Sequential Patterns for Text Categorization

Simon Jaillet, Anne Laurent, Maguelonne Teisseire

Intelligent Data Analysis 9 (2006) 1–16 1
IOS Press

Sequential patterns for text categorization

S. Jaillet, A. Laurent and M. Teisseire
LIRMM-CNRS – Université Montpellier 2, 161 rue Ada, 34392 Montpellier Cedex 5 France
E-mail: {jaillet,laurent,teisseire}@lirmm.fr

Received 22 June 2005
Revised 25 August 2005
Accepted 12 November 2005

Abstract. Text categorization is a well-known task based essentially on statistical approaches using neural networks, Support Vector Machines and other machine learning algorithms. Texts are generally considered as bags of words without any order. Although these approaches have proven to be efficient, they do not provide users with comprehensive and reusable rules about their data. Such rules are, however, very important for users to describe trends in the data they have to analyze. In this framework, an association-rule based approach has been proposed by Bing Liu (CBA). We propose, in this paper, to extend this approach by using sequential patterns in the SPaC method (Sequential Patterns for Classification) for text categorization. Taking order into account allows us to represent the succession of words through a document without complex and time-consuming representations and treatments such as those performed in natural language and grammatical methods. The original method we propose here consists of mining sequential patterns in order to build a classifier. We experimentally show that our proposal is relevant, and that it is very interesting compared to other methods. In particular, our method outperforms CBA and provides better results than SVM on some corpus.

Keywords: Text mining, categorization, sequential patterns, SPaC

1. Introduction

Automatic text classification goes back at least to the 1960s [24]. But with the growing volume of available digital documents, automatic classification has been extensively addressed through research in the past few years to define efficient and scalable methods [33,39]. In this domain, two distinct types of approaches have been proposed: supervised and unsupervised classification. In supervised classification (also known as categorization), categories are defined by an expert, while they are automatically learned in the other case (also called clustering) [14,31].

In this setting, the goal is to define a function which associates texts with categories. The learning step involves automatically defining this function using a training set. This training set consists of texts for which the category is known. Then this function can be used to associate a category (e.g. politics or sport) to a new text that has never been processed. The more accurate the automatic decision, the better the classifier.

Currently, the best classifiers are mostly based on the statistical text representation TF-IDF (Term Frequency, Inverse Document Frequency) [31] and machine learning algorithms such as neural networks or Support Vector Machines (SVM). However, most of these methods do not provide understandable descriptions of the extracted knowledge. In order to cope with this problem, an association-rule based approach was first proposed by Bing Liu (CBA) [21], which has subsequently been enhanced in [5,8,12, 16,19,37], etc.

All these methods consider each text as a so-called *bag of words* where no order between words is taken into account for categorization. This textual representation has proven to be useful and almost as efficient as complex representations which require time-consuming methods like syntactic analysis. It is thus interesting to investigate methods that take order into account while remaining scalable. For this purpose, we study sequential patterns. Sequential Patterns aim at discovering temporal relationships between facts embedded in a database. In a market basket analysis problem, sequential patterns refer to rules like *a customer who bought a TV together with a DVD player later bought a recorder*. In this framework, the databases being considered consist of customer transactions, recording the *items* bought at some *dates* by customers (*clients*).

In this paper, we thus propose to extend the CBA method by taking the order into account. In our SPaC (Sequential Patterns for Categorization) approach, the order is considered by using sequential patterns instead of association rules. Sequential patterns have, in this framework, three main advantages: first they provide understandable rules (contrary to SVM, Rocchio, naive Bayes, ...). Secondly they allow trend analysis, as shown in [18]. Thirdly, they extract patterns that are more precise and informative than association rules.

In the original text classification method using sequential patterns we propose here, sentences are distinguished and ordered in each text. This means that the text is considered as being an ordered list of sentences. Each sentence is considered as being an unordered set of words. If we compare the market basket analysis problem to our approach, then a text plays the role of a client; the sentences from a text play the role of all the transactions for this text; the position of the sentence within the text plays the role of the date; and the set of words from a sentence plays the role of a list of items bought.

Experiments show that sequential pattern-based classification with SPaC is very efficient, particularly when Support Vector Machines do not perform well. Our approach is not only evaluated using accuracy, but also using the precision and recall measures merged into the F_β-measure [30]. These measures have indeed proven to be more relevant for comparing text classification methods [33].

The paper is organized as follows. Section 2 presents the background of the problem addressed by introducing sequential patterns and textual representations. Section 3 details existing methods that deal with text mining with "frequent patterns" and "sequential patterns". Section 4 details our method based on Sequential Patterns (SPaC). Section 5 shows that our method performs well on datasets in French and English. Finally, Section 6 summarizes the paper and presents future work.

2. Problem statement

First, we introduce the categorization problem. Secondly, we formulate the concept of sequence mining by summarizing the formal description of the problem introduced in [3] and extended in [35].

2.1. Textual representation and categorization

Text categorization is the task of assigning a boolean value to each pair (document, category). For instance, text categorization is used to automatically determine whether a text belongs to the *politics* or *sport* category. In order to build such automatic classifiers, a textual database is considered. In this database, the class to which each text belongs is known. The textual database is partitioned into two databases. The first sub-database is a training set and the second one is a test set used in order to evaluate the classifier quality.

In usual methods, texts are represented as *bags of words* [33], meaning that the order is not considered. Each document is represented by a vector where each component is a word weighted by a numerical value. The most used weight is $TF\text{-}IDF$ (Term Frequency – Inverse Document Frequency) [31]. For a word w, we have:

$$tfidf(w) = tf(w).\log\frac{N}{df(w)}$$

where $tf(w)$ is the number of occurrences of w in the document, $df(w)$ if the number of documents containing w and N is the total number of documents. The weight $tfidf(w)$ thus represents the relative importance of the word in the document.

These vectors describing documents are used to extract knowledge using common algorithms such as k-nearest neighbors, SVM, naive Bayes on the training set.

2.2. Mining sequential patterns

Let DB be a set of customer transactions where each transaction T consists of customer-id, transaction time and a set of items involved in the transaction.

Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of literals called $items$. An $itemset$ is a non-empty set of items. A sequence s is a set of itemsets ordered according to their timestamp. It is denoted by $< s_1 s_2 \ldots s_p >$ where s_j, $j \in 1..n$, is an itemset. An n-sequence is a sequence of n items (or of length n). For example, let us consider a given customer who purchased items 1, 2, 3, 4, 5, according to the following sequence: $s =<$ (1) (2, 3) (4) (5)$>$. This means that apart from 2 and 3 that were purchased together, i.e. during the same transaction, items in the sequence were bought separately. s is a 5-sequence.

A sequence $< s_1 s_2 \ldots s_p >$ is a sub-sequence of another sequence $< s'_1 s'_2 \ldots s'_m >$ if there are integers $i_1 < i_2 < \ldots i_j \ldots < i_n$ such that $s_1 \subseteq s'_{i_1}, s_2 \subseteq s'_{i_2}, \ldots, s_p \subseteq s'_{i_n}$. For example, the sequence $s' =<$ (2) (5) $>$ is a sub-sequence of s because (2) $\subseteq$ (2, 3) and (5) $\subseteq$ (5). However $<$ (2) (3) $>$ is not a sub-sequence of s since the items were not bought during the same transaction.

All transactions from the same customer are grouped together and sorted in increasing order. They are called a *data sequence*. A support value (*supp*(*s*)) for a sequence gives its number of actual occurrences in DB. Nevertheless, a sequence in a data sequence is taken into account only once to compute the support, even if several occurrences are discovered. In other words, the support of a sequence is defined as the fraction of total distinct data sequences that contain s. In order to decide whether a sequence is frequent or not, a minimum support value (*minSupp*) is specified by the user. A sequence s is said to be *frequent* if the condition $supp(s) \geqslant minSupp$ holds.

Given a database of customer transactions the problem of sequential pattern mining is to find all sequences whose support is greater than a specified threshold (minimum support) [28].

Sequential patterns are usually extracted from a database built on the following scheme: *date, client, items*. For instance, we consider the database of client purchases in a supermarket, as shown in Table 1. Each line (transaction, tuple) from this table corresponds to the set of items bought by the client at the corresponding date.

In this example, Peter has bought the items 1, 2, 3, 4, 5 in the sequence <(1)(2,3)(4)(5)>, meaning that he first bought 1, then he bought 2 together with 3, then he bought 4 and finally he bought 5.

3. Related work

Text mining has been widely investigated [1,4,18,33]. In this section, we focus on text classification and frequent patterns since our method is based on rules.

Table 1
Database of purchases

Client	Date	Items
Peter	04/01/12	TV (1)
Martin	04/02/28	Chocolate(5)
Peter	04/03/02	DVD_Player (2) , Camera (3)
Peter	04/03/12	Printer (4)
Peter	04/04/26	Chocolate (5)

3.1. Classification based on associations: the CBA method

In [21] the authors propose CBA: a text categorization method based on association rules. Contrary to C4.5 [29], CN2 [9], or RIPPER [10], which use heuristic search to learn a subset of the regularities in data to build a classifier, CBA is based on exhaustive search and aims at finding all rules respecting a minsup value.

CBA consists of two parts, a rule generator (CBA-RG), which is based on the well-known Apriori algorithm [2], and a classifier builder (CBA-CB), which is based on generated rules.

3.1.1. CBA-RG

In this first step, each assignment <text, category> is represented by a ruleitem defined by: $\rho =< condset, C_i >$ where condset is a set of items and C_i is a class label[1] Each ruleitem ρ is equivalent to a rule of the type $condset \rightarrow C_i$ where support and confidence are defined by:

$$sup(\rho) = \frac{\text{\#texts from } C_i \text{ matching condset}}{\text{\#texts in D}}$$

$$conf(\rho) = \frac{\text{\#texts in } C_i \text{ matching condset}}{\text{\#texts in D matching condset}}$$

Ruleitems that satisfy the minimum support are called *frequent ruleitems.* If two ruleitems have the same condset, only the one having the highest confidence is chosen as a possible rule (PR). If some ruleitems have the same condset and the same confidence, the PR is then randomly chosen on this set. PRs is then a subset of the frequent ruleitems set determined by the two previous constraints. The set of class association rules (CARs) thus consists of all PRs whose confidence is greater than a minimum confidence.

CARs (classification rules) thus consists of all ruleitems that satisfy the minimum support and minimum confidence levels.

In these approaches, frequent patterns are extracted using a single minimum support threshold. However, categories are not always equi-distributed. It is thus not relevant to consider such a single value. Choosing a relevant minimum support threshold is crucial so that frequent patterns will be relevant for the categorization task. A high support will indeed prevent the system from finding frequent patterns for a small category, while a low support will lead to generation of a huge number of rules, which is not interesting because it will result in overfitting.

Works have been proposed to define a multiple minimum support application (msCBA) [16,22]. In these approaches, ruleitems are extracted using a multiple minimum support strategy. The minimum

[1] In data mining, $condset$ is also called $itemset$.

support level of each category is defined according to the distribution frequency of each category and the user-defined minimum support threshold:

$$minSup_{C_i} = minSup_{user} * freqDistr(C_i)$$

where, $freqDistr(C_i) = \frac{\text{\#texts from } C_i}{\text{\#texts}}$.

3.1.2. CBA-CB

Once all CARs are generated, they are ordered according to the total order described below.

Definition 1. Let r_i and r_j be two classification rules (CARs), $r_i \prec r_j$ if:

- $conf(r_i) > conf(r_j)$;
- or $conf(r_i) = conf(r_j)$ and $supp(r_i) > supp(r_j)$;
- or $conf(r_i) = conf(r_j)$ and $supp(r_i) = supp(r_j)$ and r_i has been generated before r_j;

Let R be the set of CARs and D be the training data. The basic idea of the algorithm is to choose a set of high precedence rules in R to cover D. The categorizer is thus represented by a list of rules $r_i \in R$ ordered according to the total defined above (Definition 1). We thus have:

$$< (r_1, r_2, ..., r_k), C_i >$$

(where C_i is the target category and r_j one of the associated rules).

Each rule is then tested over D. If a rule does not improve the accuracy of the classifier, then this rule and the following ones are discarded from the list.

Once the categorizer has been built, each ordered rule $r_i \in R$ is tested on each new text to classify. As soon as the condset part of a rule is supported by the text, the text is then assigned to the target class of the rule. If no rule is appropriate, then the text is assigned to the default class C_i.

3.2. Enhancements and other approaches

Many other research studies have been performed using association rules to classify. In [16], the authors replace the confidence by the intensity of implication, in CBA, when sorting the rules to build the classifier. According to the author, this new "measure" is powerful when classes are not equally distributed and for low minsup value.

In [23], the authors integrate the CBA method with other methods such as decision trees, naive Bayes, RIPPER, etc. to increase the classification score.

In [8], the authors investigate L^3, i.e. a method for association rule classification. Contrary to CBA-CB which takes only one rule into account, they propose to determine the category of a text by considering several rules that are mixed using majority voting. In order to cope with the huge number of rules, the authors propose a pruning method during extraction of the classification rules using χ^2, as done in [19]. But contrary to most pruning strategies, L^3 performs a "lazy" pruning in order to eliminate only "harmful" rules and not "useful knowledge". In L^3, $maxrules$ stands for the maximum number of rules used to classify new cases. Moreover, rules are separated in two levels in order to increase the classification accuracy. L^3 has also been enhanced by considering several minimum support thresholds for each category in [7].

In [4], association rules are used for partial classification. Partial classification means that the classifier does not cover all cases. In particular, this work is interesting when dealing with missing values.

CAEP [12], LB [27], ADT [37], CMAR [19] are some other existing classification systems using association rules. LB and CAEP are based on rule aggregation rather than rule selection. The particularity of ADT is to prune rules with low support which are considered as meaningless like in [16]. To avoid overfitting rules, ADT also uses a learning strategy based on a decision tree The advantage of CMAR is the categorization policy and the data structure used which allow the user to store a large number of extracted rules.

But all of these methods are hampered by the same problem. Except for msCBA and L^3, all of these methods use a single minsup value. This limitation leads to overlooking minority classes or overfit majority classes (depending on the chosen minsup value). Moreover, most of them are based on an Apriori-like method to extract association rules and the number of generated rules increase dramatically when a low support has to be used. Apart from relatively small numerical datasets from the UCI archives [13], most of these methods are unusable for classification tasks that require low minsup definition (like in text categorization).

ARC-CB [5] proposes a solution for multi-classification (i.e. a text is associated to one or more classes). But no comparison to other association rules based classifiers is performed in order to highlight the impact of the method.

In the text mining framework [18,38], propose to use sequential patterns. In [38], the proposal is based on two methods. The first method is based on the visualization of word occurrences in order to detect sequential patterns. The second method is based on classical methods to extract sequential patterns. However, the authors do not propose a method to classify texts using sequential patterns. Moreover, the texts considered are associated with a date. The corpus consists of 1,170 articles collected over 6 years. This point makes it very different from our proposal and more difficult to apply since texts are rarely associated with a date.

In [18], the authors demonstrate how sequential patterns are useful for text mining. Sequential patterns are used in order to extract trends from textual databases.

In [34,36], the authors propose to use sequential patterns for categorization. However, this approach does not make use of all the power of sequential patterns since the patterns considered consist of lists of items, whereas we aim at considering lists of *itemsets*. Each element from the patterns is indeed only composed of a single morpheme (or n-gram), whereas it would be interesting to consider patterns composed by elements which may be more complex (set of words) and automatically composed.

We thus propose an original method based on sequential patterns for classification. We argue that this method is able to deal with order in texts without being time-consuming. The next section details our approach. Section 5 shows that our method obtains good results compared to others.

4. Sequential patterns for classification: the SPaC method

In this paper, we propose an original method (SPaC) for text classification based on sequential patterns [15]. This method consists of two steps. In the first step, we build sequential patterns from texts. In the second step, sequential patterns are used to classify texts.

Hereafter, we use the notations introduced in Table 3.

4.1. From Texts to sequential patterns

Each text is a set of words. Our method is based on sequential pattern mining. Texts are represented as ordered sets of words using the TF-IDF representation. Each text is thus considered as being the

equivalent of a client. The text consists of a set of sentences. Each sentence is associated with a date (its position in the text). Finally the set of words contained in a sentence corresponds to the set of items purchased by the client in the market basket analysis framework. Table 2 summarizes the two terminologies.

This representation is coupled with a stemming step and a *stop-list*. The stemming step involves replacing each word by its root word. The stop-list prevents the system from learning from noisy words such as "*the, a*".

Some words are discarded by considering the entropy of each stem over the corpus. This method eliminates words that could skew the classifier since they are not discriminant enough. Moreover, this method allows us to apply low supports in the sequential pattern discovery without deteriorating the results. For this purpose, a user-defined threshold is considered. For each word w, we consider its entropy $H(w)$ over all classes C_i defined as:

$$H(w) = -\Sigma C_i[p(w).p(C_i|w).log(p(C_i|w)) + ((1-p(w)).p(C_i|\bar{w}).\log\,(p(C_i|\bar{w})))]$$

In SPaC, sequential patterns are extracted using a multiple minimum support strategy as done in msCBA. This means that a different support is applied for each category C_i.

In our approach, the training set is divided into n training sets, one for each category. Texts are thus grouped depending on their category. Sequential pattern mining algorithms are applied separately on these n databases using the corresponding minimum supports.

For each category, frequent sequential patterns are computed and their supports stored. The support of a frequent pattern is the number of texts containing the sequence of words.

Definition 2. Let $< s_1 \ldots s_p >$ be a sequence. The support of $< s_1 \ldots s_p >$ is defined as:

$$supp(< s_1 \ldots s_p >) = \frac{\#\text{texts matching} < s_1 \ldots s_p >}{\#\text{texts}}$$

Contrary to msCBA, minimum supports are defined automatically in the following way:

(1) the minimum support is set at the lowest value, i.e. one text (for example, if the training set contains 200 texts, the minsup is set to 0.5%).
(2) if the mining step provides more than X rules, the process is started again with a higher minsup value, i.e. on more texts (for example, if the training set contains 200 texts, the minsup is increased by 0.5%).

The use of SPAM [6] to find sequential patterns makes the training step extremely fast. Indeed this step, which extracts all sequential patterns from the training set, takes only a few minutes on a current deskstop.[2] The limited number of rules (i.e. X) will also be detailed in the experiments (Section 5).

Algorithm 1 describes SPaC sequential pattern generation. The SPAM algorithm is used through the $SPMining()$ function in order to find all frequent sequences in the transactional databases (DB) [6].

For instance, the following frequent patterns have been extracted from the "Purchasing-Logistics" category of our French database:

[2] Pentium IV 2.4 GHz, 520 Mo.

Algorithm 1: SPaC rules generation

Data : T_{Train} : the training Set

$\{minSup_{C_i}\}$: the set of minimum supports for each category C_i

X: the maximum number of rules to be generated

Result : SP: the set of sequential patterns

begin

$SEQ \leftarrow \emptyset$; $cust \leftarrow 0$; $date \leftarrow 0$;

foreach *Category* $C_i \in \mathcal{C}$ **do**

foreach *Text* $T_j \in T^{C_i}$ **do**

foreach *Sentence* $S_k \in T_j$ **do**

V_s=TFIDF(Stem(S_k)); // Compute the $TF\text{-}IDF$ vector of the sentence

for $(s = 0; s <\mid V_s \mid; s++)$ **do**

if $V_s[s] > 0$ **then**

$SEQ[C_i][cust][date]$.additem(s);

date++;

$date \leftarrow 0; cust++$;

$cust \leftarrow 0$;

foreach *Category* $C_i \in C$ **do**

repeat

R=SPMining(SEQ[C_i],$minSup_{C_i}$,X); // Abord when #rules > X

until *R is not valid*;

SP[C_i]=R;

end

< (cacao) (ivoir) (abidjan)>
< (blé soja) (maï)>
< (soj)(blé lespin victor)(maï soj)(maï)(grain soj)(soj)>

The first sequential pattern means that some texts contain words *cacao* then *ivoire* then (ivory) *Abidjan* in three different sentences. The second sequential pattern means that some texts contain the words *blé* and *soja* in the same sentence and then *maïs*[3](*mai*). The third sequential pattern means the word *mai* occurs in two successive sentences before the word *grain*.

Experiments have led us to consider a threshold that eliminates about 5 to 10% (according to the Zipf law [32]) of the words. Note that sequential patterns consider multiple occurences of the same word in

[3]In French, *blé* means *wheat* and *maïs* stands for *corn*.

Table 2
Application of the sequential pattern terminology to textual data

Usual Databases		fTextual Databases
client	$\leftrightarrow$	text
item	$\leftrightarrow$	word
items/transaction	$\leftrightarrow$	sentence (set of words)
date	$\leftrightarrow$	position of the sentence

Table 3
Notations

Notation	Meaning
$\mathcal{C} = \{C_1, \ldots, C_n\}$	set of n categories.
$C_i \in \mathcal{C}$	a given category.
$minSup_{C_i}$	user-defined minimum support for category C_i.
T	set of texts.
$T^{C_i} \subseteq T$	set of texts belonging to category C_i.
$T_{Train} = \{(C_i, T^{C_i})\}$	Training set constituted by a set of texts associated with their category.
SEQ	set of sequences found for category C_i, customer c at time t.
SP	table of sequential patterns.
$RuleSP$	table of tuples $(sp_j, C_i, conf_{i,j})$. corresponding to the sequence sp_j, the category C_i and the confidence $conf_{i,j}$ of the rule $sp_j \rightarrow C_i$.

the text, contrary to association rules. Moreover, some frequent co-occurrences can be identified with sequential pattern mining.

4.2. From sequential patterns to categories

Once sequential patterns have been extracted for each category, the goal is to derive a categorizer from the obtained patterns.

This is done by computing, for each category, the *confidence* of each associated sequential pattern. To solve this problem, a rule γ is generated in the following way:

$$\gamma :< s_1 \ldots s_p > \rightarrow C_i$$

where $< s_1 \ldots s_p >$ is a sequential pattern for category C_i. This rule means that if a text contains s_1 then $s_2 \ldots$ then s_p then it will belong to category C_i. Each rule is associated with its confidence level, indicating the extent to which the sequential pattern is characteristic of this category:

$$conf(\gamma) = \frac{\text{\#texts from } C_i \text{ matching} < s_1 \ldots s_p >}{\text{\#texts matching} < s_1 \ldots s_p >}$$

Rules are sorted depending on their confidence level and the size of the associated sequence. When considering a new text to be classified, a simple categorization policy is applied: the K rules having the best confidence level and being supported are applied. The text is then assigned to the class mainly obtained within the K rules. This method is the same as majority voting in [8]. If two categories obtain the same score, a random choice is made. This prevents the system from always choosing the same category. The SPaC classifier step (SPaC-C) is described in Algorithm 2.

Algorithm 2: SPaC-C Method

Data : T_{Test}: the test Set
KFS: the K First Satisfied Parameters
SP: the sequential Pattern Table from SPaC-RG

begin

$nb \leftarrow 1$;
foreach *Category* $C_i \in C$ **do**
  foreach $sp_j \in SP[C_i]$ **do**
    $RuleSP[nb] \leftarrow (sp_j, C_i, conf(sp_j \rightarrow C_i)$;
    $nb++$;
Sort $RuleSP$ by rule confidence and size of sequence ;
$nfs \leftarrow 0$; $classable \leftarrow 0$;
foreach *Text* $T_k \in T_{Test}$ **do**
  foreach *rule* $(sp_j \rightarrow C_i) \in RuleSP$ **do**
    if T_k *supports* SP_j **then**
      T_k.score[C_i]++; $classable \leftarrow 1$; nfs++ ;
      if $nfs \geq KFS$ **then** break
  if *classable* **then**
    Set T_k to the Most Valued Category;
  classable $\leftarrow 0$; nfs $\leftarrow 0$;

end

5. Experiments

Experiments are conducted on three databases. The first two ones are the well-known English databases 20 Newsgroups (with 30% of training) and Reuters [13]. The third one is a real database. It describes French texts (news) with 8,239 texts divided into 28 categories. For this corpus, a training set of 33% was used.

Experiments compare our approach to results obtained with CBA and SVM. Sequential Patterns are mined using SPAM [6]. Table 4 details these results. Comparisons are based on the F_β measure [33]. This measure allows us to combine recall and precision for a global evaluation. The F_β measure is thus more relevant than accuracy since accuracy does not take into account the case when a text is not classified. This leads us to consider that classifying no text (thus never making errors) would have an accuracy of near 100%! Accuracy was taken as the reference measure in [8,20]. However, for the reasons presented above, we argue that this is not as relevant as relying on recall and precision, as mentioned in [33]

Table 4
Comparison of SPaC, msCBA and SVM tested on three different corpuses

	French news			Reuters			20 Newsgroups		
	SPaC	msCBA	SVM	SPaC	msCBA	SVM	SPaC	msCBA	SVM
$F1^M$	0.461	0.367	**0.485**	0.322	0.082	**0.500**	**0.452**	0.423	0.423
$F1^\mu$	**0.497**	0.401	0.486	0.694	0.679	**0.840**	**0.494**	0.436	0.455
Acc.	0.963	0.956	**0.969**	0.992	0.992	**0.996**	**0.946**	0.941	0.941
#Rules	31060	315	–	80985	640	–	19938	642	–

Table 5
Results of SPaC on the French news corpus

Minsup allowed	Acc.	$F1^M$	$F1^\mu$	#Rules	Time
1%	0.963	0.461	0.497	31060	441 s
3%	0.963	0.455	0.495	29931	414 s
4%	0.963	0.441	0.488	12227	271 s
5%	0.962	0.435	0.479	5328	221 s
6%	0.962	0.433	0.477	3019	208 s
12%	0.957	0.352	0.390	394	181 s
21%	0.958	0.262	0.320	63	181 s
49%	0.963	0.018	0.033	1	180 s

Definition 3. The F_β measure is defined as follows:

$$F_\beta = \frac{(\beta^2+1)\pi_i\rho_i}{\beta^2\pi_i + \rho_i}$$

where ρ stands for the recall, and π stands for the precision. This measure is computed for each class C_i.

Definition 4. Precision and recall are defined as follows:

$$\pi_i = \frac{TP_i}{TP_i + FP_i}, \rho_i = \frac{TP_i}{TP_i + FN_i}$$

where TP_i, FP_i, FN_i stand, respectively, for the number of texts in the class C_i which are correctly classified (True Positive), the number of texts mistakenly put in class C_i (False Positive), the number of texts mistakenly put in a different class from C_i (False Negative).

Accuracy is defined as follows:

Definition 5. Accuracy:

$$\text{Accuracy}_i = \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i}$$

In order to evaluate the classifier for all classes, we consider *Micro-averaging* (μ) and *Macro-averaging* (M) [33,39], defined as follows:

Definition 6. Macro-averaging and Micro-averaging

$$\hat{\pi}^M = \frac{\sum_{i=1}^{|C|}\hat{\pi}_i}{|C|}, \hat{\rho}^M = \frac{\sum_{i=1}^{|C|}\hat{\rho}_i}{|C|}$$

Table 6
Results of SPaC on the 20 Newsgroups corpus

Minsup allowed	Acc.	$F1^M$	$F1^\mu$	#Rules	Time
1%	0.946	0.452	0.494	19938	594 s
3%	0.946	0.446	0.491	13246	480 s
4%	0.945	0.428	0.478	9441	446 s
6%	0.942	0.386	0.441	4950	400 s
12%	0.942	0.275	0.348	1209	250 s
21%	0.943	0.148	0.231	32	178 s
49%	0.942	0.050	0.079	2	177 s

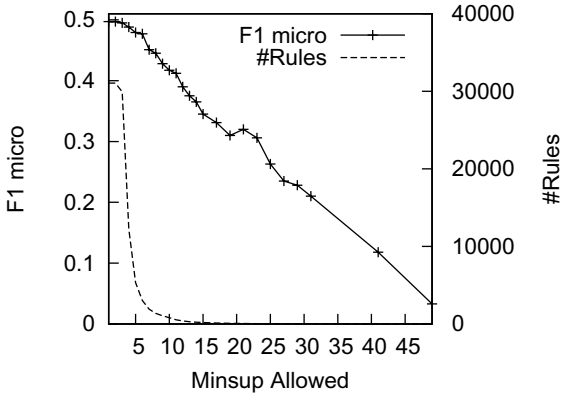

Fig. 1. Results of SPaC on the French news corpus.

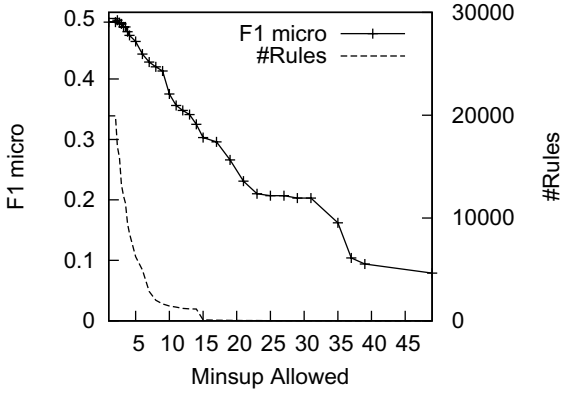

Fig. 2. Results of SPaC on the 20 Newsgroups corpus.

$$\hat{\pi}^\mu = \frac{\sum_{i=1}^{|C|} VP_i}{\sum_{i=1}^{|C|}(VP_i + FP_i)}, \hat{\rho}^\mu = \frac{\sum_{i=1}^{|C|} VP_i}{\sum_{i=1}^{|C|}(VP_i + FN_i)}$$

Micro-averaging gives the same importance to each document, contrary to *macro-averaging* which computes the average class by class (thus placing more importance on small categories).

In this paper, we consider that recall and precision have the same importance. We thus have: $\beta = 1$ for the F_β measure.

Table 7
Results of SPaC on the Reuters corpus

Minsup allowed	Acc.	$F1^M$	$F1^\mu$	#Rules	Time
1%	0.992	0.322	0.694	80985	196 s
2%	0.992	0.307	0.691	79556	172 s
13%	0.991	0.283	0.619	27931	147 s
30%	0.989	0.208	0.545	6929	45 s
50%	0.987	0.216	0.432	791	22 s
92%	0.986	0.154	0.148	45	19 s
99%	0.986	0.020	0.030	23	16 s

SVM results are obtained using a linear kernel (no better result is provided by a more complex kernel) with SVMLight [17]. The supports chosen here are the values providing the best results while remaining computable (too low supports lead to a very time-consuming application). SPaC is applied with $K = 10$ rules taken into account when classifying new documents. The automatic minsup definition is done by limiting the mining process to $X = 3000$ rules per category. If SPAM exceeds this limit then the mining process is started again with a higher minsup value. The experiments showed that a larger limit did not provide significant enhancement during the evaluation step.

In this work, we argue that obtaining understandable knowledge is as (or even more) important than getting the most accurate classifier. For this reason, comparisons are essentially studied between CBA and SPaC. CBA is tested with the msCBA version of the algorithm, using the best results we obtained when testing different supports. The results show that SPaC is always better than CBA. This is due to the fact that SPaC is able to use more specific rules to classify. SPaC is similar to SVM for French texts, but obtains better performance than SVM when dealing with the 20 Newsgroups database. Nevertheless, SVM is still the best classifier on the Reuters corpus. But this corpus had particularities. Indeed, 2 categories (over 90) contain 2/3 of the train test. Moreover, Table 7 shows that few rules with a very high support (>90%) alone provide a great part of the score.

Up to here, the automatic definition of minsup in SPaC provides a lot of rules compared to CBA. In the following experiments, we reduce the number of categorization rules for SPaC by increasing the start minsup value (in the automatic minsup definition process).

Tables 5, 6 and 7 together with Figs 1, 2 and 3 show the number of rules used for categorization and the associated performance in the $F1^\mu$ measure. These experiments show the stability of $F1^\mu$, even when a lot of rules are pruned. On the other hand, SPaC probably reaches almost its best performance with $X = 3,000$ since it requires an exponential number of rules to increase its performaces. Currently, CBA is not able to increase the number of generated rules to increase its performance. Indeed, CBA is based on a hard rule pruning strategy.

Moreover, with a slight drop in performance, the number of generated sequential patterns is small enough to be usable by an expert or for a trend analysis or for manually tuning the classifier.

In Fig. 4, we compare the results obtained for the $F1$ measure regarding the number of rules considered for the choice of class. Experiments have shown that $K = 10$ provides good results (similar to [8] where best results correspond to $maxrules = 9$). Experiments are done (1) with order: the sequential pattern (sequence) is supported by the text (2) without order: the sub-sequences are supported by the text in an unspecified order. The corresponding sentences are in the document but are not ordered as in the sequential pattern. We note that the results are always better when order is taken into account.

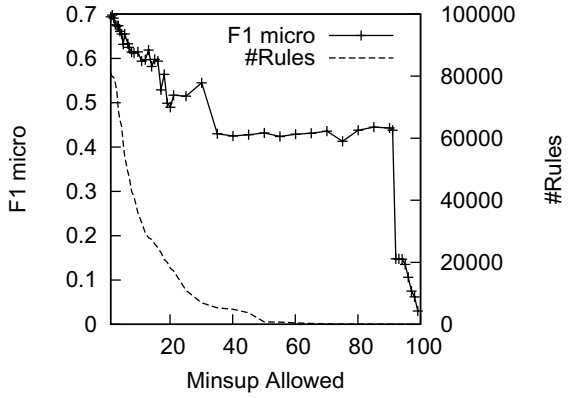

Fig. 3. Results of SPaC on the Reuters corpus.

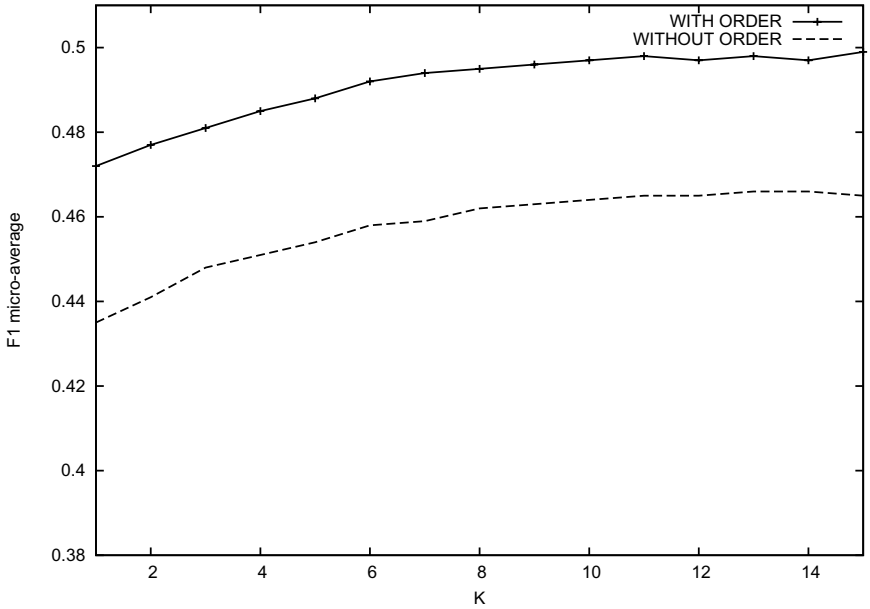

Fig. 4. SPaC: $F1$ as a function of the number of rules considered.

6. Conclusion and further work

In this paper, we address the problem of text categorization using sequential patterns. In our framework, texts are represented by TF-IDF vectors, and each category is associated with a set of sequential patterns. When classifying new data, a text is matched to a category depending on the number of sequential patterns involved. The corresponding category is determined using majority voting. Even if SVM have proven to be efficient in such a task, we argue that it is very important to provide users with understandable knowledge about their data. In this framework, sequential patterns are well-adapted. They provide rules that are used for classification. We show that this approach is efficient and relevant, in particular when SVM do not perform well.

Moreover, the method we propose is simple and adaptable to drifting concepts since it is possible to

update sequential patterns without performing the whole process using incremental sequential pattern mining [25]. This possibility is of great importance for text categorization, especially for the automatic analysis of news which is a very fast and variable area. This is thus a first step towards an On-Line Classification Process (OLCP) using sequential patterns.

Future works include the integration of our approach for different foreign languages in order to determine how important order is for each language. We are working on the automatic definition of the best number of rules to take into account (the K parameter of our method). Our approach may also be enhanced by mining generalized sequential patterns [3]. This framework allows us to integrate time constraints, as shown in [26]. Finally, we aim at integrating muti-level sequential patterns, as proposed in [7]. Very specific rules can thus be kept without damaging the classifier performances. In this framework, we argue that it is interesting to build a very compact set of rules (rules where the left-hand part is as short as possible). For this purpose, we aim at extending studies on δ-free sets [11] to sequential patterns.

References

[1] R. Agrawal, R.J. Bayardo, Jr. and R. Srikant, *Athena: Mining-Based Interactive Management of Text Databases*, In Proc. of the 7rd Int. Conf. on Extending Database Technology, (EDBT'00), Springer, March 2000, 365–379.

[2] R. Agrawal and R. Srikant, Fast Algorithms for Mining Generalized Association Rules, in: *Proc. 20th Int. Conf. Very Large Data Bases*, (*VLDB'94*), J.B. Bocca, M. Jarke and C. Zaniolo, eds, Morgan Kaufmann, 1994, pp. 487–499.

[3] R. Agrawal and R. Srikant, *Mining Sequential Patterns*, In Proc. of the 11th Int. Conf. on Data Engineering (ICDE'95), Tapei, Taiwan, March 1995. IEEE Computer Society Press, 3–14.

[4] K. Ali, S. Manganaris and R. Srikant, *Partial Classification Using Association Rules*, In Proc. of the 3rd Int. Conf. on Knowledge Discovery and Data Mining, (KDD'97), CA, USA, August 1997. AAAI Press, 115–118.

[5] M.-L. Antonie and O. Zaiane, *Text Document Categorization by Term Association*, In Proc. of the 2002 IEEE Int. Conf. on Data Mining (ICDM'02), December 2002, 19–26.

[6] J. Ayres, J. Gehrke, T. Yiu and J. Flannick, *Sequential Pattern Mining Using Bitmaps*, In Proc. of the 8th Int. Conf. on Knowledge Discovery and Data Mining, (KDD'02), July 2002.

[7] E. Baralis, S. Chiusano and P. Garza, *On support thresholds in associative classification*, In Proc. of the 2004 ACM Symposium on Applied Computing (SAC'04), ACM Press, March 2004, 553–558.

[8] E. Baralis and P. Garza, *Majority Classification by Means of Association Rules*, In Proc. of the 7th European Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD'03), Springer, September 2003, 35–46.

[9] P. Clark and T. Niblett, The CN2 Induction Algorithm, *Machine Learning* **3** (March 1989), 261–283.

[10] W. Cohen, *Fast Effective Rule Induction*, In Proc. of the 12th Int. Conf. on Machine Learning, (ICML'95), CA, USA, July 1995. Morgan Kaufmann, 115–123.

[11] B. Cremilleux and J.F. Boulicaut, *Simplest rules characterizing classes generated by delta-free sets*, In Proc. of the 22nd BCS SGAI Int. Conf. on Knowledge Based Systems and Applied Artificial Intelligence, (ES'02), Springer, December 2002, 33–46.

[12] G. Dong, X. Zhang, L. Wong and J. Li, *CAEP: Classification by aggregating emerging patterns*, In Proc. of the 2nd Int. Conf. on Discovery Science, (DS'99), December 1999, 30–42.

[13] S. Hettich and S.D. Bay, *The UCI KDD Archive*, [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Department of Information and Computer Science., 1999.

[14] M. Iwayama and T. Tokunaga, *Cluster-based text categorization: a comparison of category search strategies*, In Proc. of SIGIR-95, 18th ACM Int. Conf. on Research and Development in Information Retrieval, ACM Press, 1995, 273–281.

[15] S. Jaillet, A. Laurent, M. Teisseire and J. Chauché, *Order and Mess in text categorization: Why using sequential patterns to classify*, In Proc. of 3rd Workshop on Mining Temporal and Sequential Data, in conjunction with The 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (TDM'04/KDD'04), WA, USA, August 2004.

[16] D. Janssens, G. Wets, T. Brijs, K. Vanhoof and G. Chen, *Adapting the CBA-algorithm by means of intensity of implication*, In Proc. of the 1st Int. Conf. on Fuzzy Information Processing Theories and Applications, China, March 2003, 397–403.

[17] T. Joachims, *Text categorization with support vector machines: learning with many relevant features*, In Proc. of ECML-98, 10th European Conf. on Machine Learning, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE., 137–142.

[18] B. Lent, R. Agrawal and R. Srikant, it Discovering Trends in Text Databases, In Proc. of the 3rd Int. Conf. on Knowledge Discovery and Data Mining, (KDD'97), CA, USA, August 1997. AAAI Press, 227–230.

[19] W. Li, J. Han and J. Pei, *CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules*, In Proc. of the 2001 IEEE Int. Conf. on Data Mining (ICDM'01), CA, USA, November 2001, 369–376.
[20] Y. Li and A. Jain, Classification of text documents, *The Computer Journal* **41**(8) (1998), 537–546.
[21] B. Liu, W. Hsu and Y. Ma, *Integrating Classification and Association Rule Mining*, In Proc . of the 4th Int. Conf. on Knowledge Discovery and Data Mining (KDD'98), NY, USA, August 1998. AAAI Press, 80–86.
[22] B. Liu, Y. Ma and C.-K. Wong, *Improving an Association Rule Based Classifier*, In Proc. of the 4th European Conf. on Principles of Data Mining and Knowledge Discovery (PKDD'00), France, September 2000. Springer, 504–509.
[23] B. Liu, Y. Ma and C.-K. Wong, Classification Using Association Rules: Weaknesses and Enhancements, in: *Data Mining for Scientific Application and Engineering Applications*, V. Kumar, ed., Kluwer Academic, 2001.
[24] M. Maron, Automatic indexing: An experimental inquiry, *Journal of the ACM* (*JACM*) **8** (1961), 404–417.
[25] F. Masseglia, P. Poncelet and M. Teisseire, Incremental mining of sequential patterns in large databases, *Data and Knowledge Engineering* **46**(1) (2003).
[26] F. Masseglia, P. Poncelet and M. Teisseire, *Pre-processing Time Constraints for Efficiently Mining Generalized Sequential Patterns*, In Proc. of the 11th Int. Symposium on Temporal Representation and Reasoning, (TIME'04), IEEE Computer Society Press, July 2004, 87–95.
[27] D. Meretakis and B. Wuthrich, *Extending Naive Bayes Classifiers Using Long Itemsets*, In Proc. of the 5th Int. conf. on Knowledge Discovery and Data Mining (KDD'99), CA, USA, August 1999, 165–174.
[28] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal and M.-C. Hsu, *PrefixSpan mining sequential patterns efficiently by prefix projected pattern growth*, In International Conference on Data Engineering (ICDE'01), Heidelberg, Germany, 2001, 215–226.
[29] J. Quinlan, *C4.5 – Programs for Machine Learning*. Morgan Kaufman, CA, USA, 1993.
[30] C.J. Van Rijsbergen, *Information Retrieval*, Butterworths, sec. edition, 1979.
[31] G. Salton and M.J. McGill, *Introduction to modern information retrieval*, McGraw-Hill, New York, 1983.
[32] G. Salton, C. Yang and C. Yu, A theory of term importance in automatic text analysis, *Journal of the American Society for Information Science* **36** (1975), 33–44.
[33] F. Sebastiani, *Machine learning in automated text categorisation*, (Vol. 34), In Proc. of ACM Computing Surveys, 2002, 1–47.
[34] M. Shimbo, T. Yamasaki and Y. Matsumoto, *Automatic classification of sentences in the medline abstracts: A case study of the power of word sequence features*, In Proc. of the 6th Sanken (ISIR) International Symposium, Osaka, Japan, 2003, 135–138.
[35] R. Srikant and R. Agrawal, *Mining Sequential Patterns: Generalizations and Performance Improvements*, In Proc. of the 5th Int.Conf. on Extending Database Technology (EDBT'96), September 1996, 3–17.
[36] M. Takechi, T. Tokunaga, Y. Matsumoto and H. Tanaka, *Feature selection in categorizing procedural expressions*, In Proc. Sixth International Workshop on Information Retrieval with Asian Languages (IRAL'2003), Sapporo, Japan, July 7 2003.
[37] K. Wang, S. Zhou and Y. He, *Growing Decision Trees on Support-less Association Rules*, In Proc. of the 6th Int. Conf. on Knowledge discovery and data mining (KDD'00), MA, USA, August 2000. ACM Press, 265–269.
[38] P.-C. Wong, W. Cowley, H. Foote, E. Jurrus and J. Thomas, *Visualizing Sequential Patterns for Text Mining*, In Proc of the 2000 IEEE Symposium on Information Visualization, (INFOVIS'00), UT, USA, October 2000. IEEE Computer Society Press, 105–114.
[39] Y. Yang, An Evaluation of statistical approaches to text categorization, *Information Retrieval Journal* **1**(1/2) (1999), 69–90.