Mining User-Aware Rare Sequential Topic Patterns in Document Streams

Jiaqi Zhu, Member, IEEE, Kaijun Wang, Yunkun Wu, Zhongyi Hu, and Hongan Wang, Member, IEEE

Abstract—Textual documents created and distributed on the Internet are ever changing in various forms. Most of existing works are devoted to topic modeling and the evolution of individual topics, while sequential relations of topics in successive documents published by a specific user are ignored. In this paper, in order to characterize and detect personalized and abnormal behaviors of Internet users, we propose Sequential Topic Patterns (STPs) and formulate the problem of mining User-aware Rare Sequential Topic Patterns (URSTPs) in document streams on the Internet. They are rare on the whole but relatively frequent for specific users, so can be applied in many real-life scenarios, such as real-time monitoring on abnormal user behaviors. We present a group of algorithms to solve this innovative mining problem through three phases: preprocessing to extract probabilistic topics and identify sessions for different users, generating all the STP candidates with (expected) support values for each user by pattern-growth, and selecting URSTPs by making user-aware rarity analysis on derived STPs. Experiments on both real (Twitter) and synthetic datasets show that our approach can indeed discover special users and interpretable URSTPs effectively and efficiently, which significantly reflect users’ characteristics.

Index Terms—Web mining, sequential patterns, document streams, rare events, pattern-growth, dynamic programming.

1 INTRODUCTION

D ocument streams are created and distributed in various forms on the Internet, such as news streams, emails, micro-blog articles, chatting messages, research papers, archives, web forum discussions, and so forth. The contents of these documents generally concentrate on some specific topics, which reflect offline social events and users’ characteristics in real life. To mine these pieces of information, a lot of researches of text mining focused on extracting topics from document collections and document streams through various probabilistic topic models, such as classical PLSI [15], LDA [7] and their extensions [5], [6], [16], [18], [19], [24], [33], [34], [38], [39].

Taking advantage of these extracted topics in document streams, most of existing works analyzed the evolution of individual topics to detect and predict social events as well as user behaviors [8], [11], [12], [23]. However, few researches paid attention to the correlations among different topics appearing in successive documents published by a specific user, so some hidden but significant information to reveal personalized behaviors has been neglected.

In order to characterize user behaviors in published document streams, we study on the correlations among topics extracted from these documents, especially the sequential relations, and specify them as Sequential Topic Patterns (STPs). Each of them records the complete and repeated behavior of a user when she is publishing a series of documents, and are suitable for inferring users’ intrinsic characteristics and psychological statuses. Firstly, compared to individual topics, STPs capture both combinations and orders of topics, so can serve well as discriminative units of semantic association among documents in ambiguous situations. Secondly, compared to document-based patterns, topic-based patterns contain abstract information of document contents and are thus beneficial in clustering similar documents, and finding some regularities about Internet users. Thirdly, the probabilistic description of topics helps to maintain and accumulate the uncertainty degree of individual topics, and can thereby reach high confidence level in pattern matching for uncertain data.

For a document stream, some STPs may occur frequently and thus reflect common behaviors of involved users. Beyond that, there may still exist some other patterns which are globally rare for the general population, but occur relatively often for some specific user or some specific group of users. We call them User-aware Rare STPs (URSTPs). Compared to frequent ones, discovering them is especially interesting and significant. Theoretically, it defines a new kind of patterns for rare event mining, which is able to characterize personalized and abnormal behaviors for special users. Practically, it can be applied in many real-life scenarios of user behavior analysis, as illustrated in the following example.

Scenario 1 (Real-time monitoring on abnormal user behaviors). Recently, micro-blogs such as Twitter are attracting more and more attentions all over the world. Micro-blog messages are real-time, spontaneous reports of what the users are feeling, thinking and doing, so reflect users’ characteristics and statuses. However, the real intentions of users for publishing these messages are hard to reveal directly from individual messages, but both content information and temporal relations of messages
are required for analysis, especially for abnormal behaviors without prior knowledge. What’s more, if illegal behaviors are involved, detecting and monitoring them is particularly significant for social security surveillance. For example, the lottery fraud behaviors via Internet usually accord with the following four steps, which are embodied in the topics of published messages: (1) make award temptations; (2) diddle other users’ information; (3) obtain various fees by cheating; (4) take illegal intimidation if their requests are denied. STPs happen to be able to combine a series of inter-correlated messages, and can thus capture such behaviors and associated users. Furthermore, even if some illegal behaviors are emerging, and their sequential rules have not been explicit yet, we can still expose them by URSTPs, as long as they satisfy the properties of both global rareness and local frequentness. That can be regarded as important clues for suspicion and will trigger targeted investigations. Therefore, mining URSTPs is a good means for real-time user behavior monitoring on the Internet.

It is worth noting that the ideas above are also applicable for another type of document streams, called browsed document streams, where Internet users behave as readers of documents instead of authors. In this case, STPs can characterize complete browsing behaviors of readers, so compared to statistical methods, mining URSTPs can better discover special interests and browsing habits of Internet users, and is thus capable to give effective and context-aware recommendation for them. While, this paper will concentrate on published document streams and leave the applications for recommendation to future work.

To solve this innovative and significant problem of mining URSTPs in document streams, many new technical challenges are raised and will be tackled in this paper. Firstly, the input of the task is a textual stream, so existing techniques of sequential pattern mining for probabilistic databases cannot be directly applied to solve this problem. A preprocessing phase is necessary and crucial to get abstract and probabilistic descriptions of documents by topic extraction, and then to recognize complete and repeated activities of Internet users by session identification. Secondly, in view of the real-time requirements in many applications, both the accuracy and the efficiency of mining algorithms are important and should be taken into account, especially for the probability computation process. Thirdly, different from frequent patterns, the user-aware rare pattern concerned here is a new concept and a formal criterion must be well defined, so that it can effectively characterize most of personalized and abnormal behaviors of Internet users, and can adapt to different application scenarios. And correspondingly, unsupervised mining algorithms for this kind of rare patterns need to be designed in a manner different from existing frequent pattern mining algorithms.

To sum up, this paper makes the following contributions:

- To the best of our knowledge, this is the first work that gives formal definitions of STPs as well as their rarity measures, and puts forward the problem of mining URSTPs in document streams, in order to characterize and detect personalized and abnormal behaviors of Internet users;
- We propose a framework to pragmatically solve this problem, and design corresponding algorithms to support it. At first, we give preprocessing procedures with heuristic methods for topic extraction and session identification. Then, borrowing the ideas of pattern-growth in uncertain environment, two alternative algorithms are designed to discover all the STP candidates with support values for each user. That provides a trade-off between accuracy and efficiency. At last, we present a user-aware rarity analysis algorithm according to the formally defined criterion to pick out URSTPs and associated users.
- We validate our approach by conducting experiments on both real and synthetic datasets.

The rest of the paper is organized as follows. Section 2 reviews related works including topic mining and sequential pattern mining for deterministic and uncertain databases. In Section 3, we give the key definitions related to STPs, and formulate the problem of mining URSTPs in document streams. The processing framework, preprocessing algorithms and mining algorithms are presented in detail in Section 4. Section 5 shows the experimental results on real Twitter datasets, and leaves the synthetic results to Appendix. Section 6 concludes the paper and discusses future directions.

2 RELATED WORK

Topic mining in document collections has been extensively studied in the literature. Topic Detection and Tracking (TDT) task [3], [9], [35] aimed to detect and track topics (events) in news streams with clustering-based techniques on keywords. Considering the co-occurrence of words and their semantic associations, a lot of probabilistic generative models for extracting topics from documents were also proposed, such as PLSI [15], LDA [7] and their extensions integrating different features of documents [5], [19], [24], as well as models for short texts [16], [34], like Twitter-LDA [39].

In many real applications, document collections generally carry temporal information and can thus be considered as document streams. Various dynamic topic modeling methods have been proposed to discover topics over time in document streams [6], [18], [33], [38], and then to predict offline social events [8], [11], [23]. However, these methods were designed to construct the evolution model of individual topics from a document stream, rather than to analyze the correlations among multiple topics extracted from successive documents for specific users.

Sequential pattern mining is an important problem in data mining, and has also been well studied so far. In the context of deterministic data, a comprehensive survey can be found in [21], [25]. The concept support [25] is the most popular measure for evaluating the frequency of a sequential pattern, and is defined as the number or proportion of data sequences containing the pattern in the target database. Many mining algorithms have been proposed based on support, such as PrefixSpan [29], FreeSpan [13] and SPADE [36]. They discovered frequent sequential patterns whose support values are not less than a user-defined threshold, and were extended by SLPMiner [30] to deal with length-decreasing support constraints. Nevertheless, the obtained
patterns are not always interesting for our purpose, because those rare but significant patterns representing personalized and abnormal behaviors are pruned due to low supports. Furthermore, the algorithms on deterministic databases is not applicable for document streams, as they failed to handle the uncertainty in topics.

For uncertain data, most of existing works studied frequent itemset mining in probabilistic databases [1], [10], but comparatively fewer researched addressed the problem of sequential pattern mining. Muzzammal et al. focused on sequence-level uncertainty in sequential databases, and proposed methods to evaluate the frequency of a sequential pattern based on expected support, in the frame of candidate generate-and-test [28] or pattern-growth [26]. Since expected support would lose the probability distribution of the support, a finer measure frequentess probability was defined for general itemsets [4], [32], [37], and used in mining frequent sequential patterns for sequence-level and element-level uncertain databases [20], [27], [40]. However, these works did not consider where the uncertain databases come from and how the probabilities in the original data are computed, so cannot be directly employed for our problem which takes document streams as input. Moreover, they also focused on frequent patterns and thus cannot be utilized to discover rare but interesting patterns associated with special users.

In the aspect of sequential patterns for topics, Hariri et al. [14] presented an approach for context-aware music recommendation based on sequential relations of latent topics. The topic set of each song is at first determined by the topic set of each document according to Definition 3. The topic set of each song is at first determined by the topic number $K$, latent topics of these documents can be learnt through probabilistic topic models like LDA [7] and Twitter-LDA [39], and comprise a set $T$. Each topic is defined as follows.

Definition 2 (Topic). A semantically coherent topic $z$ in the text collection $D$ is represented by a probabilistic distribution of words in the given vocabulary $V$. It is denoted as $\{p(w|z)\}_{w \in V}$, which satisfies $\sum_{w \in V} p(w|z) = 1$.

In this way, each document can be represented by a probabilistic mixture (proportion) of these $K$ independent topics, which form a structured topic-level document.

Definition 3 (Topic-Level Document). Given an original document $d \in D$ and a topic set $T$, the corresponding topic-level document $td^d$ is defined as a set of topic-probability pairs, in the form of $td^d = \{(z, p(z|d)) : z \in T \}$. Here, the probabilities are obtained through some topic model and satisfy $\sum_{z \in T} p(z|d) = 1$. The superscript $d$ can be omitted when the original document is not cared.

Actually, we can select some representative topics from $T$ to approximately describe the document, which will be discussed in the preprocessing procedure in the next section.

3 Sequential Topic Patterns

On the Internet, the documents are created and distributed in a sequential way and thus compose various forms of published document streams for specific websites. In this paper, we abbreviate them as document streams.

Definition 4 (Document Stream). A document stream is defined as a sequence $DS = \langle (d_1, u_1, t_1), (d_2, u_2, t_2), \ldots, (d_N, u_N, t_N) \rangle$, where $d_i (i = 1, \ldots, N)$ is a document published by user $u_i$ at time $t_i$ on a specific website, and $t_i \leq t_j$ for all $i \leq j$.

Usually, one user cannot write two documents simultanously, so we can assume that at any time point, for any specific user, at most one document is published. Formally, if $t_i = t_j$, then $u_i \neq u_j$ always hold.

Obviously, each document stream can be transformed into a topic-level document stream of the form $TDS = \langle (td_1, u_1, t_1), (td_2, u_2, t_2), \ldots, (td_N, u_N, t_N) \rangle$, by extracting topics for each document according to Definition 3.

In this paper, we pay attention to the correlations among successive documents published by the same user in a document stream. A kind of fundamental but important correlations is the sequential relation among topics of these
documents, which can be defined by sequential topic patterns, and abbreviated as STPs. They are suitable to characterize users’ complete and personalized behaviors when publishing documents in a website.

**Definition 5 (Sequential Topic Pattern).** A Sequential Topic Pattern (STP) \( \alpha \) is defined as a topic sequence \( \langle z_1, z_2, \ldots, z_n \rangle \), where each \( z_i \in T \) is a learnt topic. \( n = |\alpha| \) denotes the number of topics contained in \( \alpha \), and is called the length of \( \alpha \). A pattern with length \( n \) is called an \( n \)-STP.

Since STPs reflect users’ characteristics which probably show repeated behaviors, their instances should be discovered not in the whole document stream involving different users and a long time period, but in some subsequences related to a specific user during a certain time period. Each of such subsequences, called a session of the document stream, consists of a series of possibly correlated messages posted by a user during a time period on some micro-blog sites or Internet forums. Hence, in order to find significant STPs, a document stream should be divided into independent sessions in advance with the definition below.

**Definition 6 (Session).** Given a topic-level document stream \( TDS \), a session \( s \) is defined as a subsequence of \( TDS \) associated with the same user, i.e., \( s = \langle (td_1, u, t_1), (td_2, u, t_2), \ldots, (td_i, u, t_i) \rangle \subseteq TDS \).

For a specific user, there may exist multiple sessions in a document stream, which should be disjoint and consecutive. A sketch map of session identification is shown in Fig. 1. Each ellipse represents a session, and all the sessions in each line constitute a document subsequence for a specific user. Many heuristic techniques can be applied for session identification in different scenarios. We leave our methods to the next section, as another preprocessing procedure.

When the session set of a topic-level document stream is obtained, we can find some concrete instances of an STP for each session. This process can be regarded as sequence matching between the ordered topics specified in the STP and the probabilistic topics occurring in the ordered documents belonging to a specific session, and can be formally defined in the following way.

**Definition 7 (Pattern Instance).** Given an STP \( \alpha = \langle z_1, z_2, \ldots, z_n \rangle \) and a session \( s \) associated with a user \( u \), if we can extract a subsequence \( s' = \langle (td_1, u, t_1), \ldots, (td_i, u, t_i) \rangle \subseteq s \), and for each \( i = 1, \ldots, n \), \((z_i, p_i) \in td_i \), holds for some \( p_i \), then we say that \( \bar{\alpha} = \langle \langle z_1, p_1 \rangle, \ldots, \langle z_n, p_n \rangle \rangle \) is a pattern instance of \( \alpha \) in the session \( s \). The occurrence probability of the instance can be simply calculated as a product \( P(\bar{\alpha}) = \prod_{k=1}^{n} p_k \), with the independence assumption similar to [27].

One should note that the subsequence \( s' \) is not necessarily contiguous in \( s \), but just the order needs to be retained. In addition, when analyzing the pattern instances in a session, the user component and the time component of documents are irrelevant, so we can omit them in the representation of a session. Two examples of sessions in this form are given in Table 1. For the STP \( \langle z_1, z_3 \rangle \), it has one instance in \( s_1 \) with probability \( 0.5 \times 0.4 = 0.2 \), and two instances in \( s_2 \) with probabilities 0.3 and 0.12, respectively.

<table>
<thead>
<tr>
<th>Session ID</th>
<th>Topic-level document sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s_1 )</td>
<td>( \langle \langle z_1, 0.5 \rangle, \langle z_2, 0.8 \rangle \rangle \langle \langle z_2, 0.3 \rangle, \langle z_3, 0.4 \rangle \rangle \rangle )</td>
</tr>
<tr>
<td>( s_2 )</td>
<td>( \langle \langle z_1, 0.6 \rangle \rangle, \langle \langle z_2, 0.5 \rangle \rangle \langle \langle z_3, 0.2 \rangle \rangle \rangle )</td>
</tr>
</tbody>
</table>

We can see that in a specific session, there may be multiple pattern instances for an STP, each with a respective occurrence probability. These instances may be not independent with each other, as some common topics extracted from common documents are involved. Therefore, the probability that an STP \( \alpha \) occurs in \( s \), denoted as \( Pr(\alpha \subseteq s) \), cannot be obtained by simply adding up the probability of each pattern instance. For example, the probability \( Pr(\langle z_1, z_3 \rangle \subseteq s_2) \) is not \( 0.3 + 0.12 = 0.42 \), but should be calculated as \( 0.6 \times (0.5 + (1 - 0.5) \times 0.2) = 0.36 \). The idea of dynamic programming can be utilized here, which will be presented in the next section.

### 3.3 User-Aware Rare Sequential Topic Patterns

Most of existing works on sequential pattern mining focused on frequent patterns, but for STPs, many infrequent ones are also interesting and should be discovered. Specifically, when Internet users’ publish documents, the personalized behaviors characterized by STPs are generally not globally frequent but even rare, since they expose special and abnormal motivations of individual authors, as well as particular events having occurred to them in real life. Therefore, the STPs we would like to mine for user behavior analysis on the Internet should be distinguishing features of involved users, and thus satisfy the following two conditions:

1. They should be **globally rare** for all sessions involving all users of a document stream;
2. They should be **locally and relatively frequent** for the sessions associated with a specific user.

Next, we will formally specify this kind of STPs step by step, starting with the classical concept of **support** to describe the frequency. For deterministic sequential pattern mining, the support of a pattern \( \alpha \) is defined as the number or proportion of the sequences containing \( \alpha \) in the target database [25], but inapplicable for uncertain sequences like topic-level document streams. Instead, the expected support is appropriate to measure the frequency on uncertain sequences [10], and can be computed by summing up the occurrence probabilities of \( \alpha \) in all sequences [28]. In other words, it expresses the expected number of sequences containing \( \alpha \). To measure the frequency of STPs, we modify it a little to

![Fig. 1. Sketch map of session identification.](image-url)
record the proportion of sessions where $\alpha$ occurs also in terms of expectation, via dividing the summation by the number of sessions. That is necessary because the session number here is no longer a constant when we consider both the global frequency and the local frequency of $\alpha$ for different users. For simplicity, this measure is still denoted as support instead of expected support in this paper.

**Definition 8 (Support of STP).** Given a session set $S = \{s_1, s_2, \cdots, s_n\}$ as the database and an $n$-STP $\alpha = (z_1, z_2, \cdots, z_n)$, the support of $\alpha$ in $S$ is defined as

$$supp(\alpha, S) \equiv \frac{\sum_{i=1}^{n} Pr(\alpha \subseteq s_i)}{|S|} \tag{1}$$

Taking the session set in Table 1 for example, we have $supp(\langle z_1, z_3 \rangle, \{s_1, s_2\}) = (0.2 + 0.36)/2 = 0.28$. The support defined above can be explained in the following two ways:

- When the session set $S$ contains all the sessions for all users in a document stream $TDS$, the support reflects the global frequency of $\alpha$, and can be denoted as $supp(\alpha, TDS)_\alpha$.
- When the session set $S$ contains all the sessions for a specific user $u$ in $TDS$, the support reflects the local frequency of $\alpha$ for $u$, and can be denoted as $supp(\alpha, TDS)_u$.

If the document stream is fixed with no ambiguity, the second parameter $TDS$ can be omitted from the notations.

Since the lengths of significant STPs are unknown, we have to consider all the possible STPs with different lengths and compute their support values. Notice that for an $n$-STP, the probability of each instance is the product of $n$ probability values (decimals), and its support can be approximately thought as a combination (summation) of these products, so longer STPs tend to have lower support values. In order to fairly compare among the supports of STPs with different lengths, we should normalize their values to the same order of magnitude, i.e., a single probability value, approximately thought as a combination (summation) of these values. That will reduce the impact of pattern lengths in the later comparison and unified computation. To this end, we further define scaled support as follows.

**Definition 9 (Scaled Support of STP).** Given the support $supp(\alpha, S)$ for an $n$-STP $\alpha$, the scaled support $scsupp(\alpha, S)$ is defined as the $n$-th root of the support value. Formally,

$$scsupp(\alpha, S) \equiv supp(\alpha, S)^{1/n} \tag{2}$$

This formula is applicable not only for the general definition on a session set, but also for the global and local variants on a (possibly defaulted) document stream, such as $scsupp(\alpha)_\alpha$ and $scsupp(\alpha)_u$.

Based on the scaled support, we can evaluate these STPs in terms of their abilities in characterizing personalized and abnormal behaviors of Internet users, and pick out some significant and representative ones. Firstly, each selected STP should be linked to a specific user, so can be called a user-aware STP. Secondly, it should reflect the particularity on frequency, not only at user level but also at pattern level. According to these ideas, we define two new measures, absolute rarity and relative rarity. They regularize the scaled support of an STP for a user over other users and other STPs respectively, and will be computed successively.

**Definition 10 (Absolute Rarity of STP).** The absolute rarity of an STP $\alpha$ for a user $u$, denoted by $AR(\alpha)|_u$, is defined as the difference between the scaled support of $\alpha$ in the sessions for $u$ (local support) and that in all sessions (global support). Formally,

$$AR(\alpha)|_u \equiv scsupp(\alpha)|_u - scsupp(\alpha), \quad (3)$$

**Definition 11 (Relative Rarity of STP).** The relative rarity of an STP $\alpha$ for a user $u$, denoted by $RR(\alpha)|_u$, is defined as the difference between the absolute rarity of $\alpha$ for $u$ and the average absolute rarity among all the discovered STPs for $u$. Formally,

$$RR(\alpha)|_u \equiv AR(\alpha)|_u - \frac{\sum_{\beta \in \Phi_u} AR(\beta)|_u}{|\Phi_u|}, \quad (4)$$

where $\Phi_u$ is the set containing all the discovered STPs for the user $u$, i.e., for all $\beta \in \Phi_u$, $supp(\beta)|_u > 0$ holds. Compared to [17] where the average value is taken among all STPs for any user, this measure is fully user-specific and thus more accurate to reflect the particularity of the STP with respect to the user.

Now, we can define the core concept of this paper, user-aware rare STPs, abbreviated as URSTPs.

**Definition 12 (User-Aware Rare STP).** Given a topic-level document stream $TDS$, a scaled support threshold $h_{ss}$, and a relative rarity threshold $h_{rr}$, an STP $\alpha$ is called a User-Aware Rare STP (URSTP) if and only if both $scsupp(\alpha) \leq h_{ss}$ and $RR(\alpha)|_u \geq h_{rr}$ hold for some user $u$.

Here, the first condition indicates the global rareness of $\alpha$, and the second one assures its relatively high frequency for the specific user $u$. That is consistent with the requirement of our mining task proposed at the beginning of this subsection. Based on these definitions, the problem of mining URSTPs in this paper can be formulated as follows: Given a (published) document stream on the Internet, discover all URSTPs and associated users, which characterize users’ personalized and abnormal behaviors.

## 4 Mining URSTP

In this section, we propose a novel approach to mining URSTPs in document streams. The main processing framework for the task is shown in Fig. 2. It consists of three phases. At first, textual documents are crawled from some micro-blog sites or forums, and constitute a document stream as the input of our approach. Then, as preprocessing procedures, the original stream is transformed to a topic-level document stream and then divided into many sessions to identify complete user behaviors. Finally and most importantly, we discover all the STP candidates in the document stream for all users, and further pick out significant URSTPs associated to specific users by user-aware rarity analysis.

In order to fulfill this task, we design a group of algorithms. To unify the notations, many variables are denoted and stored in the key-value form. For example, $User_{Sess}$
Algorithm 1. Main$(DS, h_{ss}, h_{rr})$

1: User$_{Sess} ←$ preprocess$(DS)$;
2: User$_{STP} ← \emptyset$;
3: for all $(u : S_u)$ ∈ User$_{Sess}$ do
4:     Start a new thread;
5:     STP$_{Supp} ←$ UpsSTP$(\emptyset, S_u, \emptyset, \emptyset)$;
6:     User$_{STP} ←$ User$_{STP}$ ∪ \{(u : STP$_{Supp}$)\};
7:     User$_{URSTP} ←$ URSTPMiner$(User$_{STP}$, User$_{Sess}$, h_{ss}, h_{rr})$;
8: return User$_{URSTP}$;

After preprocessing, we obtain a set of user-session pairs. For each of them with a specific user $u$, a new thread is started and a pattern-growth based subroutine UpsSTP is recursively invoked to find all the STP candidates for $u$, paired with their support values, and add

Since each session should contain a complete publishing behavior of an individual user, we need to at first divide the document stream according to different users, which is an easy job as the author of each document is explicitly given in the input stream. The result for each user $u$ is a subsequence of the topic-level document stream restricted to that user, i.e., $TDS_u = \{(d_1, u, t_1), (d_2, u, t_2), \ldots, (d_N, u, t_N)\}$. After that, we also need to partition the subsequence to identify complete and repeated activities as consecutive and non-overlapped sessions. They constitute a session set $S_u = \{s_1, s_2, \ldots, s_m\}$ satisfying $TDS_u = s_1 \circ s_2 \circ \cdots \circ s_m$, where $\circ$ is the concatenation operator.

Making a correct partition to reconstruct these meaningful sessions is very hard, because there is no enough
information to determine the starting point and the ending point of each real complete behavior. Moreover, the partition should depend on the characteristics of the input stream, so document streams from different websites should be suited to different partition techniques. Two classical time-oriented heuristic methods [31] can be applied here, each of which is based on a reasonable assumption. In the following, we give the corresponding algorithms designed for our mining task.

1) Time Interval Heuristics. It assumes that the time interval of any two adjacent documents in the same session is less than or equal to a predefined threshold $h_{ts}$. The algorithm named $\text{tiPartition}$ is shown in Algorithm 2. It examines each document on the input stream orderly to see whether it should be the starting point of a new session, by checking the condition that the time difference between it and its previous document exceeds the threshold (line 3).

2) Time Span Heuristics. It assumes that the duration of each session is less than or equal to a predefined threshold $h_{ts}$. The algorithm is named $\text{tsPartition}$, and the only distinction from Algorithm 2 is the condition for a new session in line 3. Specifically, the time point of the previous document $t_{s-1}$ is replaced by the starting time point of the current session $t_k$, and $h_{ts}$ is used as threshold. Due to the page limit, the pseudocode is omitted here.

**Algorithm 2. $\text{tiPartition}(TDS_u, h_{ts})$**

1: $j, k \leftarrow 1$
2: for $i = 1$ to $N$ do
3: if $t_i - t_{i-1} > h_{ts}$ then
4: $s_j \leftarrow \langle (td_k, u, t_k), \cdots, (td_{s-1}, u, t_{s-1}) \rangle$
5: $j \leftarrow j + 1$
6: $kA[k, j] = p_{kj} \times [A[1(5)] + 1 (pkj) \times A[k, j-1]]$
7: $s_j \leftarrow \langle (td_k, u, t_k), \cdots, (td_N, u, t_N) \rangle$
8: $m \leftarrow j$
9: return $S_u = \{s_1, s_2, \cdots, s_m\}$

Intuitively, the first heuristic with unfixed durations is more suitable for our problem than the second one. In addition, these two heuristics do not require additional information from the texts of documents, and presume that the behaviors of all users manifested in a document stream conform to a unified time-oriented rule, so neglect different publishing characteristics of users. Beyond that, some websites allow users to build hyperlinks among published documents, so in this case, it is possible to find more accurate and user-specific partitions if users really produce these links to indicate complete behaviors, but this policy is limited by the function of the website and the publishing habits of users. Therefore, for a specific document stream, we need to choose an appropriate partition algorithm to make a feasible and comparatively correct session identification.

### 4.2 STP Candidate Discovery by Pattern-Growth

As shown in Fig. 3, the subprocedure of STP candidate discovery is executed in parallel for each user. It aims to find all STPs occurring in the document stream associated with a specific user, paired with the (expected) support values of these STPs. According to Equation 1, the key step is to compute $\Pr(\alpha \sqsubseteq s_i)$, which is the probability that an STP $\alpha$ occurs in a session $s_i$ belonging to the user. In this subsection, we at first present a DP-based algorithm to derive all STPs for the user and exactly compute the support values of them. Then, in order to improve the efficiency of our approach, we also give an approximation algorithm to estimate the support values for all STPs. Both algorithms are designed in the manner of pattern-growth.

#### 4.2.1 DP-Based Algorithm

Similar to [26], the occurrence probability of an STP in a session (a sequence of topic-level documents) can be computed by dynamic programming. Assuming that the STP $\alpha = (z_1, z_2, \cdots, z_n)$, and the session $s = \langle td_1, td_2, \cdots, td_q \rangle$, we define an $((n + 1) \times (q + 1))$ DP-matrix $A$ to calculate the probability $Pr(\alpha \sqsubseteq s)$. For $1 \leq k \leq n$ and $1 \leq j \leq q$, $A[k, j]$ denotes the probability of a subpattern (prefix) of $\alpha$ occurring in a subsequence (prefix) of $s$, i.e., $Pr(z_1, \cdots, z_k) \subseteq (td_1, \cdots, td_j)$. Besides, the values in line 0 and row 0 can be assigned with initial values. Specifically, we let $A[0, j] = 1$ for all $0 \leq j \leq q$ and $A[k, 0] = 0$ for all $1 \leq k \leq n$. With the help of these initial values, other meaningful values in the matrix can be computed row by row according to the following formula:

$$k_j$$

*where $p_{kj}$ denotes the probability that the document $td_j$ belongs to the topic $z_k$.* Specifically, if $(z_k, p') \in td_j$ for some $p'$, then $p_{kj} = p'$, and $p_{kj} = 0$ otherwise.

We can see that through traversing and computing all the entries in the DP-matrix one by one, $Pr(\alpha \sqsubseteq s)$ can be obtained eventually as it is just the value of $A[n, q]$. However, this method will bring high complexity with respect to both time and space, i.e., $O(n \cdot q)$. In fact, some entries in the matrix are not necessary to be maintained and computed. If $p_{kj} = 0$, then $A[k, j] = A[k, j-1]$ holds, so there will be no actual computation, but just a copy of previous entries. Hence, to get the value of $A[k, j]$, we could resort to a previous entry with the same value in the current line (the $k$-th line), instead of maintaining it as a new entry. In this way, we can just pay attention to the positions of the sequence where the topic $z_k$ indeed occurs in the topic-level document, and we call the corresponding entries valid.
Formally, when $p_{kj} > 0$, the entry $A[k,j]$ will be set valid, and its value will be computed by

$$A[k,j] = p_{kj} \times A[k-1,j'] + (1-p_{kj}) \times A[k,j']$$  \hspace{1cm} (6)

where $j' = \max \{ x \mid x < j \land A[k-1,x] \text{ is valid } \}$ and $j'' = \max \{ x \mid x < j \land A[k,x] \text{ is valid } \}$ are the nearest valid entries (including initial entries) in the current line and the last line respectively. Although the worst-case complexity cannot be changed, the time cost will be practically reduced.

Based on the ideas above, we design the subprocedure \texttt{UpsSTP} shown in Algorithm 3 to recursively discover STP candidates for a specific user by pattern-growth and compute their support values. It extends the classical algorithm PrefixSpan [29] with uncertainty calculation. We define two key structures to record the information of STPs which have already been generated and will be used for later extension. For an STP $\alpha$ and a session set $S = \{s_1, s_2, \cdots, s_{|S|}\}$,

- $\text{Pref}_\alpha$ is a set of $\alpha$-prefix triples of the form $(i,j,p)$, which records the ending position $j$ of each instance of $\alpha$ in $s_i$, as well as the occurrence probability $p$ of $\alpha$ in the prefixes of $s_i$ ending with the position. Corresponding to the DP-matrix, it means that the entry $A[i\alpha,|j]$ for the session $s_i$ is valid with value $p$.
- $S_\alpha$ is a set of $\alpha$-projected subsequences (suffixes) of $S$. For each sequence $s_i \in S$, if it contains some instances of $\alpha$, a corresponding subsequence will be retained in $S_\alpha$ by projecting out the prefix ending with the last position (document) of the first instance of $\alpha$; otherwise, $s_i$ is removed from $S_\alpha$. Here, we adopt the pseudo-projection method proposed in PrefixSpan [29] to reduce the number and the size of sequences. Specifically, $S_\alpha$ is represented by a set of session-index pairs of the form $(i,j)$, which means that $s_i$ is retained in $S_\alpha$, and the projected suffix starts from the $j$-th position of $s_i$. In this way, the suffixes in $S_\alpha$ are not required to be generated and stored additionally, and the indexes of sequences in $S$ can be used in a unified manner for all STPs.

The subprocedure has four input parameters: $\alpha$ is the currently derived STP and will be extended in this subprocedure; $S$ is the session set for the specific user $u$, and is fixed during recursions; $\text{Pref}_\alpha$ is a set of $\alpha$-prefix triples; and $S_\alpha$ is a set of $\alpha$-projected suffixes in the form of session-index pairs. For the first invocation by the main procedure, $\alpha$ is $\varnothing$, so $\text{Pref}_\alpha$ is also $\varnothing$, and $S_\alpha$ is just the complete session set $S$ with original starting positions, represented by $\{(i, \lfloor i \rfloor \mid 1 \leq i \leq |S|\}$.

Each execution of $\text{UpsSTP}$ performs one step of pattern-growth from the input STP $\alpha$ to an extended one $\beta = \alpha \circ z$, by appending a new element (topic) $z$. At first, we scan all the sequences in $S_\alpha$ to obtain the set $E$ containing all the possible topics which can be appended to $\alpha$ (line 2). Then, for each topic $z$ in $E$ and each suffix pair $(i,j_0)$ in $S_\alpha$, we find all the instances of $z$ in the projected suffix of $s_i$ (line 8), compute the probability of $\beta = \alpha \circ z$ occurring in the corresponding suffixes of $s_i$, and record new prefix triples in $\text{Pref}_\beta$ (lines 9-19). Specifically, for each instance of $z$, if $\text{Pref}_\alpha = \varnothing$, then $\alpha$ is an empty sequence, and $\beta$ contains just one topic $z$, which corresponds to the line 1 of the DP-matrix, so the probability value can be computed by Equation 6 with $A[k-1,j'] = 1$ (line 14); otherwise, it can be computed by the same formula utilizing the input prefix triple information in $\text{Pref}_\alpha$ to get $A[k-1,j']$ (line 18). When all the instances of $z$ are handled in order, the probability for the last one is just $Pr(\beta \subseteq s_i)$, so the support of $\beta$ in $S$ can be updated by accumulating this value divided by the number of sessions in $S$ (line 20), according to Equation 1. In addition, we get the $\beta$-projected suffix of $s_i$ by projecting out the first occurrence of $z$ from $\alpha$-projected suffix of $s_i$, as long as the instance set is not empty, and add it to $S_\beta$ (lines 21-22). When the support of $\beta$ in $S$ is obtained, the pair $\langle \beta : \text{supp}_\beta \rangle$ is recorded in the set $\text{STP}_\text{Supp}$ (line 23). Finally, we recursively call $\text{UpsSTP}$ taking $\beta$ as the current STP to generate longer STPs, and return all the STP-support pairs for $u$, obtained from both here and the return values of subsequent recursions (lines 24-25).

**Algorithm 3.** $\text{UpsSTP}(\alpha, S, \text{Pref}_\alpha, S_\alpha)$

1: $\text{STP}_\text{Supp} \leftarrow \varnothing$
2: find all the possible elements (topics) in $S_\alpha$ which can be appended to $\alpha$ to form a new STP, and record them in $E$;
3: for all $z \in E$ do
4: $\beta \leftarrow \alpha \circ z$;
5: $\text{supp}_\beta \leftarrow 0$;
6: $\text{Pref}_\beta, S_\beta \leftarrow \varnothing$;
7: for all $(i,j_0) \in S_\alpha$ do
8: find all the documents as instances of $z$ in the projected $s_i$, i.e., \{ $td_j \mid td_j \in s_i \land j \geq j_0 \land \exists p_i(z, p_i) \in td_j \}$, and record in order each position $j$ in $J$ together with the corresponding probability $p_j$;
9: $\text{Pref}_\beta \leftarrow \text{Pref}_\beta \cup \{(i,0,0)\}$;
10: $j'' = \max \{ x \mid x < j \land \exists (i, x, p) \in \text{Pref}_\beta \}$;
11: find $(i,j''',p''') \in \text{Pref}_\beta$;
12: for all $j \in J$ do
13: if $\text{Pref}_\alpha = \varnothing$ then
14: $p \leftarrow p_3 + (1-p_3) \times p''';$
15: else
16: $j' = \max \{ x \mid x < j \land \exists(i,x,p) \in \text{Pref}_\alpha \}$;
17: find $(i,j',p') \in \text{Pref}_\alpha$;
18: $p \leftarrow p_3 \times p_1 + (1-p_3) \times p'$;
19: $\text{Pref}_\beta \leftarrow \text{Pref}_\beta \cup \{(i,j,p)\}$;
20: $\text{supp}_\beta \leftarrow \text{supp}_\beta + p/|S|$;
21: if $J \neq \varnothing$ then
22: $S_\beta \leftarrow S_\beta \cup \{(i, \min \{ j \mid j \in J \})\};$
23: $\text{STP}_\text{Supp} \leftarrow \text{STP}_\text{Supp} \cup \{(\beta : \text{supp}_\beta)\}$;
24: $\text{STP}_\text{Supp} \leftarrow \text{STP}_\text{Supp} \cup \text{UpsSTP}(\beta, S, \text{Pref}_\beta, S_\beta);$
25: return $\text{STP}_\text{Supp}$;

It is worth noting that different from existing frequent mining algorithms, the pruning strategy cannot be applied here to speed up our approach. For the classical PrefixSpan [29], thanks to the downward closure property for the constant constraint on support, if a pattern is not frequent, all of its extensions are certainly not either. Even for SLPMiner with length-decreasing support constraints [30], there are still some properties supporting pruning rules. However for our URSTP mining, the criterion involves both the global support of an STP and the relative rarity of the STP for a local user. In each pattern-growth process for a specific user, we can just get the local support on sessions associated with that user, but not the global support on all sessions, so it cannot be determined whether the current STP is a URSTP, let alone those STPs extending it. Moreover, the computation of the relative rarity requires the support values of all STPs for all users, so even if an STP for a user is not a URSTP, the support value is still significant for other STPs and users,
no matter how large or small. Therefore, we have to derive all the possible STPs and retain the support values in each execution of \( \text{UpsSTP} \). Certainly, a length constraint can be imposed as an input parameter, but we do not adopt it here for a general algorithm.

### 4.2.2 Approximation Algorithm

In the algorithm above, the computation of the exact probability \( P(\alpha \subseteq s) \) is a key step, which involves interdependent instances of \( \alpha \) in the session \( s \) and thus complicated. It is a natural idea to find and use an approximate value instead to simplify the computation. That will make a good trade-off between the efficiency of the STP candidate discovery algorithm and the accuracy of the support values.

To this end, we assume that each STP actually occurs in a session at most once. That is reasonable to some extent since low-probability topics have been eliminated from the uncertain topic-level sequence, and furthermore, the frequency of an STP should be reflected by the multiple occurrences in different sessions, but not in the same session. Hence, even if an STP has more than one instances in a session, we can choose the one with the largest probability as the representative occurrence of the STP in the session. Based on this idea, an approximation formula can be given as follows.

\[
Pr(\alpha \subseteq s) \approx \max\{ P(\bar{\alpha}) \mid \bar{\alpha} \in \Psi_{\alpha,s} \}
\]

(7)

where \( \Psi_{\alpha,s} \) denotes the set of instances of \( \alpha \) in \( s \), and \( P(\bar{\alpha}) \) is the occurrence probability of an instance \( \bar{\alpha} \), which is the product of the probabilities of all topics in \( \bar{\alpha} \). If there is no instance of \( \alpha \) in \( s \), i.e., \( \Psi_{\alpha,s} = \emptyset \), we assign the probability value with 0.

The approximation algorithm is named as \( \text{UpsSTP-a} \). Compared to the pseudocode in Algorithm 3, the modification is mainly on the probability computation process (lines 9-20), and is shown as follows.

1. for all \( j \in J \) do
2. if \( \text{Pref}_\alpha = \emptyset \) then
3. \( p \leftarrow p_j \); 
4. else
5. \( \text{Pref}_\alpha \leftarrow p_j \times \max\{ p' \mid \exists x.(i,x,p') \in \text{Pref}_\alpha \land x < j \} \);
6. \( \text{Pref}_\beta \leftarrow \text{Pref}_\alpha \cup \{(i,j,p)\} \);
7. \( \text{supp}_\beta \leftarrow \text{supp}_\alpha + \max\{ p \mid \exists x.(i,x,p) \in \text{Pref}_\beta \}/|S| \);

Notice that the meaning of the prefix triple set \( \text{Pref}_\alpha \) is also changed here. For each triple of the form \((i,j,p)\), \( p \) is no longer the exact probability of \( \alpha \) occurring in the first \( j \) elements of \( s_j \), but its approximate value, i.e., the maximum probability of instances of \( \alpha \) ending with the \( j \)-th position of \( s_j \). That also affects the computation of \( P(\beta \subseteq s_j) \) in the last line, which needs to take the maximum probability value recorded in triples for the projected \( s_j \).

### 4.3 User-Aware Rarity Analysis

After all the STP candidates for all users are discovered, we will make the user-aware rarity analysis to pick out URSTPs, which imply personalized, abnormal, and thus significant behaviors. That is implemented by the subprocedure \( \text{URSTPMiner} \) shown in Algorithm 4.

It transforms the set of user-STP pairs into a set of user-URSTP pairs, with the set of user-session pairs and two thresholds, the scaled support threshold \( h_{ss} \) and the relative rarity threshold \( h_{rr} \), as input parameters. At first, we get the set \( \Phi \) containing all the derived STPs for all users (line 2), and for each of them (denoted as \( \alpha \)), compute the global support \( \text{supp}_\alpha \) as a weighted average of its local support for each user (lines 5-9), and normalize it to a scaled value \( \text{scsupp}_\alpha \) according to Equation 2 (line 10). If the STP is globally rare checked by the threshold \( h_{ss} \), it will be recorded in a set \( \Phi' \subseteq \Phi \) (lines 11-12). After that, for each user \( u \), we calculate firstly the absolute rarity \( \text{AR}_u \) for all of her STPs according to Equation 3 as well as the absolute rarity for the user (lines 15-19), and then the relative rarity \( \text{RR}_u \) for those STPs globally rare and found for \( u \) according to Equation 4 (lines 20-21). Next, the locally frequent STPs are selected by the threshold \( h_{rr} \), each of which forms an STP-RR pair for \( u \) (lines 22-23). At last, the set of these pairs together with the key value \( \alpha \) is added to the set of user-URSTP pairs (line 24), which will be returned when all the users have been handled (line 25).

#### Algorithm 4. \( \text{URSTPMiner} \)

1. \( \text{User}_\text{URSTP} \rightarrow \Phi' \leftarrow \emptyset \);
2. get the whole pattern set \( \Phi \) from \( \text{User}_\text{STP} \);
3. get the number of sessions \( |S| \) from \( \text{User}_\text{Sess} \);
4. for all \( \alpha \in \Phi \) do
5. \( \text{supp}_\alpha \leftarrow 0 \);
6. for all \( (u : \text{STP}_\alpha) \in \text{User}_\text{STP} \) do
7. \( \text{find} \ (u,S_u) \in \text{User}_\text{Sess} \);
8. \( \text{if} \ \text{there exists} \ (\alpha : p) \in \text{STP}_\alpha \) then
9. \( \text{supp}_\alpha \leftarrow \text{supp}_\alpha + p \times |S_u|/|S| \);
10. \( \text{scsupp}_\alpha \leftarrow \frac{\text{supp}_\alpha}{\text{supp}_\alpha} \);
11. \( \text{if} \ \text{scsupp}_\alpha \leq h_{ss} \) then
12. \( \Phi' \leftarrow \Phi' \cup \{\alpha\} \);
13. for all \( (u : \text{STP}_\alpha) \in \text{User}_\text{STP} \) do
14. \( \text{STP}_\text{RR}_u \leftarrow \emptyset \);
15. \( \Phi_u \leftarrow \{u | \exists p.(u : p) \in \text{STP}_\alpha \} \);
16. \( \text{avgAR}_u \leftarrow 0 \);
17. for all \( \alpha \in \Phi_u \) do
18. \( \text{AR}_u \leftarrow \text{scsupp}_\alpha \times \text{avgAR}_u + \text{AR}_u/|\Phi_u| \);
19. \( \text{avgAR}_u \leftarrow \text{avgAR}_u + \text{AR}_u/|\Phi_u| \);
20. for all \( \alpha \in \Phi' \cap \Phi_u \) do
21. \( \text{RR}_u \leftarrow \text{RR}_u + \text{AR}_u \);
22. if \( \text{RR}_u \geq h_{ss} \) then
23. \( \text{STP}_\text{URSTP} \leftarrow \text{STP}_\text{RR}_u \cup \{(\alpha : \text{RR}_u)\} \);
24. \( \text{User}_\text{URSTP} \leftarrow \text{User}_\text{URSTP} \cup \{(u : \text{STP}_\text{RR}_u)\} \);
25. \( \text{return} \ \text{User}_\text{URSTP} \);

### 5 Experiments

Since the problem of mining URSTPs in document streams proposed in this paper is innovative, there are no other complete and comparable approaches for this task as the baseline, but the effectiveness of our approach in discovering personalized and abnormal behaviors, especially the reasonability of the URSTP definition, needs to be practically validated. In this section, we conduct interesting and informative experiments on message streams in Twitter datasets, to show that most of users discovered by our approach are actually special in real life, and the mined URSTPs can indeed capture personalized and abnormal behaviors of Internet users in an understandable way.

In addition, we also evaluate the efficiency of the approach on synthetic datasets, and compare the two alternative subprocedures of STP candidate discovery to demonstrate the tradeoff between accuracy and efficiency. As these
results do not serve directly for the real-world mining task, they are left to the Appendix.

5.1 Experimental Setup

We collect two Twitter datasets as real document streams, a general dataset and a special sports-related dataset.

To get the general dataset, we start from a famous user “SteveNash”, crawl 150 latest tweets and 50 randomly selected active friends of him through Twitter’s Rest API, and put these users in a waiting queue. Here, the activeness is determined by the total tweet number (not less than 150) and friend number (not less than 50). Then, this process is repeated for the users in the queue until 2000 users are collected, which realizes a breadth-first user traversal. The direct and indirect friends of the seed user spread over various kinds of fields, so the topics of these tweets are diversified. After removing those users with too high or too low publishing rates as well as very short and non-English tweets, the dataset contains 1950 users and 183960 tweets.

The special dataset is obtained in a similar way, except that the seed user becomes a sports journalist “WojVerticalNBA”. Most of his friends are closely connected to sports, such as journalists, players and commentators. To control the tweet contents, we remove the users irrelevant to sports according to the descriptions in their profiles. Consequently, the topics of tweets in this dataset focus on sports, but the subtopics are various and reflect user’s characteristics and roles. The dataset contains 955 users and 94943 tweets.

In the preprocessing phase, we use a public package of the Twitter-LDA model [39] in Github developed by the SMU Text Mining Group, with the topic number $K = 15$ and $K = 10$, respectively for the two datasets. In this model, each tweet is assumed to talk about only one topic, so each derived topic-level document just contains a single topic with probability 1. Although later computations are still feasible, the uncertainty degree is totally lost. We recover it by recording the topic values of each tweet at 10 iteration points after the burn-in period (1000 iterations), and computing the proportion of them to get probabilistic topics. We find 60% of tweets involve a unique topic, and others follow biased distributions. That implies convergence and coincides with the characteristics of short tweets. Then, the Topic Probability Threshold with value 0.3 is adopted to select representative topics, and sessions are identified through the Time Interval Heuristics with the threshold set to 5 hours.

Afterwards, STP candidates for all users are discovered by calling the subprocedure $\text{UpsSTP}$ with five parallel threads. Here, we restrict the STP length in between 2 and 4, as longer STPs are generally insignificant and hard to interpret. Then, we apply URSTPMiner on these STPs to mine user-aware rare ones. Notice that our target is to find special and abnormal behaviors of Internet users, which are intuitively in minority for the general population, so the effectiveness of our approach should be reflected by the quality of those URSTPs with topmost values of the relative rarity, as well as their associated users. To this end, we set $h_{ss} = 0.05$ and $h_{sr} = 0.3$ to get relatively strong conditions, and evaluate on a small but representative test set.

In addition, we also use the approximation algorithm $\text{UpsSTP-a}$ to replace $\text{UpsSTP}$, and carry out the two steps of mining for comparison. Very similar results of URSTPs are obtained and omitted here due to the page limit.

All the experiments were conducted on a desktop with Intel(R) Core(TM) 3GHz i5 CPU and 6GB RAM. The algorithms are implemented in Java, and run in command line with Java 1.7.0_79 on Ubuntu 12.04.

5.2 Quality of Associated Users

At first, we check whether the mined URSTPs by our approach (denoted as URSTP) are really associated to the users with special or abnormal behaviors. Apparently, it is very hard to obtain the exact ground truth of these users for the randomly crawled datasets. Here, we make a reasonable assumption that “verified” users in Twitter are more likely to have special and repeated behaviors than ordinary users, so they can be regarded as approximate ground truth of special users. But for the sports-related dataset, most of users are verified, and the user particularity is not obvious in a specific field, so the test here is only conducted on the general dataset. In all, there are 232 verified users (23% of the total number) by checking the profiles of users. As discussed above, we mainly concern a small fraction of users with topmost relative rarity values, so recall is insignificant, especially when the ground truth is approximate. Hence, we take precision@K as the evaluation metrics. For comparison, we also consider some alternative methods. The first one named URSTP-L is almost same as our approach except that LDA in the toolkit Mallet [22] is used to directly extract probabilistic topics. In addition, as baseline methods to solve this innovative mining problem, special users can be found by computing the relative rarity of a single topic $z$ for a user $u$, instead of an STP. The computational formulas are similar to Equations 3 and 4.

$$AR(z)_{u} \equiv \frac{\text{supp}(z)_{u} - \text{supp}(z)_{\neq u}}{|D_{u}|}$$

$$RR(z)_{u} \equiv AR(z)_{u} - \sum_{z' \neq z} AR(z')_{u}$$

where $D_{u}$ is the set of documents published by $u$. The topics are still extracted by Twitter-LDA or LDA, and the methods are named Topic-L and Topic respectively.

The precision results are shown in Table 3. We can see that our approach achieves a high precision (much higher than 23% by random selection), which certifies that the users discovered by our approach are indeed inclined to be special among general users. The existence of counterexamples against the assumed “ground truth” is reasonable and even expected, as special and repeated behaviors of ordinary users are probably abnormal and should be discovered as clues for further investigations. URSTP-L shows a little worse performance, as the inaccurate topic extraction by LDA affects the later computation process, but is still better than baseline methods. Moreover, the difference induced by the two topic models for URSTP mining is much smaller than that for simple topic mining. That indicates compared to individual topics, the sequential patterns can integrate the information in successive and interrelated documents, which weakens the effect of incorrect topic assignments, so we guess that our approach is not so sensitive to the
5.3 Quality of Mined URSTPs

Besides the users associated to mined URSTPs, the quality of these patterns themselves also needs to be validated. They can reveal those special and abnormal behaviors on the Internet concretely, and should be self-interpretable and consistent with tweet contents. Similar as above, we mainly pay attention to the topmost URSTPs in terms of relative rarity. Specifically, for each mined URSTP, we firstly examine top words of involved topics from the topic model, summarize a rough description of each topic, and see whether an abstract and reasonable understanding can be obtained by integrating the meanings of ordered topics in the pattern. Afterwards, we check the profile of the user associated to the URSTP and the original tweets published by the user, to form a concrete understanding of her behaviors, and determine whether it is consistent with the abstract understanding above.

Intuitive examples for this process from the two datasets are demonstrated here. Table 4 shows a part of top-10 URSTPs in the respective datasets, their associated users, scaled supports and relative rarities, and Table 5 lists some top words as well as simple descriptions of involved topics. The superscripts "g" and "s" on topics or users represent the results come from the general dataset or the special sports-related dataset respectively.

From the results of the general dataset, we can infer some personalized characteristics and abnormal missions of Twitter users. For the first two users who are verified, $u_{299}^{g}$ connects sports games with entertainment, so he is probably a sports advocate and tries to introduce pleasure to boring physical exercise; while $u_{1914}^{s}$ often talks about TV shows followed by family issues, so he may be responsible for sharing and commenting new TV shows which are mainly about family life. These conjectures are proved to be correct by user profiles, as the former comes from a fitness club in the Nike company and the latter calls himself the source of TV shows and movies. As to ordinary users, $u_{246}^{g}$ is likely to be a sports fan, since he often plays some games after buying corresponding equipments, and $u_{207}^{g}$ may be an Internet cosmetic salesman with the distinctive publishing sequence {health, products, buying}. In addition, $u_{107}^{g}$ has an interesting sequence {sports} game, family} with two seemingly irrelevant topics. A reasonable explanation is that the user regards his team as a family, so often quotes some life philosophy to encourage his teammates and harmonize the team atmosphere. These pieces of implicit information are indeed consistent with real situations reflected in the original tweets of the three users. Although these examples seem not particularly significant, if $u_{215}^{g}$ were about to do something criminal, $u_{107}^{g}$ were selling prohibited products, or $u_{107}^{g}$ often spread viewpoints harmful to social peace, it would be important and urgent to expose them automatically from the Internet. In these cases, our approach will be very effective and indispensable for real-life security surveillance.

For the dataset restricted to a specific field (sports here), there may be few abnormal behaviors, but our approach can effectively distinguish different roles the users play in the field, and find out those typical users for each role through their complete and repeated behaviors. We can see that the user $u_{895}^{g}$ often talks about NCAA matches, then expresses happy feelings several times, and finally recollect some match details. This publishing sequence indicates he is probably a basketball player in the NCAA league. $u_{26}^{g}$ and $u_{861}^{g}$ involve same topics, but the order is opposite. We can guess that the former is a news reporter who always publishes official broadcasts followed by the introduction of players, but the latter is just an ordinary fan who forwards
improve the mining algorithms mainly on the degree of parallelism, and study on-the-fly algorithms aiming at real-time document streams. Moreover, based on STPs, we will try to define more complex event patterns, such as imposing timing constraints on sequential topics, and design corresponding efficient mining algorithms. We are also interested in the dual problem, i.e., discovering STPs occurring frequently on the whole, but relatively rare for specific users. What’s more, we will develop some practical tools for real-life tasks of user behavior analysis on the Internet.

### Appendix

#### Synthetic Experimental Results

We utilize and extend the IBM data generator [2] to get probabilistic datasets, which mimic topic-level document streams.

As the first step, we imagine some users and create some sessions for each of them. Each session is a sequence of itemsets directly obtained from the generator, where each itemset can be regarded as a document and each item represents a topic. These topics are divided into two types, common topics and special topics. For 80% of users, we only assign common topics, while for the other 20%, both types of topics are involved, and each special topic is just for one user. These special topics result in globally rare, but locally frequent STPs, which can be found out manually and serve as the ground truth of URSTPs to be mined by our approach. Then, we assign a probability to each topic occurring in the dataset following a uniform distribution over (0, 1), and normalize these values to guarantee that the summation for each document is less than 1, controlled by a dummy topic in each document like [40]. The obtained sets of element-level probabilistic sequences conform to the topic-level document stream defined in this paper. Notice that the stream has already been divided into sessions, so the preprocessing phase is not required here, and the test will thus concentrate on the mining algorithms.

Initially, the user number is set to 50, the number of sessions for each user is picked from a Poisson distribution with the mean $\bar{m} = 100$, the size of each session (length of sequences) is also drawn from a Poisson distribution with the mean $\bar{q} = 5$, and the number of topics in each document (size of itemsets) is randomly chosen from $\{1, 2, 3\}$. In addition, the numbers of common topics and special topics are set as $K_c = 20$ and $K_s = 10$. All the reported results are averaged on five runs for each specified configuration.

Obviously, the values of the two thresholds would directly affect the accuracy of mined URSTPs. We determine the optimal values of them in terms of F1-measure via fixing one and changing the other. For the exact mining which invokes UpsSTP, the optimal values are $h_{ss} = 0.015$ and $h_{sr} = 0.01$. While for the approximate mining with UpsSTP-a, the former is same, but the latter becomes 0.05.

### 6 Conclusions and Future Work

Mining URSTPs in published document streams on the Internet is a significant and challenging problem. It formulates a new kind of complex event patterns based on document topics, and has wide potential application scenarios, such as real-time monitoring on abnormal behaviors of Internet users. In this paper, several new concepts and the mining problem are formally defined, and a group of algorithms are designed and combined to systematically solve this problem. The experiments conducted on both real (Twitter) and synthetic datasets demonstrate that the proposed approach is very effective and efficient in discovering special users as well as interesting and interpretable URSTPs from Internet document streams, which can well capture users’ personalized and abnormal behaviors and characteristics.

As this paper puts forward an innovative research direction on Web data mining, much work can be built on it in the future. At first, the problem and the approach can also be applied in other fields and scenarios. Especially for browsed document streams, we can regard readers of documents as personalized users and make context-aware recommendation for them. Also, we will refine the measures of user-aware rarity to accommodate different requirements.

---

**Table 6**

<table>
<thead>
<tr>
<th>URSTP quality</th>
<th>$\theta_5$</th>
<th>$\theta_10$</th>
<th>$\theta_15$</th>
<th>$\theta_20$</th>
<th>$\theta_{50}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-interpretability-g</td>
<td>4.20</td>
<td>4.00</td>
<td>3.73</td>
<td>3.55</td>
<td>3.23</td>
</tr>
<tr>
<td>Consistency-g</td>
<td>4.60</td>
<td>4.20</td>
<td>4.20</td>
<td>3.20</td>
<td>3.57</td>
</tr>
<tr>
<td>Self-interpretability-s</td>
<td>4.00</td>
<td>3.75</td>
<td>3.70</td>
<td>3.25</td>
<td>2.80</td>
</tr>
<tr>
<td>Consistency-s</td>
<td>4.30</td>
<td>4.50</td>
<td>4.26</td>
<td>4.45</td>
<td>3.82</td>
</tr>
</tbody>
</table>
Fig. 4. Values of precision, recall and F1-measure of the two algorithms with changing user numbers.

Fig. 4. For the exact mining, the precision varies between 0.90 and 0.98 and the recall varies between 0.86 and 0.95. They are both high and thus compelling. In addition, as the number of users increases from 40, recall shows an upward trend, while precision maintains a high value but declines moderately. The reason is that the patterns will become sparser with more users, and the discrepancy among users will get more obvious, so some insignificant but relatively rare STPs will also be found. As a trade-off metric, F1-measure is comparatively stable. The results for the approximation mining is a little worse, especially for precision and larger user scales, but still acceptable. The F1-measure is around 0.8 and tends toward stability.

To validate the scalability of the our algorithms, we aim at the most time-consuming subprocess which discovers and computes all the user-STP pairs, including the exact version UpsSTP and the approximate version UpsSTP-a. The baselines are chosen as the two Apriori-based frequent sequential pattern mining algorithms for probabilistic databases, Depth-First Exploration and Breadth-First Exploration [28]. We modify them to accommodate our problem, and compare the execution times of the four algorithms with changing values of the average session number $\bar{n}$ and the average session size $\bar{q}$ respectively. As no pruning strategies are applied here, the values of thresholds would not affect the time cost.

For the first case, we fix $\bar{q} = 5$, but change $\bar{n}$ for each user from 100 to 30000. For the second case, we fix $\bar{n} = 100$, but change $\bar{q}$ from 4 to 18. The results are shown on the left of Fig. 5 and Fig. 6. With the data size increasing, the performances of both UpsSTP and UpsSTP-a are almost stable, while for the Apriori-based algorithms, they decline sharply at $\bar{n} = 6000$ and $\bar{q} = 10$, and are respectively very sensitive to one parameter.

To make a finer comparison between our two algorithms, we amplify the lower parts of the two charts and show them on the right of respective figures. It is observed that each execution of this subprocess is just for one user, so when the user number increases, the time difference for the whole approach will become more and more evident, even with some extent of parallelism. Therefore, together with the results in Fig. 4, we can conclude that the two algorithms have their respective advantages. Which one is appropriate for the real task reflects a trade-off between mining accuracy and execution speed, and should depend on the specific requirements of application scenarios.

Fig. 5. Time costs of the four algorithms with changing session numbers.

Fig. 6. Time costs of the four algorithms with changing session sizes.

ACKNOWLEDGMENTS
This work is supported by the National Key Basic Research Program of China (973 Program, No.2013CB329305), the State Key Program of National Natural Science Foundation of China (NSFC-61232013) and the National Natural Science Foundation of China (NSFC-61402452).

REFERENCES


