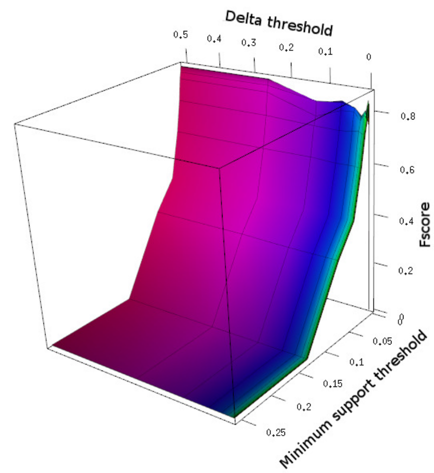


Fig. 5. Effects of the δ-freeness and minimal support (σ) parameters on the F-measure

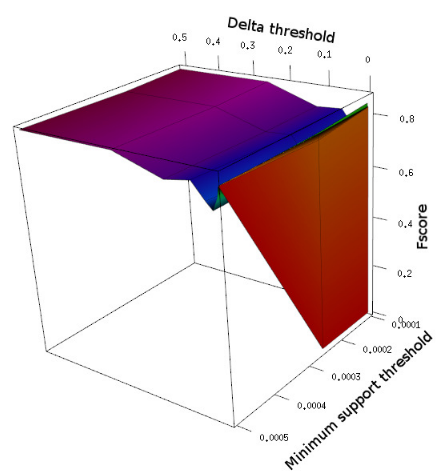


Fig. 6. Effects of **low** δ-freeness and minimal support (σ) parameters on the F-measure

In order to better understand the effect of the interaction between the parameters σ and δ on the number of extracted patterns for the DEFT data set, Figure 8 plot the number of patterns as a function of these two parameters. The extraction

[1]http://deft.limsi.fr/2008
[2]http://wapiti.limsi.fr/
[3]http://www.cs.rpi.edu/~zaki/www-new/pmwiki.php/Software/Software
[4]http://psyco.sourceforge.net/

TABLE IV. TEXT CLASSIFICATION RESULTS. δ AND THE MINIMUM SUPPORT σ ARE THE PARAMETERS OF THE EXTRACTION ALGORITHM.

	Model	σ	δ	F-measure	# of model parameters	Model size
Baselines	bag of word	0	-	0.863	646.488	21Mb
	frequent word	5	-	0.865	210.820	7Mb
	4-gram (no gap)	0	-	**0.870**	33.967.272	1306Mb
	frequent 4-gram (no gap)	5	-	0.865	477.188	21Mb
	7-gram (no gap)	0	-	0.853	73.060.660	3036Mb
	frequent 7-gram (no gap)	5	-	0,865	483.464	16Mb
	VOGUE (gap max. of 5)	0.05%	-	0,23	-	1902Mo
DEFFED	0-free	0.05%	0	0,823	104.240	4Mb
	10%-free	0.05%	10%	**0.870**	**26.764**	**0.8Mb**

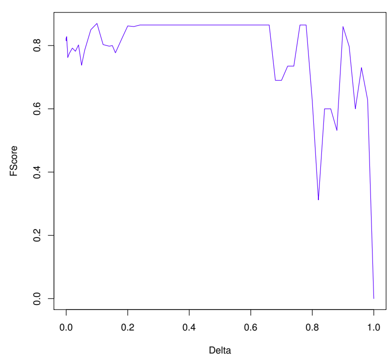


Fig. 7. Detailed effect of δ-freeness (0 to 100%) on classification F-measure with a minimal support of 0.05%

of sequential patterns with $\sigma = 0.01\%$ and $\delta = 0$ failed because of the large number of patterns to explore. However, when $\delta > 0$, we are able to extract the $\delta-$free patterns even with a minimum support as low as $\sigma = 0.01\%$, whereas it is not possible in the absence of the δ parameter (i.e., $\delta = 0$).

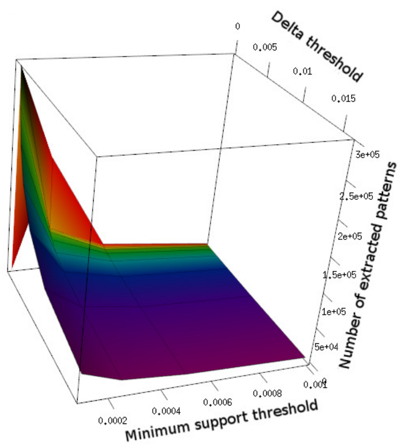


Fig. 8. Effect of δ-freeness and minimal support (σ) on the number of extracted patterns

We can see in Figure 9, which represent the running time of our algorithm on the DEFT data set, that the δ-freeness also play a major role in the efficiency of the extraction process.

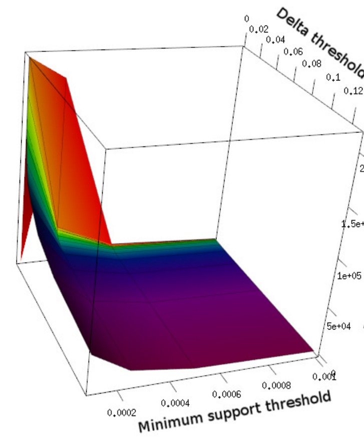


Fig. 9. Effect of δ-freeness and minimal support (σ) on extraction time

The experiments described in this section provide an empirical evidence on the usefulness of DEFFED as a feature selection method which results in a much smaller number of features without sacrificing the classification performance.

IV. DEFFED FOR EARLY PREDICTION

In this section we introduce the δ-strong sequential rules and our early-prediction sequence classifier. We show that the δ-free sequences are very efficient to build early prediction sequence classifiers that rely on high accuracy of the prediction coupled with minimal costs. We discuss our experiments in Section IV-B.

A. Sequential classification rules based on δ-free sequential patterns

A sequential classification rule r is an implication of the form $r : s \rightarrow c$ in which the premise of the rule is a sequence from $\mathbb{T}(\mathcal{I})$ and the conclusion c is a class label from $\mathbb{L}$. Such a rule can be evaluated with the usual support-based measures which are based on the support of the sequence s in the partition of the data set which contains class label c (denoted $Support(s, \mathcal{D}_c)$). The confidence of a sequential rule is Confidence$(r, \mathcal{D}) = \frac{Support(s,\mathcal{D}_c)}{Support(s,\mathcal{D})}$.

Sequence database $\mathcal{D}$ can then be partitioned into n subsets $\mathcal{D}_i$ where $\mathcal{D}_i$ contains all data sequences related to class label $c_i \in \mathbb{L}$. A sequential δ-strong rule is an implication of the form $r : s \rightarrow c_i$ if, given a minimum support threshold σ and an integer δ, the following conditions hold:

$$Support(s, \mathcal{D}) \geq \sigma \text{ and } Support(s, \mathcal{D}) - Support(s, \mathcal{D}_i) \leq \delta$$

A δ-strong rule accepts at most δ errors, that is, its confidence is lower-bounded: $1 - \frac{\delta}{\sigma} \leq \text{Confidence}(s \rightarrow c_i, \mathcal{D}) \leq 1$.

Given the property of minimality of the δ-free patterns that we present in section II-B, if we use a δ-free pattern as a premise of a δ-strong rule we are ensuring that there does not exist $s' \prec s$ such that s' is the premise of a δ strong rule. By considering the property of minimal body, the number of sequential rules is highly reduced. To understand that, observe that for a given δ-strong sequential rule $s \rightarrow c_i$, the following inequalities hold,

$$\begin{aligned} Support(s, \mathcal{D}) &\geq \sigma \ ; \\ Support(s, \mathcal{D} \backslash \mathcal{D}_i) &\leq \delta \ ; \\ Support(s, \mathcal{D}_i) &\geq \sigma - \delta \ . \end{aligned}$$

In particular,

$$\sigma - \delta \leq Support(s, \mathcal{D}) \leq |\mathcal{D}_i| + \delta$$

Minimal δ-strong sequential classification rules also satisfy interesting properties on rule conflicts. Indeed, several rule conflicts properties proved in [31] also hold for sequential patterns. When inequality $\delta < \frac{\sigma}{2}$ is respected, it is obvious that it will be impossible to find a specialization of a premise leading to a different conclusion, *i.e.,* a different class label.

For the selection of the best δ-strong rules, we have to use a set of rules avoiding classification conflicts. Thanks to the properties of the δ-free sequential patterns, if $\delta < \frac{\sigma}{2}$, we cannot have two δ-strong sequential classification rules $r_1 : s \rightarrow c$ and $r_2 : s' \rightarrow c'$ such that $s' \preceq s$ and $c \neq c'$.

Early prediction oriented sequence classifiers have to process itemsets from a sequence in a consecutive and progressive way. Obviously, these classifiers rely, for the prediction, exclusively on the prefix of a sequence. Each sequence itemset i, processed by a classifier is associated with a *cost* value $c(i)$. The total cost of prediction for a sequence S, denoted $c(S)$, is the sum of the costs of each item in the minimal prefix sequence to achieve the classification task.

According to early prediction purpose, we assume that new sequences arrive item by item. The goal of the early prediction is to associate a class label to the new sequence as soon as possible. At each update of the new unclassified sequence, the classifier tries to match the sequence to the premises of the rules. The best way to directly focus on the new incoming item of the sequence is to store the δ-strong rules of the classifier in a *suffix tree structure*. The suffix tree stores all the rules of the classifier. The leaves of the tree contain class labels and support information. The use of a suffix tree to store δ-strong rules enables to directly concentrate on promising rules. Notice that suffix tree structure was successfully applied by [32] to approximate sequence probabilities and discover outliers in sequence databases.

B. Experiments

We report qualitative results of our early-based sequence classifier over real-world data sets. The different data sets used for the experiments and their parameters are summarized in table V. The $SENSOR$ and $PIONEER$ data sets are downloaded from the UCI Machine Learning Repository. The data were collected through sensors as robots navigate through a room [33]. The data sets used in this experiment are discretized to meet the classical sequential patterns needs.

TABLE V. DIFFERENT DATA SETS USED FOR THE EXPERIMENTS.

Data Set	**Items Items**	**Avg. size of itemsets**	**Avg. size of sequences**	**# of data sequences**
ROBOT	102	1	20	5456
PIONEER	350	1	72	159

TABLE VII. CONFUSION MATRIX FOR DATA SET $ROBOT$ WITH $\sigma = 0.05$ AND $\delta = 20$.

	Predicted A	**Predicted B**	**Unknown**
Class A	1745	150	310
Class B	161	1936	0

In this set of experiments, we analyze the effectiveness of the classification in terms of *accuracy* and *earlyness* cost. The data sets $ROBOT$ and $PIONEER$ are first mined for δ-free sequential patterns, then the early-prediction sequence classifier is built upon carefully selected δ-strong rules as discussed previously. Table VI presents the different extraction results and the classification results. For the $ROBOT$ data set, the optimal results are obtained with a minimal support of 0.05 and $\delta = 20$. The average prediction costs in this precise case is 8.7043 meaning that the classifier needs in average to read 9 items before predicting the sequence's class. Notice that in average in this data set a sequence contains 24 items, so our classifier needs a little bit more than the third of the sequence to be able to fire its prediction. Table VII presents the confusion matrix built from the evaluation of this data set with $\sigma = 0.05$ and $\delta = 20$ and 2 classes. The important thing is to notice that the third column contains all sequences that did not get classified by any rule. This may be caused by: (i) a high support threshold that is not low enough to find sequential patterns that will cover more data sequences or (ii) restrictive rule selection (via a black listing scheme) that favor some non-optimal covering leading to accuracy losses. The last experiment is presented in order to illustrate the weak point of our approach. Because the data set $PIONEER$ contains a few but long data sequences, the minimal support that can be used to extract sequential patterns is indeed very high : 0.55. Here the δ value is attaining a critical case of almost $\frac{\sigma}{2}$ which generates rules of confidence 50% with a high rate of conflicts. This explains the very poor accuracy of 0.20625. Furthermore, any lower value of δ is not enough to generate an interesting set of δ-strong rules.

TABLE VI. DIFFERENT CLASSIFICATION RESULTS WITH VARYING δ, σ PARAMETERS.

Data Set	σ	δ	# frequent δ-free	# δ-strong rules	# classifier rules	Early pred. cost	Avg. pred. cost per sequence	Accuracy
$ROBOT$	0.2	1	13	3	3	28496	6.6238	0.4867
$ROBOT$	0.1	40	100	19	19	36146	8.4021	0.6052
$ROBOT$	0.05	20	695	320	292	37446	8.7043	0.8556
$PIONEER$	0.55	170	189	5	3	2327	14.54375	0.20625

V. RELATED WORK

Since the key paper of *Mannila and Toivonen* [34], subsequent research has focused on building *concise* representations for frequent patterns. That is, lossless subsets of frequent patterns with the same expressiveness power. However, most of the work (and results) focused on frequent itemset patterns (i.e., sets of items), mainly because of the deeper relations and understanding already developed in various mathematical fields like set theory, combinatorics, and Galois connections in order theory. Indeed, researchers introduced closed sets [35], free sets [22], and non-derivable itemsets [36]. However, finding concise representations for structured data is a more challenging exercise as pointed out by the authors of [23]. Closed patterns were successfully extended to sequence in [26], [37], [38]. Recently, generator sequences were proposed in [24], [25], [39]. Subsequently, a general framework for minimal pattern mining was introduced by *Soulet et al.* in [40], but this was limited to chains (i.e., sequences without gaps). Our first proposition in this paper (the DEFFED algorithm) is a generalization of sequence generators that are a particular case of δ-free sequences ($\delta = 0$). Moreover, DEFFED is able to discover δ-free frequent sequences of itemsets whereas work about sequence generators are limited to sequence of items.

The classification of sequence data has been extensively studied [14]–[16], [41]. Most previous work has combined sequence feature selection and common classification methods. For instance, the authors of [15], [16] study the prediction of outer membrane proteins from protein sequences by combining several feature selection methods and support vector machines (SVMs) [42]. Other methods are based on Hidden Markov Models (HMM) which are stochastic generalizations of finite-state automata have been proposed for sequence classification [43], [44]. In a paper by *Zaki et al.* [45], the authors proposed the VOGUE method, which addresses the main limitations of HMMs. VOGUE is a two steps method: it first, mines sequential patterns and then builds HMMs based on the extracted features. Some criteria for feature selection are proposed in [14], [41]. The authors of [14] use the confidence measure to quantify the features. Our work can lead to a generalization of this previous work by allowing the use of any frequency-based measure. In a similar way, *Grosskreutz et al.* [41] showed the utility of minimum patterns for classification, but their approach is restricted to items for binary classifications. Other methods of classification rely on string kernels to extend methods such as SVMs [42] to be able to handle sequence data [46], [47]. However these approaches focus more on strings than general sequences, as in our work.

VI. CONCLUSION

We have studied in this paper a new type of patterns in sequential data, the δ-free sequential patterns. These patterns are the shortest sequences of equivalence classes on the support w.r.t the δ threshold. We described the anti-monotonicity property which does not hold in sequential data and we presented novel pruning properties based on the projected databases of the sequences. A correct and complete algorithm to mine these δ-free sequential patterns is tested and we show that the number of extracted patterns is greatly reduced compared to a frequent or closed patterns extraction approach. The δ-free sequential patterns are also extracted more efficiently in term of time and memory consumption.

We have then showed how δ-free patterns can be employed to address two problems related to sequence classification, namely feature selection in a statistical approach and early prediction in a symbolic approach. First, using the DEFFED algorithm for feature selection allows to explore the entire feature space and to retain only promising patterns. This method results in smaller and more interpretable classification while at the same time it contains richer information than simpler feature selection methods. Second, we have shown that δ-free patterns can be used to identify δ-strong symbolic classification rules with minimal prefix, which turn out to be highly efficient for early prediction by maximizing the earliness constraint.

In future work, we will investigate the use of δ-free sequential patterns in natural language processing problems in order to incorporate more information into the classification process, such as part-of-speech tags.

ACKNOWLEDGMENTS

This work is supported by the French National Research Agency (ANR) as part of the project Hybride ANR-11-BS02-002 and the "Investissements d'Avenir" program (reference: ANR-10-LABX-0083).

REFERENCES

[1] Z. Xing, J. Pei, and E. Keogh, "A brief survey on sequence classification," *SIGKDD Explor. Newsl.*, vol. 12, no. 1, pp. 40–48, Nov. 2010. [Online]. Available: http://doi.acm.org/10.1145/1882471.1882478

[2] M. Deshpande and G. Karypis, "Evaluation of techniques for classifying biological sequences," in *Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, ser. PAKDD '02. London, UK, UK: Springer-Verlag, 2002, pp. 417–431. [Online]. Available: http://dl.acm.org/citation.cfm?id=646420.693671

[3] L. Wei and E. Keogh, "Semi-supervised time series classification," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '06. New York, NY, USA: ACM, 2006, pp. 748–753.

[4] C. Aggarwal and C. Zhai, "A survey of text classification algorithms," in *Mining Text Data*, C. C. Aggarwal and C. Zhai, Eds. Springer US, 2012, pp. 163–222.

[5] F. Sebastiani, "Machine learning in automated text categorization," *ACM Comput. Surv.*, vol. 34, no. 1, pp. 1–47, Mar. 2002.

[6] H. Liu and H. Motoda, *Computational Methods of Feature Selection (Chapman & Hall/Crc Data Mining and Knowledge Discovery Series)*. Chapman & Hall/CRC, 2007.

[7] N. A. Chuzhanova, A. J. Jones, and S. Margetts, "Feature selection for genetic sequence classification." *Bioinformatics*, vol. 14, no. 2, pp. 139–143, 1998.

[8] C. S. Leslie, E. Eskin, and W. S. Noble, "The spectrum kernel: A string kernel for svm protein classification," in *Pacific Symposium on Biocomputing*, 2002, pp. 566–575.

[9] Z. Xing, J. Pei, G. Dong, and P.-S. Yu, "Mining sequence classifiers for early prediction," in *SDM*, 2008, pp. 644–655.

[10] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Proceedings of the 11th International Conference on Data Engineering (ICDE 95)*, Tapei, Taiwan, 1995, pp. 3–14.

[11] M.-R. Berthold, K. Morik, and A. Siebes, Eds., *Parallel Universes and Local Patterns*, vol. 07181. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2007.

[12] A. Knobbe, B. Crémilleux, J. Fürnkranz, and M. Scholz, "From local patterns to global models: The lego approach to data mining," in *From Local Patterns to Global Models: Proceedings of the ECML PKDD 2008 Workshop*, 2008, pp. 1–16.

[13] M. J. Zaki, C. D. Carothers, and B. K. Szymanski, "VOGUE: A variable order hidden markov model with duration based on frequent sequence mining," *ACM Transactions on Knowledge Discovery in Data*, vol. 4, no. 1, Jan 2010.

[14] N. Lesh, M.-J. Zaki, and M. Ogihara, "Mining features for sequence classification," in *KDD*, 1999, pp. 342–346.

[15] K.-J. Park and M. Kanehisa, "Prediction of protein subcellular locations by support vector machines using compositions of amino acids and amino acid pairs," *Bioinformatics*, vol. 19, no. 13, pp. 1656–1663, 2003.

[16] R. She, F. Chen, K. Wang, M. Ester, J.-L. Gardy, and F.-S.-L. Brinkman, "Frequent-subsequence-based prediction of outer membrane proteins," in *KDD*, 2003, pp. 436–445.

[17] T.-P. Exarchos, M.-G. Tsipouras, C. Papaloukas, and D.-I. Fotiadis, "An optimized sequential pattern matching methodology for sequence classification," *Knowl. Inf. Syst.*, vol. 19, no. 2, pp. 249–264, 2009.

[18] V.-S. Tseng and C.-H. Lee, "Effective temporal data classification by integrating sequential pattern mining and probabilistic induction," *Expert Syst. Appl.*, vol. 36, no. 5, pp. 9524–9532, 2009.

[19] J. Li, H. Li, L. Wong, J. Pei, and G. Dong, "Minimum description length principle: Generators are preferable to closed patterns," in *AAAI*, 2006.

[20] J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M. Hsu, "Mining sequential patterns by pattern-growth: The prefixspan approach," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 11, pp. 1424–1440, 2004.

[21] H. Mannila, H. Toivonen, and A.-I. Verkamo, "Discovery of frequent episodes in event sequences," *Data Min. Knowl. Discov.*, vol. 1, no. 3, pp. 259–289, 1997.

[22] J.-F. Boulicaut, A. Bykowski, and C. Rigotti, "Free-sets: A condensed representation of boolean data for the approximation of frequency queries," *Data Min. Knowl. Discov.*, vol. 7, no. 1, pp. 5–22, 2003.

[23] C. Raïssi, T. Calders, and P. Poncelet, "Mining conjunctive sequential patterns," *Data Min. Knowl. Discov.*, vol. 17, no. 1, pp. 77–93, 2008.

[24] C. Gao, J. Wang, Y. He, and L. Zhou, "Efficient mining of frequent sequence generators," in *WWW*, 2008, pp. 1051–1052.

[25] D. Lo, S.-C. Khoo, and J. Li, "Mining and ranking generators of sequential patterns," in *SDM*, 2008, pp. 553–564.

[26] J. Wang, J. Han, and C. Li, "Frequent closed sequence mining without candidate maintenance," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 8, pp. 1042–1056, 2007.

[27] J. Wang and J. Han, "Bide: Efficient mining of frequent closed sequences," in *Proceedings of the 20th International Conference on Data Engineering*, ser. ICDE '04. Washington, DC, USA: IEEE Computer Society, 2004, pp. 79–. [Online]. Available: http://dl.acm.org/citation.cfm?id=977401.978142

[28] J. Pei, J. Han, B. Mortazavi-asl, H. Pinto, Q. Chen, U. Dayal, and M. chun Hsu, "Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth," in *Proceedings of the 17th International Conference on Data Engineering*, ser. ICDE '01. Washington, DC, USA: IEEE Computer Society, 2001, pp. 215–224. [Online]. Available: http://dl.acm.org/citation.cfm?id=876881.879716

[29] K. Nigam, "Using maximum entropy for text classification," in *In IJCAI-99 Workshop on Machine Learning for Information Filtering*, 1999, pp. 61–67.

[30] T. Lavergne, O. Cappé, and F. Yvon, "Practical very large scale CRFs," in *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics, July 2010, pp. 504–513. [Online]. Available: http://www.aclweb.org/anthology/P10-1052

[31] B. Crémilleux and J.-F. Boulicaut, "Simplest rules characterizing classes generated by delta-free sets," in *In Proc. of the 22nd BCS SGAI International Conference on Knowledge Based Systems and Applied Artificial Intelligence*, 2002.

[32] P. Sun, S. Chawla, and B. Arunasalam, "Mining for outliers in sequential databases," in *SDM*, 2006.

[33] A. Frank and A. Asuncion, "UCI machine learning repository," 2010. [Online]. Available: http://archive.ics.uci.edu/ml

[34] H. Mannila and H. Toivonen, "Multiple uses of frequent sets and condensed representations (extended abstract)," in *KDD*, 1996, pp. 189–194.

[35] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal, "Discovering frequent closed itemsets for association rules," in *ICDT'99*, 1999, pp. 398–416.

[36] T. Calders and B. Goethals, "Mining all non-derivable frequent itemsets," in *PKDD*, 2002, pp. 74–85.

[37] X. Yan, J. Han, and R. Afshar, "Clospan: Mining closed sequential patterns in large databases," in *SDM*, 2003.

[38] D. Fradkin and F. Mrchen, "Margin-closed frequent sequential pattern mining," in *In Proceedings of Useful Patterns Workshop, Fifteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2010.

[39] E. Baralis, S. Chiusano, R. Dutto, and L. Mantellini, "Compact representations of sequential classification rules." in *Data Mining: Foundations and Practice*, ser. Studies in Computational Intelligence, 2008.

[40] A. Soulet and F. Rioult, "Efficiently depth-first minimal pattern mining," in *Advances in Knowledge Discovery and Data Mining*, ser. Lecture Notes in Computer Science, V. Tseng, T. Ho, Z.-H. Zhou, A. Chen, and H.-Y. Kao, Eds. Springer International Publishing, 2014, vol. 8443, pp. 28–39. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-06608-0_3

[41] H. Grosskreutz, B. Lang, and D. Trabold, "A relevance criterion for sequential patterns." in *ECML/PKDD (1)*, 2013.

[42] T. Joachims, "Text categorization with suport vector machines: Learning with many relevant features," in *Proceedings of the 10th European Conference on Machine Learning*, ser. ECML '98. London, UK, UK: Springer-Verlag, 1998, pp. 137–142. [Online]. Available: http://dl.acm.org/citation.cfm?id=645326.649721

[43] R. Durbin, *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998. [Online]. Available: http://books.google.fr/books?id=R5P2GlJvigQC

[44] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," in *PROCEEDINGS OF THE IEEE*, 1989, pp. 257–286.

[45] M. J. Zaki, C. D. Carothers, and B. K. Szymanski, "VOGUE: A variable order hidden markov model with duration based on frequent sequence mining," *ACM Transactions on Knowledge Discovery in Data*, vol. 4, no. 1, p. Article 5, Jan 2010.

[46] C. Watkins, "Dynamic alignment kernels," in *Advances in Large Margin Classifiers*. MIT Press, 1999, pp. 39–50.

[47] C. S. Leslie, E. Eskin, and W. S. Noble, "The spectrum kernel: A string kernel for svm protein classification." in *Pacific Symposium on Biocomputing*, 2002, pp. 566–575. [Online]. Available: http://dblp.uni-trier.de/db/conf/psb/psb2002.html#LeslieEN02